TextQuest

Version 4.2

April 2013

Software for the analysis of texts

Contents

1. Introduction

1.1 Some history

Text**Q**uest was written for applications in the humanities and the social sciences. The first version named INTEXT (INhaltsanalyse von TEXTen – content analyses of texts) was developed in 1983 on an IBM-mainframe (IBM 3032 running under MVS) at the computer centre of the university of Münster/Germany and written in PL/1. In 1988 the original PL/1 programs were completly redesigned and rewritten an an MS-DOS version using C as a program language was published. These versions were designed and written by Harald Klein. The current version named **T**ext**Q**uest– text analysis software is written in C++ using wxWidgets.

Text**Q**uest version 1.x was written by Net.Sys GmbH, Ilmenau, Germany in 1999. The most modules from the former INTEXT were included in these versions. **T**ext**Q**uest versions 2.x and 3. were written by RF Techniques, Trinidad & Tobago. The new vocabulary comparison module and the category manager were added. A complete overhaul followed with version 4.0 in 2010, written by TT-Solutions, Achéres, France. This version allows to process texts encoded either in Latin-1 or UTF-8 encoding, and this version is available for MS-Windows and Apple Mac OS-X.

1.2 The manual

This manual was produced with TEX and LATEX. It is designed to help you to benefit from all features **T**ext**Q**uest offers. This manual was written using the orthography of British English. Like nearly all other manuals this one not free of errors, and maybe you find some descriptions and explanations annoying, clumsy, or otherwise not fulfilling your needs. If this is the case, please write your suggestions to the address mentioned below. Error corrections and improvements can only be made if you tell us what errors you found and what to improve. Please contact:

Dr. Harald Klein
Social Science Consulting
Lutherstr. 2
49082 Osnabrück
Germany

Tel/Fax: +49 541 18 19 492

http://www.textquest.de
e-mail: info@textquest.de

1.3 What TextQuest can do – an overview

- vocabularies: these are word lists, word sequences, and word permutations. All vocabularies can be sorted by alphabet and/or by frequency, compared with each other, reversed, and filtered by means of frequency, length, or occurence in an exclusion list (STOP-words).
- word list: a list of all strings that occur in the text together with their frequency.
- word sequence: like a word list, output are parts of the text containing x words, where x is variable. Word sequences can be sorted by the first or by the last word of the word sequence. If x takes the value 1, a normal word list is generated. Phrases like "United States" or "United Arab Emirates" can be counted.
- word permutations: like a word list, each string is combined with each following string as a two word sequence.
- reverse vocabulary: like a vocabulary, but the order of the characters is changed from left to right, so the first character in a normal word list is the last character in a reverse word list. *TextQuest* reversed is *tseuQtxeT*.
- search patterns in text unit: output is the search pattern and the complete text unit in which the search pattern occurs.
- cross reference: for each string the locations with all external variables are written to the output file. The locations can be formatted in multiple columns.
- vocabulary comparisons: two or more word lists, word sequences or word permutations can be compared in different formats. Also statistics are computed.
- content analysis: with powerful search patterns for single and multiple word coding, controlling of ambiguity and negation with log files or by interactive coding, adaptable negation algorithm. Instead of developing own category system, the ones delivered can be used.
- readability analysis: there are currently 78 formulas for different languages (mostly for English, French, German, Spanish and Italian). The syllable count algorithm is language independent and can be adapted for other languages. Also other statistics are computed.
- concordance: the context of search patterns is written to a line with variable length (KWIC/KWOC).
- style analysis: a special form of a content analysis
- data management: backup and restore the files of a project to and from another storage device.

1.4 New in TextQuest version 4.2

Some extensions are new in version 4.2:

- word list: statistics now include frequencies of frequencies and length of entries as well as hapax legomena in absolute and relative values. Statistics of Somers D and Tuldava's T were transferred to the readability analysis.
- word sequences: the range of strings now has another meaning. In older versions e.g. the value 4 meant that all sequences of exactly 4 strings are written to the output file. In version 4.2 this value means that sequences of 1 (this is the word list), 2, 3 and 4 strings are written to the output file.
- readability analysis: new formulas were added for Italian, and also Tuldava's language independent formulas and TTR values based on words and on all strings are new.

1.5 Installation of TextQuest

The installation of **T**ext**Q**uest for **MS-Windows** is done with a setup program that installs all files into a directory that the user specifies during the installation process. Also the manual is stored in this directory, as well as the sample files (texts and category systems).

If you want to remove **T**ext**Q**uest from your system, you can do so with the system control software section and select **T**ext**Q**uest there. All installed files will be deleted, the files you created will be kept.

The **Mac OS-X** version is delivered as a *.DMG file that can be mounted. You will find the `tq.app` and can start it by clicking on it.

If you are asked for your user name and serial number, you can enter this with the first execution. If you have a time limited license, the counting of days starts with the first execution. About 10 days before the license period expires, you will be notified.

1.6 Installation problems under MS-Windows

The most of these problems occured in earlier version, and a lot of these problems are already fixed. If the program crashes, fixes can be:

- You use the sample files and write the results into the installation directory. If you have administrator rights, this is okay, but if you use a restricted user account this might have caused the crash. Store your data in another directory/folder than the installation directory.

- **T**ext**Q**uest runs, but some files are reported to be missing although the files exist. Do the following: move your mouse to the desktop icon of **T**ext**Q**uest and right-click on it. Choose the tab *properties* and look for the line under the specification of the directory/folder where **T**ext**Q**uest is installed, this line is empty. Insert the name of the installation directory/folder in this line, you can copy this information from the line above but do not copy the file name. Mostly this line looks like this: c:\program files\textquest
- missing DLLs: can be mailed to you or you can download these from http://www.microsoft.com

1.7 The philosophy of TextQuest

Text**Q**uest is a kind of toolbox with a lot of analyses provided. Nevertheless the applications can be used also for other purposes than originally intended and described in this manual. The use of **T**ext**Q**uest and its design to use the facilities **T**ext**Q**uest provides are explained in this chapter.

open system: **T**ext**Q**uest is an open system. The texts are stored in files that can be used by **T**ext**Q**uest, but also by other programs. Most files are plain text files or HTML-files, control sequences are only used for emphasis purposes like bold face and are described in the relevant chapters. The format of the files can be found in chapter 11 on page 141.

project name: It is used to generate the file names using the built-in system. All texts belonging to one project are stored in the same directory. Therefore the project name may contain drive and directory names. Thus it is possible to work with different options (e.g. sort order or negation orders) in different projects.

file names: Due to its design as an open system quite a number of files are generated. A system of files names is a built-in feature in **T**ext**Q**uest and you are advised to use it. The advantage is that file names – derived from the project name – need not to be specified by a open/close file dialogue. The generated file names are shown, and you can alter them if you want.

information messages: are written to the screen and to a log file, date and time stamped. It is not necessary to copy results manually from the screen.

changing of TextQuest-tables: Due to its design as an open system **T**ext**Q**uest can be altered to one's own needs, e.g. the sort order table, the lists of indicators for negations, the syllable table and the indicators for foreign words are plain text files and can be changed or adopted to other languages.

samples: The text file can be processed completly, or a sample can be defined on the basis of external variables. Then only the text units are processed which are selected. The sample has to be defined before performing an analysis. In most analyses one can chose between processing the whole text or the defined sample.

language independence: Language specific files like the sort order table can be adapted to the language of the text, also multiple characters are possible.

1.8 Getting started – working with projects

Projects help you to organise your work. At first you need a file that contains the text you want to analyse. This file must be in a format that can be processed by **T**ext**Q**uest, there are several formats available. Some require pre-editing and segmentation of the text into text units, others like the line, paragraph, and the page format do not require this. For some applications like word lists these formats are sufficient, and plain text files can be processed easily. For details see chapter 3.4 on page 24.

Before you start any analysis, you must go to the project menu and select a file. The name of this file is used to derive many other file names.

A project has many specific features that are used in the analyses, and the features are set in the project menu. These include the place of the files, the languages used, and language dependent items like sort order tables, negation indicator lists, exclusion lists etc. The reason for this design is that all information performed on one file is stored in a directory, and different projects can be kept separate and do not mix.

The following files must be copied to each directory where texts that belong to one project are stored:

analysis	file
exclusion list	*.exl (english.exl, deutsch.exl, francais.exl)
definition files	*.def

1.9 TextQuest in networks

There is no special network version of **T**ext**Q**uest. If **T**ext**Q**uest is installed in a network, internal files must be accessible for the users, some of them must have the right to be copied by the user to a local drive. The following table gives an overview which files have to be considered:

*.def	copyable
.exc	copyable

1.10 The files of TextQuest

After the installation you find many files in the installation directory of **T**ext**Q**uest. The default directory is c:\program files\textquest under MS-Windows. The meaning of the files can be derived from the file extensions. If you change the name of output files, the result menu will not work. This manual contains information about the meaning of the file contents, and there are also applications that generate or alter these files.

The `refo*.def` files contain the legal combinations of vowels of a language. The files are used for readability analyses. This file is a plain text file and can be adapted for many languages.

The `*.def` files define defaults for one or several applications. Their meaning is as follows:

- ab.def – abbrevations for the automatic separation of text into grammatical sentences.
- sort.def – defines the collating sequences for sorting, e.g. umlauts or letters with accents or diactritics. It is used by the different vocabularies and their comparison(s), the cross references, and sorting.
- neg-pre.def – indicators for negation that are searched before a search pattern. This file is used in a content analysis.
- neg-post.def – indicators for negation that are searched after a search pattern. This file is used in a content analysis.
- refo.def – used in a readability analysis. It contains all valid diphtongs and vowels of a language, starting with the longest (in characters).
- fworte.def – used in a readability analysis. It contains all indicators for foreign words and is used for the calculation of the TRI readability formula.
- dalechal.def – a word list for the Dale-Chall formula (1948) for a readability analysis.
- dale.def – a word list by E. Dale for the Dale-Chall formula (1983) for a readability analysis.
- spauld.def – a word list for the Spaudling formula (1958) for a readability analysis.
- spache.def – a word list for the Spache formula (1948) for a readability analysis.
- bamvan.def – a word list for the Wiener formulas of Bamberger and Vanecek for a readability analysis.
- prepos.def – prepositions for the calculation of Dickes/Steiwer and Tränkle/Bailer readability formulas.
- konjunkt.def – conjunctions for the calculation of Dickes/Steiwer and Tränkle/Bailer readability formulas.

You can keep all `*.def` files in your project directory. If these files do not exist there, the `*.def` files stored in the installation directory will be used instead.

The names of the input and output files are derived from the project name, the following file extensions are used:

ext	application	content
def	project	definition files
exc	project	list of excluded strings
log	project	file with rapport of the results
prj	project	file with data of the project (internal use)
sam	project	definition of the sample for the project
itx	project	system file
txt	project	raw data
sco	readability analysis	control of syllables
fwp	readability analysis	rapport file of foreign words
dic	content analysis	category system (file of search patterns)
lab	content analysis	label file (category labels)
clg	content analysis control	coding control log file
ctx	content analysis control	coded text units
dse	content analysis control	multiple search patterns
ntx	content analysis control	negated text units
otx	content analysis control	overlapping text passages
utx	content analysis control	uncoded text units
tab	content analysis results	counters in a content analysis
vec	content analysis results	codes in a content analysis
sis	concordance	unsorted concordances
ssc.html	concordance	sorted concordances by code
ssa.html	concordance	sorted concordances by alphabet
sit	concordance	unsorted search patterns in text units
sis.html	search patterns in text unit	sorted search patterns in text sorted units by alphabet
sst.html	search patterns in text unit	search patterns in text units sorted by alphabet
wb	vocabulary	word list
ws	vocabulary	word sequences
wp	vocabulary	word permutations
wbf	vocabulary	word list sorted by frequency descending
wsf	vocabulary	word sequences sorted by frequency descending
wpf	vocabulary	word permutations sorted by frequency descending
xrf	vocabulary	sorted cross references
vcp	vocabulary comparison	results of a complete vocabulary comparison
ttr	vocabulary growth	TTR dynamics

2. TextQuest– an overview

2.1 The input files

Text**Q**uest is an open system: all analyses write files that other applications can use as input files. Some applications depend on each other, you find more information in the chapter structure of **T**ext**Q**uest.

A plain text file is always required. If you want to perform content analyses, concordances, or search patterns in text unit, a file of search patterns is required, and for content analyses a file of category labels is necessary. Content analyses can be performed using one of the standardised category systems that are part of **T**ext**Q**uest, otherwise you must provide these files:

- always: a file with the texts. There are several input formats available, details see chapter 3.4 on page 24.
- content analysis, concordances, search patterns in text unit: a file with the search patterns – called the category system. You can use the category manager to create and maintain one.
- content analysis: a file with the category labels for this category system, this file is also written by the category manager.

If your texts are organised in more than one file, these have to be merged into one single file. The file of the search patterns and the file of the category labels can be generated interactively (menu content analysis, submenu build category system), or using an editor that writes plain text files. If you use a text processor (e.g. MS-WORD), save the file as a text file with line breaks in either Latin-1 or UTF-8 encoding.

2.2 The output files

The results are written to the appropriate output files, they are used by **T**ext**Q**uest or they can be processed by other programs. In most cases they are in plain text format, otherwise control sequences or the like are described. The file formats are documented in chapter 11 on page 141.

2.3 Interfaces to other programs

Text**Q**uest can generate setups/scripts for the following software packages: SAS, SPSS, and SimStat. There are also converting programs to use other text analysis programs.

2.4 Working with TextQuest

With the help of the provided example files the most important text analyses can be performed. Follow this guide and you will get some experience how **T**ext**Q**uest works. At first you specify the project name so that **T**ext**Q**uest knows the data you want to work with. There are the following example files:

language	contents	file name	format
English	articles of the New York Times	nytimes1.txt	control sequences
English	speech on foreign policy by George W. Bush 2000	bush.txt	paragraph or line
English	speech on foreign policy by Al Gore 2000	gore.txt	paragraph or line
English	speech on foreign policy by John McCain 2000	mccain.txt	paragraph or line
English	comments on injuries on sport	sport.txt	control sequences
English	comments on injuries on sport	sp-fixed.txt	column format
German	personal ads	contakt.txt	control sequences
German	personal ads	qual.txt	control sequences

Each application starts with the settings window where you can specify the file names and the options. The file names always take default file names derived from the project name, so generally you do not have to change them. If you do, a file open dialog appears.

After pressing the OK-button, the application starts. Look at the counter of the text units in the window in the left lower corner. When **T**ext**Q**uest has finished the application, a window called application statistics window opens and shows the statistics the application generated, e.g. how many text units were processed and which options were used. You can see the results by changing to the resuts menu, an editor opens the appropriate file with the results. Some applications allow interactive working with the data, so you have other working windows there. If an error occurs, an error message is displayed explaining what happened and how to proceed.

3. Preparing the text

The first problem that often occurs is how one can convert a text from Microsoft-Word or another text processing program into a format that **T**ext**Q**uest can process. **T**ext**Q**uest cannot read files in MS-Word format (file extension `*.doc, *.docx` or `*.rtf`), so you store your file as a simple text file.

If you save a file, alter the format in the last line (at the bottom of the page) into text format. If you press enter, you can tick boxes for adding carriage returns (CR) or linefeeds (LF). Tick both boxes and save the file which often changes its file extension to `*.txt`. This works for most text processing programs.

Handling PDF-files can be complicated. Within the newer versions of Adobe's Acrobat Reader you can also save the text of a file, but you have to control and edit the output file. Therefore test for yourself how the text file looks like; other working techniques maybe more efficient.

In most cases you can mark all text of a PDF-file just using the strg-a button. However, an author can disallow this feature, so that your only chance is to print the document and to scan or to type it.

There are also programs that can extract text out of a PDF-file and write it to a plain text file. If this does not work, the contents of a PDF-file may origin in a scan and therefore stored as a picture. If this is the case, you can try to extract the picture from this file and process this with an OCR (Optical Character Recognition) program. Or, as already mentioned aboved, print the document and scan or type it.

The last solution is to type in the text yourself. This is often the fastest way to prepare texts, especially if the quality of the paper is bad. Please have in mind that newspapers are printed on low quality paper and therefore scanning this material and processing it with OCR-software will result in a low recognition rate lower than 99 % (this means you can expect 15-20 errors per page). If you want to benefit from encodings like UTF-8, you can use an editor (e.g. TextPad) to save a text file in UTF-8 encoding.

Text**Q**uest expects a file that consist of external variables of the text and the text itself. This is the reason that the text – called raw text – is separated into text units. External variables are assigned to each text unit, up to 50 are possible. The meaning of the external variables depends on the goal of the analysis. The units of text and analysis must be identical. Within a text unit no value of any external variable can change. The values of the external variables are dependent on the input format you use.

The following regulations must be followed:

- The system file is the basis for all further text analyses.
- The text encoding must be either Latin-1 or UTF-8.
- The maximum length of a text unit is 100,000 characters.
- The maximum size of a text file is only dependent from the mass storage device available (free space on the hard disk).
- The more external variables are used and the longer they are, the bigger the system file will be.
- At least 1 and at most 50 external variables must be defined. Some input formats allow 1 or 2 external variables only that are mostly generated automatically.
- If you use SimStat to analyse the content analysis data statistically, the external variables must not contain commas.

If the text consists of several files, these must be copied into one single file. And this is the organisation of a system file:

variables	1. external var	last external var	text (variable long)
1. text unit			The text starts here.
2. text unit			This sentence may be very long.
3. text unit			Or short.
4. text unit			But not more than 50,000 strings in a text unit.
5. text unit			Otherwise there will be no word list possible.
n. text unit			That's all, folks.

You must decide the following:

- What is the definition of a text unit?
- How many and which external variables are necessary for the planned analyses?

The definition of a text unit and its external variables are closely related. Their use, restrictions, and examples are described on the following pages.

3.1 The definition of external variables

External variables represent attributes of the text, e.g. the date when the text was published, the medium (e.g. a newspaper), a running number, and so on. One external variable is at least necessary, up to 50 are possible. Each external variable may consist of 10 characters, letters and digits may be mixed. Numeric external variables ease statistical analyses, whereas non-numeric external variables (e.g. words, abbrevations) improve the readability of cross references and concordances. Each external variable must consist of at least one character. Commas should not be used within an external variables. The values of each external variable can **not** change within one text unit. The values of the external variables are controlled by inserting control sequences into the text (control sequence format) or their position on columns on a line (column format). All other formats work with predefined external variables, e.g. line numbers, paragraph numbers, or page numbers.

Restrictions with external variables:

- up to 50 external variables are possible, at least one must be defined.
- only in control sequence format or in column format you can define external variables, all other formats have predefined external variables, e.g. line counters.
- The values of the external variables are separated by dashes within the control sequence format. In column format they are in a fixed place on a line, e.g. in column 1 to 17.
- The maximum length of each external variable is 10 characters. Using control sequence format the length of an external variable is variable, in column format the length is fixed.
- Each external variable can consist of all characters except Tilde (~), number sign (#), dash (-) and the vertical bar (|, ASCII-value: 124). Blanks within the external variable are possible, case folding within the external variables is always disabled as well as the compression of multiple blanks. (`p. 5` is not identical with `p.5`, `TIME` not with `Time`).
- The first control sequence using control sequence format must consist of initial values for **all** external variables. Using column format every line must contain the values of all external variables on the same columns of each line. All other input formats work with predefined automatic external variables.

3.2 Examples: text units and external variables

Content analysis is an empirical hypothesis testing research method. Therefore the definition of a text unit must follow the hypotheses. The following examples show different applications.

1. example: coding of open ended questions

If more than one open ended question in surveys is to be analysed, numbers for the interviewed persons and the questions are necessary, because after the coding the coding results have to be merged to all the other variables. The text unit is the answer to one open ended question. If the questionnaire consists of five questions, five text units for each interviewed person exist. If other variables (e.g. gender, age, place) are taken into account (e.g. for filtering/sampling), these are also necessary external variables.

2. example: analysis of newspapers/magazines

The most used text unit analysing printed media is the article. Necessary external variables are the name of the medium, the day of print, and a running number of the article within the issue. Also variables like place or size of the article may be useful.

In an analysis about the coverage of environment issues the following external variables were used: the name of the paper, the date, the column, the page within the column, and typographical specialities like photos, comment etc. (Kramer-Santel 1994).

3. example: readability analysis

Readability analyses can only be performed if the sentence is defined as a text unit. Also the implications of the used formulas, e.g. language and text genre, must be considered. Only one external variable is absolutely necessary: the sentence counter. If several text sources are to be compared, more external variables must be defined if these are used for a comparison.

4. example: literary science, e.g. style analysis

Literature researchers are often interested in the vocabulary of texts and which period or genre it belongs to. Text units may be chapters, paragraphs or sentences. A chapter as a text unit may cause problems because the maximum length of a text unit is 100,000 characters (approx. 45 pages). More practical are paragraphs as a text unit, and author, book, chapter, and paragraph are useful external variables.

If a comparison of several books of one author is the goal, the sentence should be the text unit, useful external variables are book, chapter, and sentence. Also the page number can be an external variable, but it might change its value within one text unit, so a page number should indicate where the text unit started.

5. example: television news

A news item is the suitable text unit for the analysis of television news. External variables are the TV station, the date and the current number of the news item. Also technical variables like length in seconds, photos, and type of presentation (e.g. interview, film) can be external variables. This study was done with INTEXT as a Ph.D. thesis (Klein 1996).

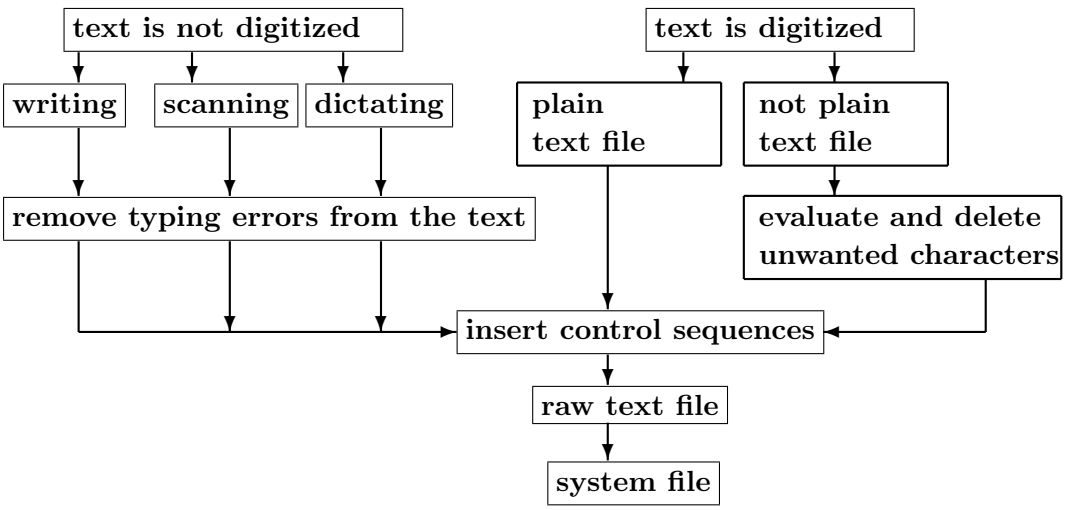
6. example: personal advertisments

If the objective is to find out whether there are differences in gender using personal advertisments and amongst different papers, necessary external variables are the name of the paper, the date of issue, and a running number of the advertisment, also external variables are necessary for the gender of the person who advertised and what gender the desired person has. The last external variable describes whether the person is writing of herself, the person that he/she is looking for, and how the relationship shall looks like. The advertisments must be separated into several text units depending on what is described. This study was done with INTEXT in 1988 (Giegler and Klein 1994).

3.3 Converting of digitised text data

There are several ways how text can be digitised, the following figure shows the working steps between data acquisition and generation of a system file:

Figure 2: Digitalisation of texts



The conversion of data into a digitised format can be achieved in four ways:

- typing of the text (keyboard): The texts are normally entered via keyboard into the computer or read by a scanner. Manual typing takes a lot of time, and it also requires principles for typing. Also control sequences must be inserted into the text while typing or after typing.

- scanning of text: If the texts are printed, you can use a scanner and OCR (optical character recognition) software. A scanner works like a photo copying machine, just the image of the page is stored as a picture in a graphic format. This file is processed by OCR-software and converted into text. Depending on the quality of the text the characters can be recognised more or less reliable. Good OCR programs have a recognition rate better than 99,9 %, that means each page contains one or two errors. Editing is required, and that has the advantage that one gets familiar with the text.

- dictating of text: If a text is dictated, the speaker's voice must be trained, this requires some time. Most systems can only listen to one speaker. Dictating speed is rather fast, but also errors occur. Both scanning and dictating require a high recognition rate, also editing the dictated text is necessary. And speaking precisly to gain a high recognition rate is quite exhausting and often limited to one hour or less.

- converting a text into a format that can be processed by **T**ext**Q**uest: if a plain text file exits, one can inserted the control sequences to set the values of the external variables, or if one does not need external variables, one can use one of the pre-defined formats (e.g. line format) to analyse texts without big effort in editing the text. If the text is stored in a file in plain text format (also known as ASCII or ANSI-file, you can use either Latin-1 or UTF-8 encoding), control sequences for the separation of the text into text units have to be inserted.

  If the text contains unwanted parts (e.g. printing characters, commands, graphics etc.), one has to remove these unwanted parts. Formatting commands get lost though, because they are not necessary for an analysis in most cases. This may require very time consuming editing work, or one must use software to do it (and maybe you have to write this software yourself).

During the preparation of the text regulations for the treatment of characters that do not belong to the English alphabet must be considered. This is easy with languages that use latin characters. One problem although is the representation of other characters, for example é, É, æ, ô, ò, ł, ÿ or ñ. It depends on the active code page which characters you can enter directly via the keyboard. Please note that not all characters are available for all code pages.

Languages with a non-latin alphabet, e.g. Greek, Russian, Arabic, or Chinese are much more difficult. Other software or working techniques have to be applied. Languages based on syllables (e.g. Chinese or Japanese) can be coded with multiple character sets. Future versions of **T**ext**Q**uest will support syllable based languages using full Unicode (UTF-16) encoding.

Other problems are characters with accents or diacritics, e.g. in French or Italian – or language specific characters like ch and ł in Spanish – that are letters. These problems however are dealt with a UTF-8 encoding of the text in **T**ext**Q**uest version 4.0 or higher (all versions after 2010).

Another point one has to think of is whether typographic variations are important and therefore kept in external variables (e.g. boldface, font size) or in the text. If the latter is the case, it is possible that they can be used as a part of a search pattern (word co-occurence).

Pre-editing of the text may also cause problems. That is the marking of phenomena of and in the text with defined character (combinations), e.g. bold face, categories etc. An application is e.g. that special categories are to be analysed and these are marked during text preparation. Pre-editing is the most used working technique for qualitative computer aided content analysis. One reason is that coding with search patterns is based on strings, and not on words in a grammatical sense.

If the text contains phenomena that are important for defining search patterns, one has to make up one's mind how to mark them, e.g. roman numbers which can be words. Also strings that start with numbers but are words might be important. Look at the following examples:

How are football results to be written? 5:2 or 5 : 2 or five two? 5:2 is one word, the second solution consists of three word - because all characters are separated by blanks. This has effects on the caluclation of text homogeneity measures like the TTR. Or what about compound words like client/server-technology. Or should it be written client-server-technology? Or client/server technology? If you have these phenomena in a text, you must define how you handle these.

Please have in mind that punctation marks (full stops, exclamation or question marks, commata) follow the words immediately without a blank between them. After punctation marks a blank must follow because otherwise long words can be the result, and that might cause problems for the software. Hyphenation probably causes problems, please avoid it. Programs like **T**ext**Q**uest cannot distinguish between the hyphenation symbol and dashes.

Today the typesetting is done by computers, and this means that the texts are already stored in files for immediate use. The sources are different, one problem might be unwanted characters like HTML or XML tags.

Word processors are mainly PC based (e.g. MS-Word, Open Office) or type setting systems like TEX/LATEX that work with commands. Common for both is that files can consist of the text and additional information, e.g. printing commands/characters. This information can be used for the external variables, and after using this information it has to be deleted. Therefore it is necessary that software can convert machine readable texts into a raw text or into a system file format using the information provided to generate text units and external variables.

The text step is the transformation of a digitised into a raw text format. This is a format that can be converted directly into a system file. **T**ext**Q**uest offers six raw text formats.

A digitised text does not mean that it can be converted into a raw text or system file format without some editing work. Control sequences to separate the text into text units and to set the values of the external variables have to be inserted. The next chapter describes the details.

3.4 Building a system file

The system file is the basis for all other analyses and requires a text in one of 6 formats (called raw text formats), described in this section. Also regulations for the writing have to be considered.

3.4.1 Regulations for writing

Words in the sense of the program are all characters surrounded by two blanks (or other separators). Multiple blanks are compressed to one blank while generating the system file.

Some punctation marks and special characters (e.g. `.,;:()` etc.) should be defined as own words, because otherwise words occur e.g. in a vocabulary with these characters. In a content analysis the search patterns and their coding is not biased, especially if the search patterns are in infix position. If you don't want to be characters treated as single words, just delete the characters that you don't want. These characters are only separated from a word if they occur at the beginning and/or the end of a word, but not within a word.

The following example demonstrates the problem: the search pattern is ' politic', so all strings are to be coded that start with `politic`, e.g. politics, political, or politician. But if a text like `Politics (political science) ...` occurs, then the string `political` will not be coded, because it starts with a bracket.

In most cases the regulations for typing texts with a type writer are sufficient. Hyphenation is to be avoided. No problems occur when there are dashes at the end of line, but errors – especially when generating a word list or its derivates – occur if dashes are at the end of a line, e.g. *pre- and post-editing*.

It is also possible to separate characters from strings and treat these characters as strings, that is important when performing a content analysis with search patterns. Separation of characters is the default.

3.4.2 Raw text formats

There are 6 raw text formats available that show you the use of the different raw text formats. The files `bush.txt, gore.txt` and `mccain.txt` can be used for all automatic raw text formats (line, paragraph, page).

control sequences format using control sequence format the control sequences within the text indicate the change of the values of external variables and separate the text units from each other. Only the values of the external variables that change their values have to be specified. `contakt.txt, nytimes1.txt` and `sport.txt` are sample files.

column format Using column format all external variables have to specified in *each* line. As long as the external variables have the same values, the text following the external variables belongs to the same text unit. `sp-fixed.txt` is a sample file. Specify one external variable and starting in column 1, the text starts in column 4.

line format Using line format every line is a regarded as a text unit. The line counter is the only external variable.

paragraph format Each paragraph is defined as a text unit. Paragraphs are separated by a blank line (CR/LF CR/LF). The paragraph counter is the only external variable.

page format Using page format each line is a text unit. There are two external variables: the page counter and the line counter. After x lines – this value can have a maximum value of 32767 – the first external variable is incremented.

sentence format Using sentence format each sentence is a text unit. The raw text file is split into grammatical sentences. The characters .?! are used as sentence delimeters, decimal points and abbrevations points are recognised. The file `ab.def` contains a list of most use abbrevations that are recognised, you can edit the file for your own purposes. A check of the correct splitting is however necessary.

3.4.3 Regulations using control sequence format

A control sequence looks like: $1(Times-980909-finance-1). If the $-symbol occurs in the text, you must change it, e.g. into USD.

Control sequences must always start with a $, and the values of the external variables of the **following** text unit are specified. Control sequences separate text units. The external variables are numbered in ascending order without gaps, starting with 1. The first control sequence at the beginning of the file of the raw text must contain values for **all external variables**. The following control sequences only have to contain the values of the external variables that change their values. If more than one external variable is changed, you must start the control sequence with the lowest variable number and specify the values of all others until the highest one. The values of the external variables are separated with a dash. If only one external variable is affected, the number of the external variable has to be specified after the start character $.

The following pages show examples.

Text**Q**uest does not change the values of the external variables, e.g. multiple blanks are not converted into a single blank. The values of external variables and their meaning (values and variables) is up to the user.

The following tables gives an overview of the available raw text formats:

format	external variables	type of external variable.	logic
variable	1-50	user defined	control sequence
fixed	1-50	user defined	fixed columns on lines
line	1	line counter	automatic
paragraph	1	paragraph counter	automatic
page	2	page and line counter	automatic

1. example: coding open ended questions

1. control sequence: $1(030295-1-1)

The external variables have the following values:

nr	variable	value
1	date	030295
2	number of person	1
3	number of question	1

The next control sequence only has to contain the values of the external variables that change their values. The control sequence for the next question is $3(2). The values of the first two external variables do not change, the value of the third external variable is set to 2. Here is an example for profession, preferred television program and washing powder of three persons:

```
$1(130994-46-1) electrician
$3(2) Cross roads, Rich man poor man, Dallas
$3(3) Persil
$2(47-1) house wife
$3(2) Sesame street, Falcon Crest, Coronation street
$3(3) Ariel
$2(48-1) shop assistant
$3(2) Open university, Sky news, Match of the day
$3(3) Dash
```

2. example: analysis of printed media

There are two examples for the analysis of printed media. This is the first example where only the necessary external variables are used.

1. control sequence: $1(Time-030295-1)

The external variables have the following values:

nr	variable	value
1	medium	Time
2	data	030295
3	number of article	1

The next control sequence only has to contain the values of the external variables that change their values. The 154. article of *Newsweek* from 10th, November 1989 is defined by the following control sequence: $1(Newsweek-101189-154).

The second example is taken from the dissertation of Claudia Kramer-Santel.

1. control sequence: $1(Time-030295-culture-p. 3-headline)

The external variables have the following values:

nr	variable	value
1	medium	Time
2	date	030295
3	column	politics
4	page	p. 3
5	specialities	head line

The 4th external variable is the page number. For better readability no pure numerical solution was chosen. This might cause problems during the statistical analysis, but it has the advantage that conordances and cross references are much easier to read. If a statistical analysis is planned, one has to have in mind that statistical software does have limitations in processing non-numerial data, e.g. SPSS only supports 8 characters in some procedures, SimStat supports 10 characters.

3. example: readability analysis

1. control sequence: $1(gazette-1-1)

The external variables have the following values:

nr	variable	value
1	genre of text	newspaper
2	running number	1
3	sentence counter	1

The next control sequence only has to contain the values of the external variables that change their values. The control sequence for the next sentence is $3(2). If the next text unit is the 3rd sentence of the 5th sample out of the genre prose, this is the control sequence: $1(prose-5-3).

The text unit must be the sentence.

4. example: literary research, e.g. style analysis

1. control sequence: $1(Conrad-Nostromo-1-1)

The external variables have the following values:

nr	variable	value
1	author	Conrad
2	book	Nostromo
3	chapter counter	1
4	paragraph counter	1

The next control sequence only has to contain the values of the external variables that change their values. If the next unit is the 23rd paragraph of the 9th chapter of *Lord Jim* from the same author, the control sequence is:
`$2(Lord Jim-9-23)`

5. example: television news

1. control sequence: $1(RTL-150486-1)

The external variables have the following values:

nr	variable	value
1	station	RTL
2	date	150486
3	item number	1

The next control sequence only has to contain the values of the external variables that change their values. If the next item of the same program follows, the control sequence is $3(2). For the 4th item of RTL-news from 14th April, 1986, the control sequence is: `$1(RTL-140486-4)`

Example with two news items (ARD Tagesschau from 14. April 1986):

```
$1(ARD-140486-1) Last weekend 14 people were killed in severe race riots in South
Africa. According to the police in Johannesburg 9 victims were blacks and killed
because they were thought to cooperate with the government. 5 blacks died in
conflicts with the police. $3(2) 46 hindu pilgrims were killed in the north indian
town Hatwar during a panic. While bathing in the holy river Ganges, some people
fell, and a panic arose. The following crowd moved over them. Estimations say that
over 4 million Hindus are in town to wash away their sins by taking a bath in the
Ganges.
```

6. example: personal advertisments

1. control sequence: $1(tip-020595-3-man-woman-self)

The external variables have the following values:

nr	variable	value
1	medium	BosGlobe
2	date	020595
3	running number	3
4	own gender	man
5	search gender	woman
6	type of image	self

The next control sequence only has to contain the values of the external variables that change their values. If the next text unit contains information what pecularities the woman shall have, the control sequence is: $6(partner). If a woman looks for another woman and describes the type of relationship in the next unit, the control sequence is: $3(4-woman-woman-relation), assuming that the ad is in the same medium on the same day. More examples are in the file `contakt.txt`.

```
$1(160188-BosGlobe-1-man-woman-self) Young man with a good job wants to
meet a $6(partner) woman between 30-40 years, also with children $6(other)
from the Boston-Amherst area $6(relation) to build up a nice friendship.
$1(160188-BosGlobe-2-man-woman-partner) Which young girl (up to 23 years)
$6(relation) is interested in conversation and spending days off with
$6(self) sensible nd honest academic? $6(other) answers with photos please
$1(160188-BosGlobe-5-man-woman-self) Young man, 35 years, 176 cm tall, slim,
with car, good income, looks for a $6(partner) lovely and big busted woman for a
$6(relation) common future.
```

3.4.4 Regulations for using column format

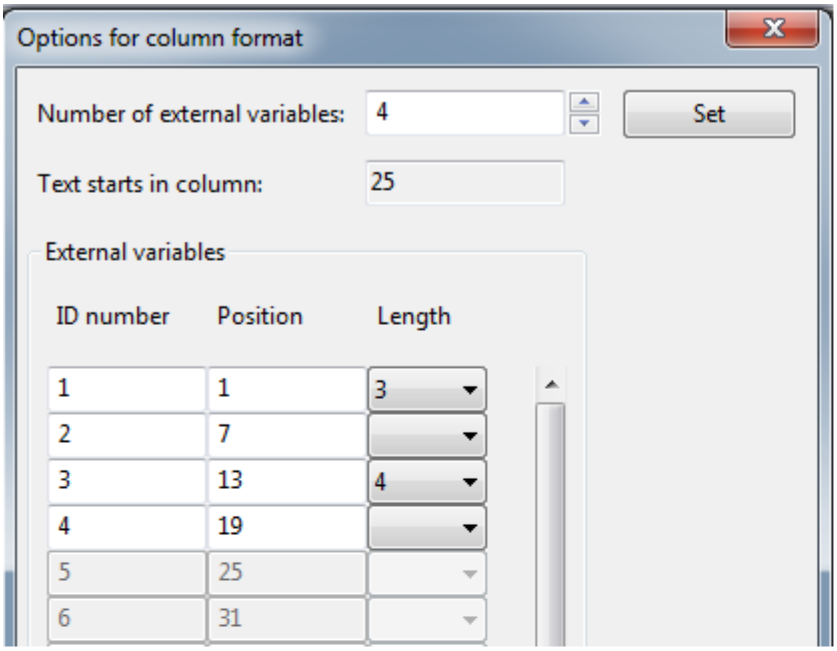
The column format is useful if your texts are already stored in a file, for example in a data base or in a statistical program like SAS, SPSS, or SimStat. These programs allow that the texts are written to a file in a format that **T**ext**Q**uest can read easily. In general each text unit is written to one line that consists of two parts:

- the external variables at the beginning of each line, followed by the text of the text unit. Each line has the same structure, so that each external variable occupies the same columns in every line. For each external variable the column where it starts and its length in characters must be specified. The minimum length is 1 character, the maximum length is 10 characters. The external variables may overlap.
- The text must start in the same column each line, no external variables may follow the text.

Example for a raw text using column format with one external variable, taken from example file `sp-fixed.txt`:

```
01 He made a sliding, and as he started too late, he hit me against the
01 ankle. I had an enormous pain.
02 By accident I smashed the ball into the audience. Fortunately
02 nobody was hurt.
03 I like boxing, you learn how strong your body is.
04 Playing cards is not dangerous, nobody can hit you with them like with a
04 ball.
05 It is the sensation, full speed riding and not collapsing.
06 In case your material is good, nothing bad will happen. Of course
06 you should not take too much risks.
07 The only problem is that you never know what your opponent does.
08 I'm always very cautious. I would feel very bad in case I would
08 kick someone into hospital.
```

The external variable is in column 1-2, the text starts in column 4. If the numbers in the beginning of the line are the same, it belongs to the text unit the line before. If the grid occurs in the second menu, fill in 1 for the numbers of external variables, then click set.



The *set* button is used after you specified the number of external variables, the grid will then be extended to the number you specified. ID (identifier) means the running number of the external variable, the numbering is automatic. You just fill in the start of each ID and its length, the end columns are computed and cannot be changed.

The following regulations have to be considered:

1. The line length of the raw text may not exceed 512 characters.
2. Each new text unit must begin on a new line.
3. Text units may consist of several lines, the external variables must be identical if this is the case.

The *set* button is used after you specified the number of external variables, the grid will then be extended to the number you specified. ID (identifier) means the number of the external variable, the numbering is automatic. You just fill in the start of each external variable and its length, the end columns are computed and cannot be entered.

3.4.5 Regulations for using line format

The line format is useful for literary research just using a line number (1. external variable). Each line is a text unit, the line counter is incremented by each new line symbol (CR). One line may have up to 32500 characters. The line format allows the analysis of texts without inserting control sequences. Sample files are `bush.txt, gore.txt` and `mccain.txt`.

3.4.6 Regulations for using paragraph format

Using paragraph format means that each paragraph is a text unit. Paragraphs are separated by **two** end-of-line characters (CR/LF CR/LF). Only one external variable is supported, the paragraph counter. Sample files are `bush.txt, gore.txt` and `mccain.txt`.

3.4.7 Regulations for using page format

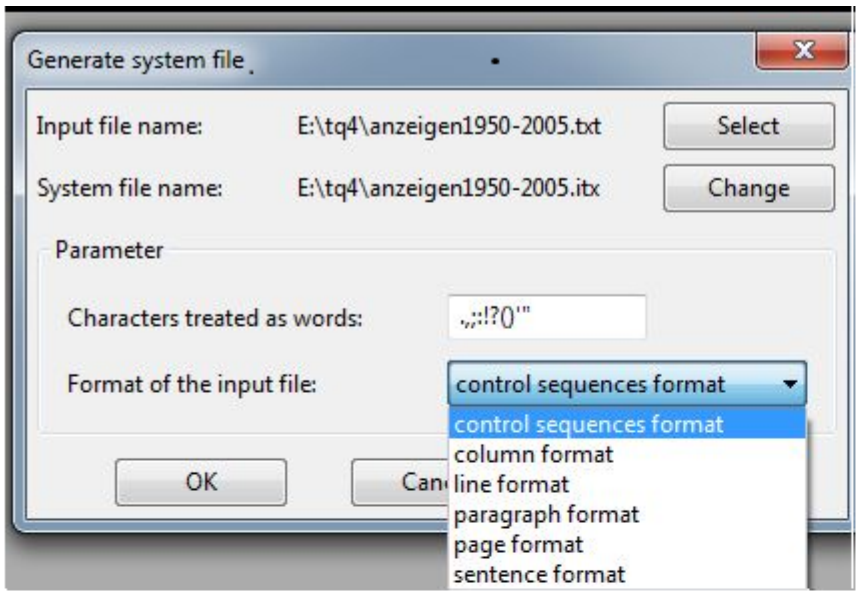
Using page format each line is a text unit, the 1. external variable is the line counter like the line format. After x lines – this value is to be specified by the user – the 2. external variable is incremented by one. Sample files are `bush.txt, gore.txt` and `mccain.txt`.

3.4.8 Regulations for sentence format

Using sentence format, a (grammatical) sentence is a text unit, there is one external variable, the sentence counter. The raw text is splitted into sentences, using delimiters (.?!). Decimal points and abbrevations are recognised, but you still have to check this process. The sentence format is required for readability analyses. The file `ab.def` contains a list of common abbrevations that are recognised.

3.4.9 Generate system file menu

The following picture shows the parameters:



name of raw text file: the name the file that the raw text has. You can accept the file whose name is displayed, or you press the [select] button of the open file dialog.

name of system file: you can accept the generated file name or press the [change] button to open the file dialog.

format of the input file: the formats supported are control sequence format, column format, line format, paragraph format, page format, and sentence format.

characters treated as words: Up to 30 different characters can be treated as a word. These characters must be entered one after the other without blanks. Every character should be entered only once. All these characters are separated from the words, that means a blank is inserted before the character if the character is at the end of the word; if it is at the beginning of a word, a blank is inserted after the character.

3.4.10 Information messages

```
TextQuest (tm) Text Analysis Software 27.02.2013 14:09
         program: ISYS
     application: generate system file
input file       E:\tq4\text\bush.txt
output file      E:\tq4\text\bush.itx
error file        ISYS.ERR
The following characters were separated: .,;:!?()'"
- I 01:       565 lines read
- I 02:         1 errors in the control sequences
- I 03:         6 invalid characters in the text
- I 20:        76 words in a text unit
- I 21:      4726 strings processed
- I 22:     24503 characters processed
- I 23:           commander-in-chief 18 characters in longest string in line 362
- I 24:       289 text units written
- I 25:    16,353 strings/text unit
- I 26:       349 characters in longest text unit 57
- I 27:        14 characters separated before
- I 28:       565 characters separated after
ISYS start: 14:10:03
ISYS end:   14:10:13
ISYS needed 10 seconds
```

Explanation of the information messages:

I 01: number of the read lines
I 02: number of data errors
I 03: number of control sequences
I 04: number of empty lines
I 05: number of lines with comments
I 20: number of strings in longest text unit
I 21: sum of strings
I 22: sum word of bytes
I 23: longest string in the read file and its length in characters and position (line number) in the text
I 24: number of text units written to the output file
I 25: average number of strings in a text unit
I 26: number of bytes in the longest text unit and its number
I 27: number of bytes separated at the beginning of a string
I 28: number of bytes separated at the end of a string

3.4.11 Printed result of a system file with external variables (sport.txt)

01	25	He made a sliding , and as he started too late , he hit me against the ankle . I had an enormous pain .
02	15	By accident I smashed the ball into the audience . Fortunately nobody was hurt .
03	12	I like boxing , you learn how strong your body is .
04	17	Playing cards is not dangerous , nobody can hit you with them like with a ball .
05	12	It is the sensation , full speed riding and not collapsing .
06	22	In case your material is good , nothing bad will happen . Of course you should not take too much risks .
07	13	The only problem is that you never know what your opponent does .
08	21	I ' m always very cautious . I would feel very bad in case I would kick someone into hospital .
09	7	You should never loose your mind .
10	25	I ' m taking care of my machine , I inspect it every day , so from this side I never have any problem .
11	5	They were very brute .
12	29	He hit me in an awful way , I became dizzy , and had to consult a physician . In this way it is no fighting any more .
13	16	He hooked Johnny in such a way that he fell , and broke his leg .
14	14	When you are running there quite alone , you sometimes feel very lonesome .
15	14	You train every day , there is no time left for other activities .
16	11	They tried to make us afraid by yelling very hard .
17	26	He fell , and said that I had hurt him and he had a lot of pain . I ' m convinced he is lying .
18	15	I played far too long , therefore a muscle in my knee got sprained .
19	12	When I went down ski-ing I fell and broke my leg .
20	19	It was wet , therefore I fell and broke my arm . I should have been more cautious .
21	26	He is an awful person , in the boxing match last week he beated his opponent on the head so that he got brain damage .
22	19	I like a rough play , but it should be fair . If so , nobody gets injured .
23	13	Rugby players learn how to fall , so they have hardly problems .
24	12	I never go sailing when there is a lot of wind .
25	12	During the training the gymnastics player fell out of the rings .
26	18	At the end of a football match I always have so many blue spots on my legs .
27	17	The opponents are always intimidating us . They really try to make us afraid of them .
28	13	I always train alone , so you miss talking to other people .
29	12	I like swimming , it is very good for your condition .
30	20	At the end of a match I am always out of breath . It gives you a good feeling .
31	22	This boy could not stand that he was loosing , so he started playing in a way that is not fair .
32	17	I don ' t understand why people like wrestling , those people almost kill each other .
33	17	She was loosing the match , she became very angry and started yelling at the referee .
34	22	He pushed his shoulder against the other cyclist , so this person fell badly and had a concussion of the brain .
35	18	Normally speaking , she is very friendly , but in the field she always is a fury .
36	24	If I can prevent a goal made by the others by tackling some one , I won ' t do it . Sorry .
37	22	Motor-riding is not dangerous , as long as you have good quality materials , and don ' t take irresponsible risks .
38	31	In a hurdle-race you have to stay in the middle of your track , if you do so you can ' t hurt anybody , and nobody can hurt you .
39	20	After a quarter of an hour I got a ball in my stomach , which causes an enormous pain .
40	16	It was wet , therefore I slipped on the course , and broke my arm .

A system file **should not be changed** with an editor, use the raw text instead and rebuilt the system file.

4. Definition of samples

Many **T**ext**Q**uest-programs can work with the whole text or also with parts of it – samples. This process of selecting text units from a file is also called filtering. Sampling or filtering works with external variables, the values of these are used.

At first you have to define the sample, the definitions are stored in a file. The definition consists of values for text units that are to be included if sampling is enabled. Each definition is written in a separate line.

For each external variable up to 10 rules can be chosen, these are connected with a logical **or**, whereas within different external variables the combination is a logical **and**. An example shows what is meant: assume you want to select the newspapers *The Times*, *Mirror*, and *Daily Telegraph* and define these as a sample. During processing all text units are selected where the external variable *medium* has the values of the three papers (logical **or**). If you specify a date or a range of dates, only the text units out of the three papers are selected that are within the date ranges (logical **and**). The following examples show how to define samples. The definitions are written to a file and can be used in the following analyses:

word lists
word sequences
word permutations
cross references
concordances
search units in text
content analysis
readability analysis

The text unit consists of the text and the external variable that represent external variables of the text. These have to be defined by the user. The external variables can be used to draw samples or to process the data in multiple steps. Details are described in chapter 3.4 on page 24.

Each definition of a sample requires a file with the extension `*.sam`, this file can be generated with an editor.

Each definition consists of one line that is structured as follows:

- number of the external variable: the number of the external variable is required in columns 1-2 right justified. If you want to define the 5th external variable, you enter the value 5. Up to 10 limitations are possible for each external variable.
- running number: for each external variable up to 10 limitations are possible. These must be numbered consecutively without gaps and must start with 1. This value is required in columns 4-5 right justified.
- minimum value: the smallest value that an external variable includes. This value can be made up to 10 characters and is written left justified in columns 7-16. This value is a character string, also numbers are treated as such.
- maximum value: the maximum value that an external variable includes. This value can be made up to 10 characters and is written left justified in columns 18-25. This value is a character string, also numbers are treated as such.

The following examples explain the definition of the samples.

1. example: coding of open ended questions, external variables are the date, the number of the person and the number of the question. Only the questions 1, 3, and 5 are to be selected.

```
03 01 1         1
03 02 3         3
03 03 5         5
```

2. example: coding of open ended questions, external variables are the date, number of person and number of question. The first three questions of the first 100 persons are to be selected.

```
02 01 1         100
03 01 1         3
```

3. example: personal advertisments, external variable are medium, date, running number, own gender, searched gender, and type of image. All partner images of women of the *Zeit* looking for men are to be selected.

```
01 01 Zeit      Zeit
04 01 Frau      Frau
05 01 Mann      Mann
06 01 Fremd     Fremd
```

5. The use of search patterns

Search patterns define a category system. They are organised in a file of search patterns together with codes and parameters. In a content analysis the search patterns are searched for within every text unit. If a search pattern is found, the code that belongs to it is written to the output file(s). The parameters are to be specified in the parameter field and control the features for the validity of the coding (files for uncoded, coded, and negated text units).

Search patterns are used for the building of a category system. These can be constructed using the category manager described in the chapter on content analysis.

There are two kinds of search patterns:

1. a string or any word of it, also parts of words and word sequences
2. word co-occurences

Search patterns can be words or parts of it, but also single letters or syllables. Every search pattern starts with a colon (') in column 7 and ends with a colon. Both colons must exist. Instead of a colon any other character that does not occur in the search pattern may be used as a delimiter. The columns 1-3 can be used for generating a concordance and must be used for a three digit code for a content analysis. Columns 4-6 are called the parameter field where parameters can be specified. These control the output of rapport files for ambiguous, uncoded and/or negated text units.

In all **T**ext**Q**uest-versions the number of search patterns that can be processed in one analysis is limited by the available memory (RAM), only the number of word co-occurences is limited to 2000.

5.1 Specifications in the parameter field

The parameter field can be used to control the treatment of each search pattern. The following parameters are possible:

C coding control All text units that contain the search patterns are written to the file of coded text units. If interactive coding is enabled, the text unit, the search pattern, the category number and the corresponding label are displayed. The coding decision (yes or no) and the code can be specified.

U Uppercase All characters of the search pattern are translated into uppercase, so that lower case and upper case are treated as the same. This is useful with words that are capitalised because they are at the beginning of a sentence.

N negation The search pattern is checked for negation. If an odd number of indicators before and after the search pattern occurs, the search pattern is not coded. The search pattern is coded when an even number (e.g. double negation – litotes) of indicators occurs. The number of words before and after the search pattern where indicators are searched can be specified (default: 2), also the list of indicators – separately for before (NEG-PRE.DEF) and after (NEG-POST.DEF) the search pattern. These files can be edited and adapted to other languages.

5.2 Strings

Strings as search patterns are a part of a text unit. It doesn't matter whether a string is just a letter or a sequence of words. Strings may also be any part of a word. The maximum length is 500 characters. Within a string the ? can be used as a wildcard character, the use is the same as in file names. A ? substitutes exactly one character. The asterisk * is the wildcard character for any number of characters before and after it, but is limited to one single word.

A line in the file of search patterns (file extension `*.dic`) is structured as follows:

column	contents
1 - 3	code
4 - 6	parameter field (may be left blank)
7	delimiter (e.g. colon)
8 - 500	search pattern (delimited with delimiter used in column 7)

An example for the definition of strings as a search pattern (with option U enabled for ignoring differences in case):

search pattern	found text
' man '	man (no other words)
' man'	man, mankind, maniac, manner, mangle
'man'	man, mankind, maniac, manner, mangle, woman, superman
' m?n '	man, men, mon
' super*man '	superman, superwoman
' super*m?n '	superman, superwoman, supermen, superwomen

Example for a string as a search pattern:

```
001 U ' president'
002 C ' america'
013 C ' environment'
```

By using blanks one can define whether a string should be treated as a word or as a part of it. So it's possible to define unambiguous words or parts of words as prefixes or suffixes. The examples mentioned above show the use, more examples are in the provided `*.dic` files (e.g. `kontakt.dic`).

5.3 Word co-occurences

Word co-occurences are similar to strings as search patterns. Word roots – these are words or parts of strings – can be defined in such a way that they must occur within a text unit in the order they are specified as a search pattern. The distance between two word roots doesn't matter. The distance between the word roots as well as their order within a text unit can be varied. There are three kinds of word co-occurences that must be marked in the parameter field:

- option D: direct mode. The strings must occur in words that follow each other without any other strings (words, colon etc.) between them within a text unit.
- option F: following mode. The strings must follow each other within a text unit, but the distance between them doesn't matter and is dependent on the definition of a text unit.
- option S: simultaneous mode. The strings must occur within a text unit, order and distance do not matter.

The definition of word co-occurences is done with the (<,>) symbols. Before and after the word root may be characters, but there are non required. < indicates, that characters in front of the word root are allowed, > indicates, that characters after the word root are allowed. Also the wild card symbols

? and * may be used, the same regulations as for strings as search patterns apply. Up to 5000 word root chains can be used in one analysis.

Examples:

word co-occurences	found text
'<intelligent man'	intelligent man, unintelligent man nonintelligent man
'good <man>'	good man, good woman, good manners, good womaniser

Examples for a word co-occurences as a search pattern:

```
004 C '<intelligent man'
005 C 'good <man>'
005 C 'bad guy>'
```

The category manager is a powerful tool if you want to develop and maintain a category system. It shows you all tyes of vocabularies (word list, word sequences and word permutations) that can help you to find adaequate search patterns.

However, you can use the standard category system that come with **T**ext**Q**uest and adapt them for your purposes. Or you can use a simple text editor to create the file of search patterns (`*.dic` file) and the file of the category labels (`*.lab` file). The category manager however knows all the rules, how to set the blanks and the options of the parameter field. So using this tool is much more comfortable than creating the files with an editor.

6. Qualitative analyses of text

The purpose of text analyses in the social science is the collection of information, its ordering and analysis. Parts of the text are marked for further analysis.

In the context of quantitative text analysis techniques this means, that a category system is developed with the help of a vocabulary, e.g. a word list, word sequences, or word permutations. Search patterns – often words or parts of it – are grouped in categories. Each search pattern must be a valid indicator for the category, and each category has a numeric code. All search patterns of a category system are searched in each text unit. If a search pattern is found, the code of the search pattern will be processed and written to an output file that can be analysed with statistical software.

In the context of qualitative data analysis the meaningful parts of a text are marked by codes. These codes beome search patterns, you can compare text segments etc. Statistical analyses are done rather seldom, although it is possible.

A prerequisite of codings are search patterns that are to be found. If **T**ext**Q**uest is used for a qualitative analysis, unique codes for marking parts of the text must be defined that can be used as search patterns. It is important that these search patterns are not ambiguous and that these cannot occur in the text to be analysed.

An example: The number sign # can be used as a unique code that can be followed by a part of text of undefined length. The file `qual.lab` (identical with `contakt.lab`) contains the labels of the codes of the category system for the quantitative analysis of the text. These can be converted into a form suitable for qualitative analysis like this (translated):

Code	category	qualitative code
1	cultural background	#culture
2	geographical mobility	#geoMobil
3	local boundedness	#local
4	open minded	#openmind
5	unconventionality	#unconv
6	academic profession	#academic
7	high economic status	#highstatus
8	low economic status	#lowstatus
10	politicially conservative	#conservative
11	politicially liberal	#liberal

Of course this example can be altered, but it is important that the # is followed by unique character combinations. It is not absolutely necessary, that the # is followed by only one word, you can use more. They can be as long as you wish, but inserting long codes into a text takes more time, and it is more likely that orthographical errors occur the longer these codes are.

Text**Q**uest works with search patterns, so the marked parts of the text must be formulated as word co-occurences. The following example shows the technique (see also file `qual.txt`):

```
Young man aged 30, 1,78cm, #single living alone #, wants to meet a
girl with #body attributes a slim figure and long hair # for a long
lasting relationship.
```

In the example two parts of the text are marked, *living alone* for the category *single* and *a slim figure and long hair* for the category *body attributes*. If these parts of the text are to be analysed using a content analysis or a concordance, one can use the following word co-occurences as search patterns (see also file `qual.dic`.):

```
001 f '#single #'
002 f '#body attributes #'
```

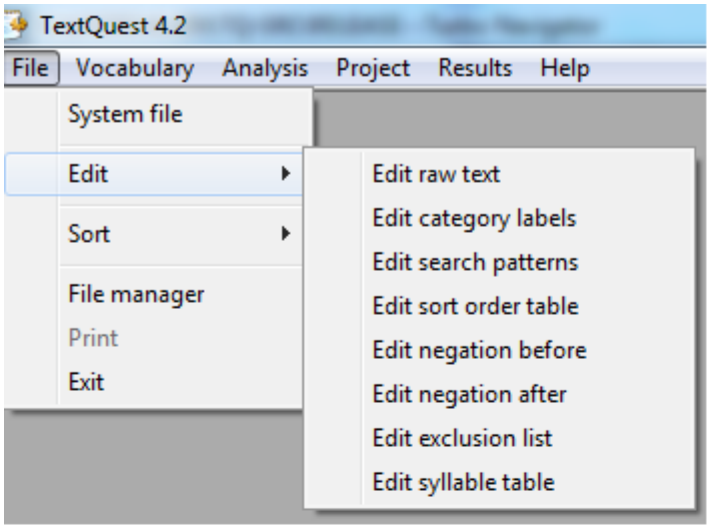
The files `qual.txt` and `qual.dic` contain some parts of the text and search patterns one can use for a qualitative text analysis with **T**ext**Q**uest.

7. The menu: files

7.1 Build system file

Before any analysis can be performed, the building of a system file is necessary. There are several formats available that allow a direct import into **T**ext**Q**uest. Please note that currently only one input file is possible. If you have more than one input file, you must merge these files into a single one.

7.2 Edit



You can browse through the files, and you can edit each file. If you changed the contents of a file and leave, you will be prompted to either save the file or quit – and leave its contents unchanged. The following files can be edited:

- raw text: The raw text is necessary for the building of the system file and should be stored safely. Depending on the input format, it may contain control sequences that determine the

values of external variables. However, if you change anything within this text, you have to build the system file again.

- category labels: The category labels are only necessary for a content analysis. You can edit an already existing file of category labels or create a new one usning a text editor or the category system editor. This file requires a special formatting described in the chapter on search patterns, details see chapter 5 on page 39.

- search patterns: Search patterns are required for content analyses, for search patterns in text units, and for concordances; also known as keywords-in-of-context (KWIC). Details are explained in chapter 5 on page 39. In short: each search pattern requires a line with a maximum length of 500 characters, the first 3 columns contain the code for the search pattern, column 4-6 is the parameter field where options can be set, and column 7 contains a delimiter for the search pattern that also must end the line. The most common used delimiter is the colon (’). A file of search patterns can also be created using the category system editor.

- sort order table: The sort order table is shown. This is important for the sort order of non-English languages that use umlauts, accents, and/or diactrics. The defaults sort order table `sort.def` is sufficient for most germanic and roman languages like French, Spanish, Dutch, or Italian. The sort order table is used for vocabularies.

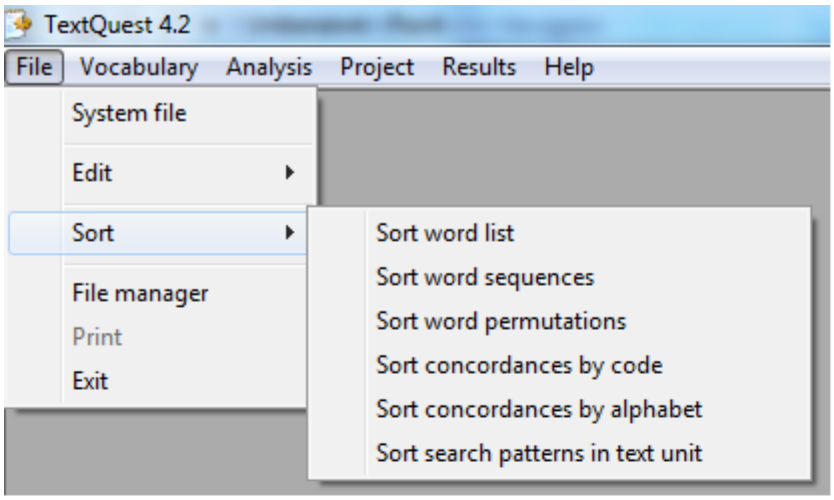
- negation before: This table specifies indicators for negation, a feature in a content analysis, that are detected before a search pattern in a certain distance (see the content analysis chapter for details).

- negation after: This table specifies indicators for negation, a feature in a content analysis, that are detected behind a search pattern in a certain distance (see the content analysis chapter for details).

- exclusion list: Vocabularies often contain words that are not important for further analyses, mostly pronouns, articles, numbers, etc. These words can be excluded from a vocabulary. Exclusion lists are language dependent and included for English, German, and French.

- syllable table: The syllable table is used for readability analyses. It contains the character combinations that are counted as vowels and/or diphtongs, and they are language specific. There are syllable tables included for English and German, and the precision is better 95 % in correct counting of syllables.

7.3 Sort

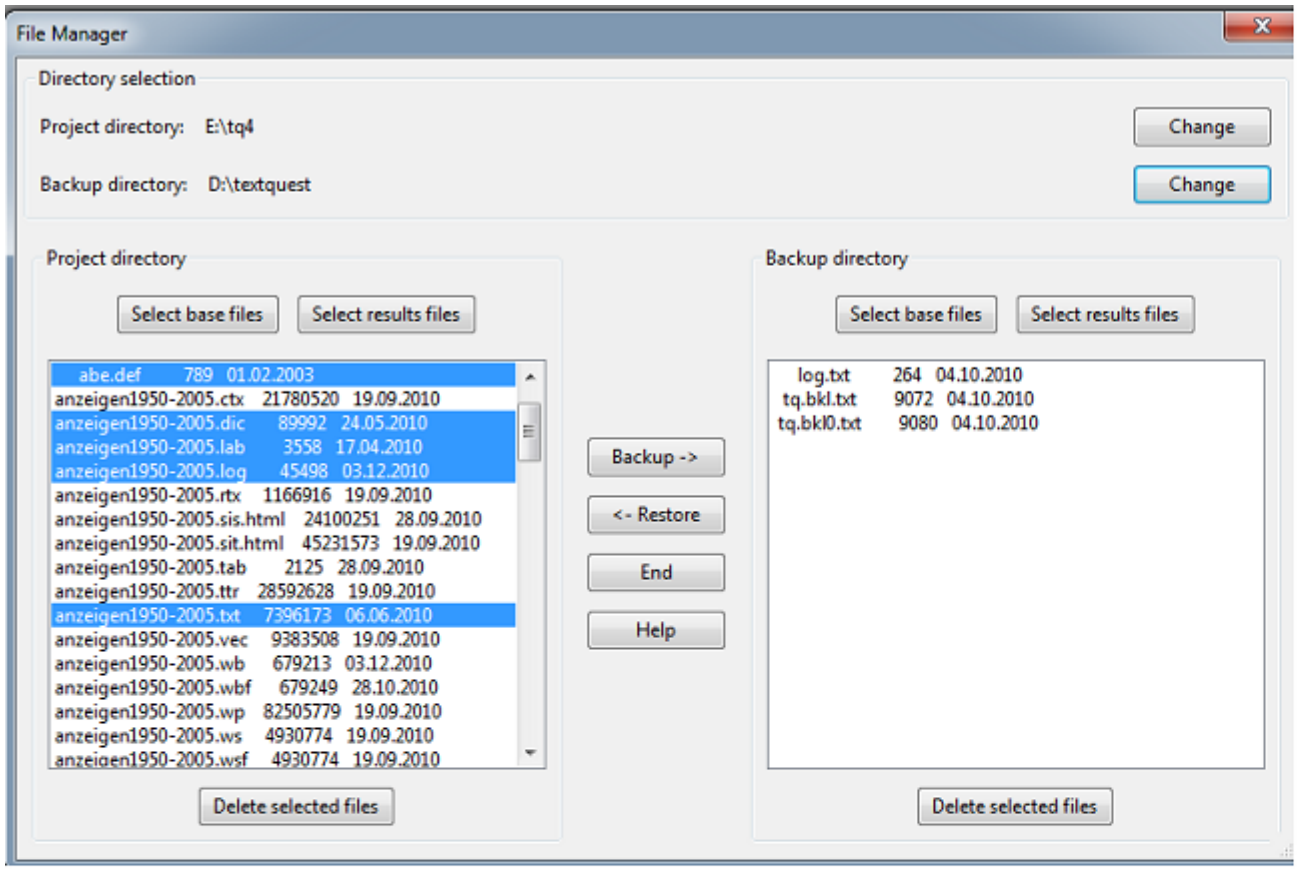


Some of the output files can be sorted to different sorting criteria and modes. The orignal files are always kept and not overwritten, you can edit the new sorted files in the Results menu.

- word list: The word list is originally sorted by alphabet ascending (from a to z). This option sorts the word list by frequency ascending, so that the most frequent words occur at the beginning of the file. The original file is kept
- word sequence: The word sequences are originally sorted by alphabet ascending (from a to z). This option sorts the word sequences by frequency ascending, so that the most frequent word sequences occur at the beginning of the file.
- word permutations: The word permutations are originally sorted by alphabet ascending (from a to z). This option sorts the word permutations by frequency ascending, so that the most frequent word permutations occur at the beginning of the file.
- concordance by code: The concordances are not sorted and listed as they occur in the text. Sorting by code means, that the concordances are sorted by the categories and their codes.
- concordance by alphabet; The concordances are not sorted and listed as they occur in the text. Sorting by alphabet code means, that the concordances are sorted by their search patterns in ascending alphabetical order.
- search patterns in tet unit: The search patterns in text unit are not sorted and listed as they occur in the text. Sorting means, that the search patterns in text units are sorted in ascending alphabetical order.

7.4 File manager

The file manager allows you to backup, restore and delete the files of your projects. One can select groups of files, e.g. the files that are absolutely necessary for a project, or the results files only.



There are two windows that show directories/folder – these can be changed. The left directory is always the project directory where you can see all files of your project and select those that you want to copy. **T**ext**Q**uest knows which are the base files and which are the results files of a project.

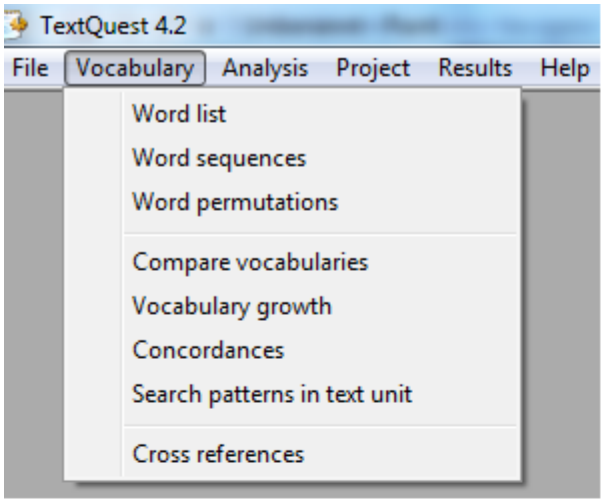
7.5 Print

The content of the active window (it is not grey) can be printed.

7.6 Exit

You leave **T**ext**Q**uest here. If you haven't done it before, backup your files before you leave. If you don't do it, no information gets lots because your work was stored in output or journal files.

8. The vocabularies



Analysing texts means that ones analyses their vocabulary. The term vocabulary is used for word lists, word sequences, word permutations, and cross references. These can be generated, reduced, and compared with each other. Vocabularies are useful for checking the spelling of the text, to describe the text, and as a basis for constructing a content analytical category system. Sometimes they are huge in size, and therefore they should be reduced.

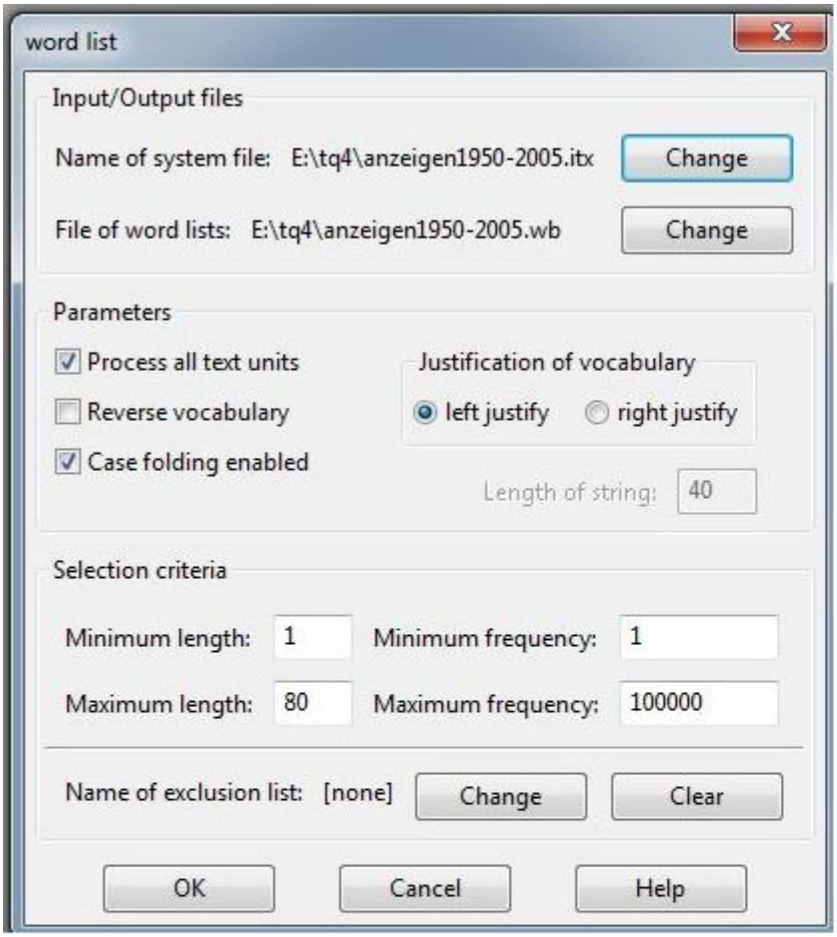
The following criteria can be used to exclude strings from processing:

- external variables in form of a sample, see chapter 4 on page 37.
- length, measured in number of characters
- frequency, both absolute values or in per cent (%) or per mille (%%), e.g. 3,4 %%. All values are inclusive. If e.g. the minimum length is 3 and the maximum length is 10, then all strings with at least 3 and at most 10 characters are processed.
- occurrence in an exclusion list (e.g. `english.exc`), these entries are not written to the vocabulary. The entries in this file need not to be sorted by alphabet. Processing takes a lot more time then without an exclusion list.

All criteria can be combined. Length and frequency are specified by minimum and maximum values (inclusive values). For each vocabulary one can process the whole text or a pre-defined sample, ignore differences due to case folding, control the format (normal or reverse), and – in case of reversed vocabularies – the justification (left- or right justified).

8.1 Word list

A word list is a table of all strings that occur within the system file (mostly words) and their frequency. It is sorted ascending by alphabet. It is used both to spot input errors and as a working help for the building of categories in a content analysis. Working with the sort order table `sort.def` and ignoring difference in upper-/lowercase (case folding) are possible. Also strings can be exluded due to their length, their frequency and/or their occurence in an exclusion list (STOP-word file).



A word list in normal form contains all strings and their frequency of a text that must have the form of an **T**ext**Q**uest system file. The following parameters are available:

name of system file: you can accept the generated file name or press the change button to open

the file dialog.

file of word list: you can accept the generated file name or press the change button to open the file dialog.

process all text units If you check this box, the complete text will be processed, otherwise the defined sample will be processed. Details are described in chapter 4 on page 37.

case folding enabled: Letters can be treated as the same, if they are different only in their case (lower or upper case).

format of vocabulary: normal form or reverse form. The reverse form is used when word endings are to be compared. Reverse means, that the first letter becomes the last, the second letter becomes the one before the last one. Example: girl becomes lrig, man becomes nam, woman becomes namow.

justification of vocabulary: left justified or right justified. If word endings are to be compared, right justified formats the words in such a way that the word endings are placed exactly under each other.

minimum length: the minimum number of characters a string must have to be included in the vocabulary.

maximum length: the maximum number of characters a stringmay have to be included in the vocabulary.

minimum frequency: the minimum number of occurences a string must have to be included in the vocabulary.

maximum frequency: the maximum number of characters a string may have to be included in the vocabulary.

name of exclusion list: If you enter a valid file name, all strings that are in the exclusion list will not be processed.

8.1.1 Information messages

```
TextQuest (tm) Text Analysis Software 27.02.2013 14:10
         program: WORDBOOK
     application: word list
input file        E:\tq4\text\bush.itx
output file       E:\tq4\text\bush.wb
options: upper-/lower case ignored
statistics:
strings (token) read:

- I 01:       289 text units
- I 03:      4083 words
- I 04:        11 numbers
- I 05:       629 other
- I 06:      4723 total
- I 07:    14,128 words/text unit
- I 08:     0,038 numbers/text unit
- I 09:     2,176 other/text unit
- I 10:    16,343 total/text unit
            types     token   TTR   type of string
strings written:

- I 21:      1278      4083   0,313 words
- I 22:        11        11   1,000 numbers
- I 23:        12       629   0,019 other
- I 24:      1301      4723   0,275 total
WORDBOOK start:  14:10:55
WORDBOOK end:    14:10:56
WORDBOOK needed 1 seconds
```

The following table shows the frequencies of character strings (types) in the text. In the first line you see 859 different types that only occur once within a text, this is a percentage of 66,026 of all types. Types that only occur once in a text are called hapax legomena, or short hapaxes. In the second line you see that 187 types occur twice within a text, the third line shows that there are 88 types that occur three times within a text and so on.

```
Frequency statistics of vocabulary
occurence frequency percentage
      1       859  66,026
      2       187  14,374
      3        88   6,764
      4        46   3,536
      5        24   1,845
      6        11   0,846
      7         7   0,538
      8         9   0,692
      9         5   0,384
     10         4   0,307
     11         5   0,384
     12         4   0,307
     13         4   0,307
     14         4   0,307
     15         2   0,154
     16         4   0,307
     17         3   0,231
     18         0   0,000
     19         3   0,231
     20         2   0,154
     21         1   0,077
     22         0   0,000
     23         0   0,000
     24         0   0,000
     25         1   0,077
    >25        28   2,152
```

The last line show that there are 28 types in the text that occur at least 26 times.

The following table show the lengths of the types in the text. The first column shows the length of a type in characters. In the first line you see how often types with one character occur in the text, and their percentage of the vocabulary of the text. So there are 12 types with a length of one character, 29 types with a length of 2 characters, and so on in the text.

Length statistics of vocabulary

length	frequency	percentage
1	12	0,922
2	29	2,229
3	66	5,073
4	173	13,297
5	173	13,297
6	195	14,988
7	190	14,604
8	154	11,837
9	120	9,224
10	101	7,763
11	42	3,228
12	22	1,691
13	14	1,076
14	4	0,307
15	1	0,077
>15	0	0,000

8.1.2 Printed results of a word list (normal form)

1	!	1	1965	1	405	1	accelerate
633	”	2	1968	1	46	2	accept
1	#1988	2	1970	2	47	4	acceptable
1	#51	1	1971	3	48	2	Acceptance
1	#90	2	1972	3	5	4	accepted
16	&	14	1973	1	50	1	accepts
7	&T	2	1977	4	500	5	access
391	’	1	1980	1	51	1	accommodate
3	(	4	1982	1	53	1	accomodate
3	)	3	1984	7	6	1	accomplish
999	,	4	1985	3	60	1	accord
19	-	2	1986	1	600	4	According
12	–	1	1987	2	64	5	account
1	—	15	1988	1	654	1	accounting
1	-a	1	1989	2	7	1	accurate
1	-ANY	3	1990	1	700	1	achievements
1	-related	1	1991	1	75	2	acknowledge
8	-rights	1	1992	2	8	1	acknowledged
1	-traditional	7	2	1	80	1	acknowledges
999	.	1	2-to-1	1	84	3	across
20	000	1	2-year-old	1	86	11	Act
1	000-word	6	20	1	89-year-old	1	acted
1	040	2	200	3	9	1	acting
11	1	1	202	4	90	18	Action
6	10	1	20th	1	91	4	actions
1	10-day	1	21	1	92	2	active
2	100	1	21-member	1	93	2	activist
1	101	2	212-square-mile	1	98	5	activists
1	106	2	22	1	99	7	activities
1	10th	1	23	25	:	2	activity
2	12	1	23-officer	13	;	1	actual
1	12-page	1	238	24	?	8	actually
2	13	3	24	550	A	7	Ada
2	130	2	25	1	A-Word	1	add
1	136	1	26	1	abandoned	9	added
1	14	1	28-year-old	2	ability	1	addicted
1	14th	1	289	4	able	3	adding
1	15	2	29	3	abnormality	1	addition
1	15-minute	1	2nd	2	abort	1	additional
1	150	2	3	316	Abortion	1	Additionally
2	16	5	30	2	abortion-rights	3	address
1	16-year-old	2	30-year-old	1	abortionist	1	addtional
1	17	2	300	1	abortionists	1	adhere
1	174	2	32	81	abortions	8	Administration
1	18	1	34	60	about	2	administrator
1	1860	1	35-year-old	1	abrogating	1	adolescents
4	19	1	36	1	absolute	2	adopted
1	191	1	375	1	absorbing	3	adoption
1	1923	4	4	1	abstinence	1	advanced
1	1930	2	4-to-1	2	abstract	1	advantage
1	1956	1	40	2	abuse	1	adverse

8.1.3 Printed result of a reverse word list

1	!	2	21	1	8891#	2	anozirA
633	”	2	22	1	89	2	arabraB
16	&	2	23	3	9	1	ardnaS
391	’	2	2791	4	91	1	are
3	(	4	2891	2	92	3	arraM
3	)	1	29	1	982	1	artxe
999	,	1	2991	1	9891	2	atinA
19	-	2	3	1	99	1	atnaltA
12	–	2	31	25	:	1	atnaS
1	—	1	32	13	;	4	atokaD
1	-dnoces	1	3291	24	?	3	atosenniM
26	-itna	1	35	550	A	1	aV
999	.	14	3791	1	A&Q	1	avlE
1	/reywaS	1	39	1	a-	1	avonalliV
20	000	4	4	1	abuC	1	ayleS
2	001	1	41	1	abuL	13	B
2	002	3	42	1	acahtI	1	beF
2	003	1	43	10	aciremA	3	bmow
4	005	1	456	1	acitU	2	boj
1	006	2	46	7	adA	2	bruC
1	007	1	471	1	ademalA	4	buH
6	01	1	48	1	adicarG	4	C
6	02	3	4891	1	adirolF	3	ciffart
5	03	3	5	1	aedi	3	cificaP
2	031	1	504	3	aera	2	cificeps
1	0391	1	51	2	aibmuloC	1	cifitneics
1	04	2	52	3	AICIRTAP	2	cigam
1	040	1	5691	2	aidem	42	cilbup
1	05	1	57	2	aihpledalihP	56	cilohtaC
1	051	1	573	1	ainamoR	1	cimonoce
3	06	4	5891	2	ainavlysnneP	1	cinapsiH
1	0681	7	6	1	ainigriV	1	cinapsiH-itna
4	0691	1	601	1	ainogataP	1	cinecs
2	0791	2	61	5	ainrofilaC	1	cinhte
1	08	1	62	1	alF	24	cinilC
1	0891	1	63	1	alkO	2	ciporhtnalihp
4	09	1	631	1	allahlaV	2	cisab
1	09#	1	64	1	allerbmu	9	citarcomeD
3	0991	1	6591	1	alleroM	1	citcaT
11	1	1	68	1	alleurroT	1	citehtapmys
1	1-ot-2	2	6891	1	allycS	1	citehtnys
2	1-ot-4	2	7	7	aloirrA	1	citeneg
1	101	1	71	1	alumrof	1	citirc
1	12	2	74	1	amabalA	1	citnaltA
1	15	2	7791	2	amgitS	1	citpes
1	15#	1	7891	1	amikaY	1	citsilarulp
1	1791	2	8	1	amohalkO	1	citsimitpo
1	19	1	81	1	anagA	1	citsiurtla
1	191	1	832	1	anaidnI	2	civiC
1	1991	3	84	1	anairaM	2	cnI

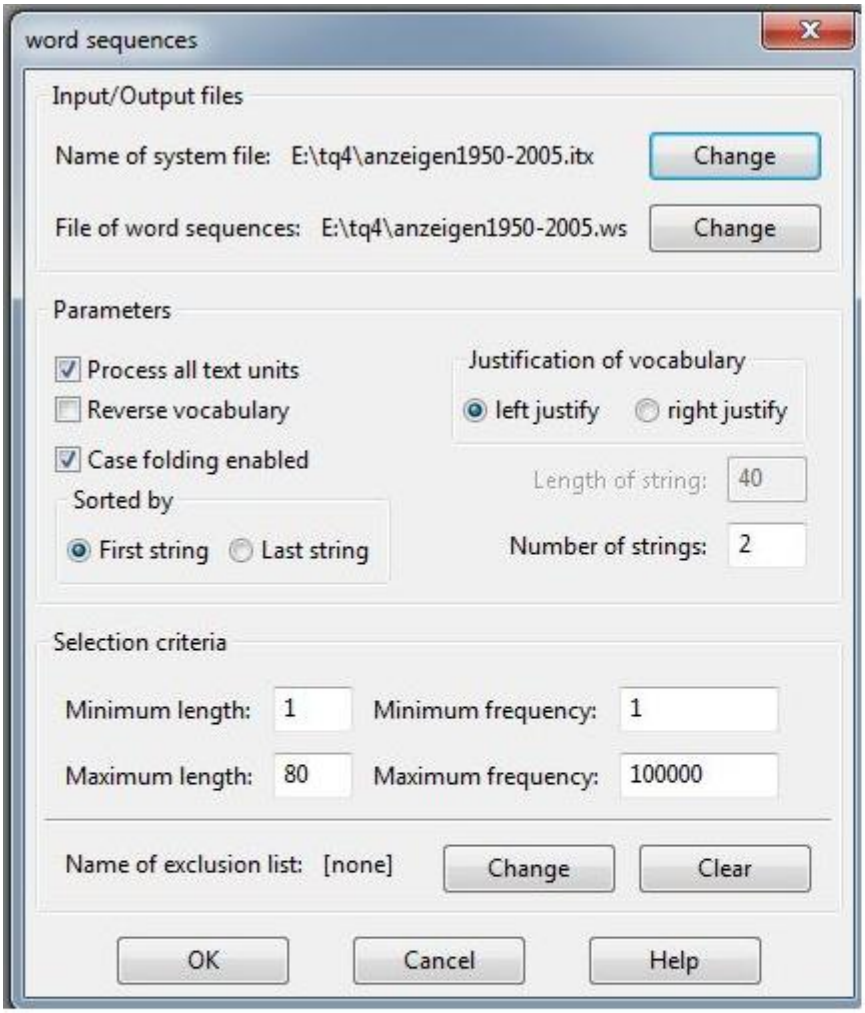
8.2 Word sequences

An analysis technique that exceeds the limits of single words is the generation of word sequences. These are parts of a text unit that consist of x words, the value of x is to be defined. If it is 1, a word list is generated, if it is greater, all word sequences up to this length will be generated. An example: if x is 4, all single words and all word sequences with 2, 3 and 4 words are generated. Please not that all punctation marks are separated from the words before when the system file is generated as a default.

If a text unit is [This is a test of a computer .] and word sequences with 2 or 3 words are to be generated, the following word sequences are generated:

2 words	3 words
[This] a test of a computer .	[This] a test of a computer .
[This is] a test of a computer .	[This is] a test of a computer .
This [is] test of a computer .	[This is a] test of a computer .
This [is a] test of a computer .	This [is] a test of a computer .
This is [a] of a computer .	This [is a] test of a computer .
This is [a test] of a computer .	This [is a test] test of a computer .
This is a [test] a computer .	This is [a] test of a computer .
This is a [test of] a computer .	This is [a test] of a computer .
This is a test [of] computer .	This is [a test of] a computer .
This is a test [of a] computer .	This is a [test] a computer .
This is a test of [a] .	This is a [test of] a computer .
This is a test of [a computer] .	This is a [test of a] computer .
This is a test of a [computer] .	This is a test [of] a computer .
This is a test of a [computer .]	This is a test [of a] computer .
	This is a test [of a computer] .
	This is a test of [a] computer .
	This is a test of [a computer] .
	This is a test of [a computer .]
	This is a test of a [computer] .
	This is a test of a [computer .]

The following picture shows the parameters:



name of system file: you can accept the generated file name or click the change button to open the file dialog.

file name of word sequences: you can accept the generated file name or click the change button to open the file dialog.

process all text units If you check the box, the complete text will be processed, otherwise the defined sample will be processed. Details are described in chapter 4 on page 37.

case folding enabled Letters can be treated as the same, if they are different only in their case (lower or upper case).

format of vocabulary: normal form or reverse form.

justification of vocabulary: left justified or right justified.

sort criterion for word sequences: the word sequences can be sorted alphabetically by the first string of the sequences or by the last string.

number of strings: This value defines, how many words form a word sequence. The default value is 2, the highest is the number of words in the shortest text unit. For example, if the shortest text unit consists of 9 words, the highest value that makes sense is 9.

minimum length: the minimum number of characters a string must have to be included in the vocabulary.

maximum length: the maximum number of characters a string may have to be included in the vocabulary.

minimum frequency: the minimum number of occurences a string must have to be included in the vocabulary.

maximum frequency: the maximum number of characters a string may have to be included in the vocabulary.

name of exclusion list: if you you can accept the generated file name or click the change button to open the file dialog. If you enter a valid file name, all strings that are in the exclusion list will not be processed. You can edit the exclusion list or use the ones that are provided.

8.2.1 Information messages

```
TextQuest (tm) Text Analysis Software 27.02.2013 14:10
         program: WORDBOOK
     application: word sequences
input file        E:\tq4\text\bush.itx
output file       E:\tq4\text\bush.ws
options: upper-/lower case ignored sorted by first string
range: 5 words

statistics:
strings (token) read:

- I 01:       289 text units
- I 03:     18749 words
- I 04:        49 numbers
- I 05:      1944 other
- I 06:     20742 total
- I 07:    64,875 words/text unit
- I 08:     0,170 numbers/text unit
- I 09:     6,727 other/text unit
- I 10:    71,772 total/text unit
            types     token   TTR   type of string
strings written:

- I 21:     14955     18749   0,798 words
- I 22:        49        49   1,000 numbers
- I 23:      1112      1944   0,572 other
- I 24:     16116     20742   0,777 total
WORDBOOK start:  14:11:07
WORDBOOK end:    14:11:08
WORDBOOK needed 1 seconds
```

Word sequences are parts of texts that consist of several words, this number is variable. One can exclude those word sequences that contain a word that also occurs in an exclusion list. Word sequences can find the number of phrases, and it also can be used to define search patterns for a content analysis, or for disambiguation.

The following tables show the frequencies of the frequencies and the length of word sequences, as explained in the chapter on wordlists.

Frequency statistics of vocabulary

occurence	frequency	percentage
1	15020	93,199
2	644	3,996
3	188	1,167
4	84	0,521
5	41	0,254
6	25	0,155
7	9	0,056
8	14	0,087
9	10	0,062
10	7	0,043
11	5	0,031
12	10	0,062
13	6	0,037
14	5	0,031
15	2	0,012
16	4	0,025
17	4	0,025
18	0	0,000
19	3	0,019
20	3	0,019
21	2	0,012
22	1	0,006
23	0	0,000
24	0	0,000
25	1	0,006
>25	28	0,174

Length statistics of vocabulary

length	frequency	percentage
1	12	0,074
2	29	0,180
3	79	0,490
4	222	1,378
5	271	1,682
6	388	2,408
7	542	3,363
8	608	3,773
9	680	4,219
10	717	4,449
11	747	4,635
12	758	4,703
13	719	4,461
14	687	4,263
15	720	4,468
>15	5409	33,563

8.2.2 Printed results of word sequences

1 Atlantic
1 Atlantic Partnership
1 Atlantic Partnership .
1 attacks
1 attacks civilians
1 attacks civilians -
1 attacks civilians - killing
1 attacks civilians - killing women
1 attaining
1 attaining peace
1 attaining peace on
1 attaining peace on the
1 attaining peace on the Korean
2 attention
1 attention .
1 attention on
1 attention on a
1 attention on a corrupt
1 attention on a corrupt and
1 attraction
1 attraction of
1 attraction of these
1 attraction of these weapons
1 attraction of these weapons for
1 austerity
1 austerity ,
1 austerity , bailing
1 austerity , bailing out
1 austerity , bailing out bankers
1 Australia
1 Australia and
1 Australia and Thailand
1 Australia and Thailand .
1 average
1 average people
1 average people -
1 average people - their
1 average people - their warm
1 avert
1 avert ,
1 avert , the
1 avert , the prosperity
1 avert , the prosperity we
1 away
1 away with
1 away with unwise
1 away with unwise treaties
1 away with unwise treaties .
1 back
1 back from
1 back from a
1 back from a nightmare
1 back from a nightmare world
1 bailing
1 bailing out
1 bailing out bankers
1 bailing out bankers while
1 bailing out bankers while impoverishing
2 balance
1 balance ancient
1 balance ancient ambitions
1 balance ancient ambitions ,
1 balance ancient ambitions , this
1 balance takes
1 balance takes time
1 balance takes time to
1 balance takes time to achieve
1 ballistic
1 ballistic missiles
1 ballistic missiles .
1 ballistic missiles . .
1 ballistic missiles . . .
1 Baltics
1 Baltics ,
1 Baltics , the
1 Baltics , the Caucasus
1 Baltics , the Caucasus and
2 Ban
2 Ban Treaty
1 Ban Treaty does
1 Ban Treaty does nothing
1 Ban Treaty does nothing to
1 Ban Treaty is
1 Ban Treaty is not
1 Ban Treaty is not the
2 bank
1 bank accounts
1 bank accounts of
1 bank accounts of corrupt
1 bank accounts of corrupt officials
1 Bank and
1 Bank and the
1 Bank and the IMF
1 Bank and the IMF .
1 Bank and the IMF .
1 bankers while

8.3 Word permutations

Word permutations are performed for each text unit. They consist of two word sequences: the first word with the second and all other following words, the second word with the third word and all other following words, and so on. Word permutations can be used as a basis for exploring word co-occurences that are in the text, and so support the definition of search patterns as word co-occurences for a content or style analysis.

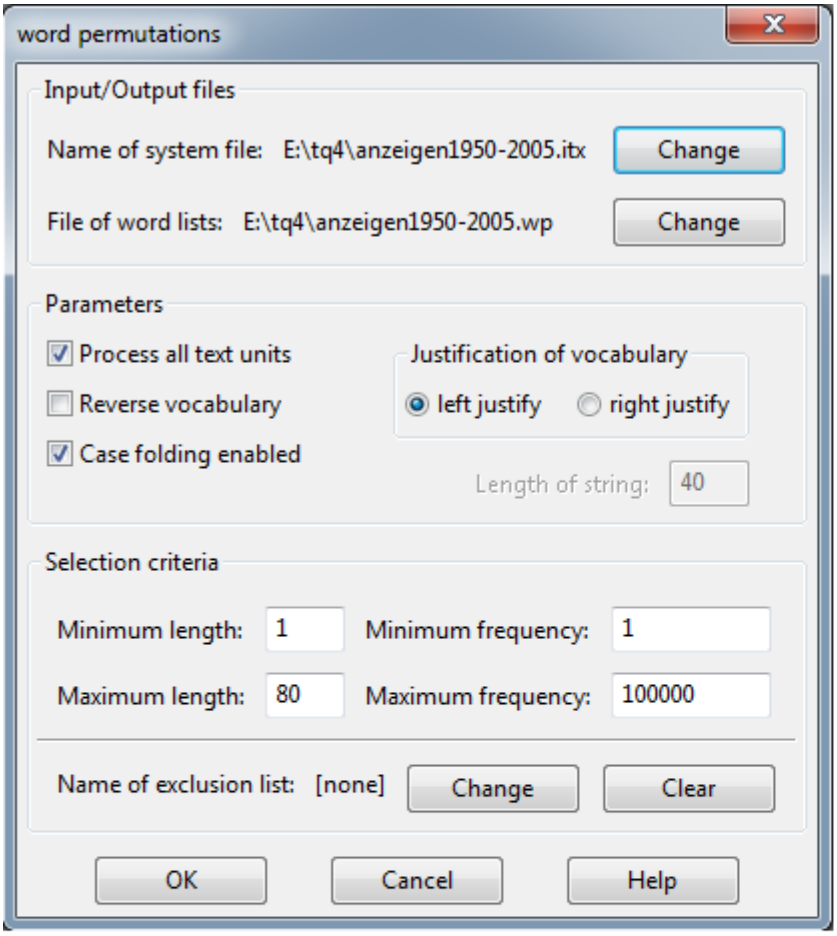
If a text unit is This is a test of a computer, the raw output (unsorted) is the first column of the following table, and the alphabetically sorted list with its frequencies is in the second column. The first column is for the demonstration of the logic of word permutations, it is never generated. Word permutations are counted and stored in a Bayer tree, thus using a lot of RAM and a lot of time. Processing huge amounts of data, you might encounter a hang of the computer, sometimes even for minutes. Please be patient, **T**ext**Q**uest writes a lot of data then and did not crash.

unsorted
This is
This a
This test
This of
This a
This computer
is a
is test
is of
is a
is computer
a test
a of
a a
a computer
test of
test a
test computer
of a
of computer
a computer

	sorted
1	a a
2	a computer
1	a of
1	a test
2	is a
1	is computer
1	is of
1	is test
1	of a
1	of computer
1	test a
1	test computer
1	test of
2	This a
1	This computer
1	This is
1	This of
1	This test

Word permutations need a lot of RAM and computing time, depending of the length of the text units and the size of the system file. The number of word permutations in each text unit is dependent on the number of words in it, the formula is:

word permutations = (number of words) ! / 2



name of system file: accept the generated file name or click change to open the file dialog.

file name of word permutations: accept the generated file name or click change to open the file dialog.

process all text units If you affirm this question, the complete text will be processed, otherwise the defined sample will be processed. Details are described in chapter 4 on page 37.

case folding enabled Letters can be treated as the same, if they are different only in their case (lower or upper case).

format of vocabulary: normal form or reverse form.

justification of vocabulary: left justified or right justified.

minimum length: the minimum number of characters a string must have-

maximum length: the maximum number of characters a string may have.

minimum frequency: the minimum number of occurences a string must have.

maximum frequency: the maximum number of characters a string may have-

name of exclusion list: If you enter a valid file name, all strings that are in the exclusion list will not be processed.

8.3.1 Information messages

```
TextQuest (tm) Text Analysis Software 27.02.2013 14:11
         program: WORDBOOK
     application: word permutations
input file        E:\tq4\text\bush.itx
output file       E:\tq4\text\bush.wp
options: upper-/lower case ignored
statistics:
strings (token) read:

- I 01:       289 text units
- I 03:     47677 words
- I 04:       104 numbers
- I 05:      5299 other
- I 06:     53080 total
- I 07:   164,979 words/text unit
- I 08:     0,360 numbers/text unit
- I 09:    18,329 other/text unit
- I 10:   183,668 total/text unit
            types     token   TTR   type of string
strings written:

- I 21:     28716     47679   0,602 words
- I 22:        96       104   0,923 numbers
- I 23:      1793      5297   0,338 other
- I 24:     30605     53080   0,577 total
WORDBOOK start:  14:11:35
WORDBOOK end:    14:11:37
WORDBOOK needed 2 seconds
```

Frequency statistics of vocabulary

occurence	frequency	percentage
1	23722	77,510
2	3879	12,674
3	1179	3,852
4	598	1,954
5	320	1,046
6	191	0,624
7	127	0,415
8	84	0,274
9	86	0,281
10	44	0,144
11	41	0,134
12	34	0,111
13	30	0,098
14	34	0,111
15	26	0,085
16	21	0,069
17	12	0,039
18	11	0,036
19	16	0,052
20	11	0,036
21	12	0,039
22	5	0,016
23	9	0,029
24	3	0,010
25	2	0,007
>25	108	0,353

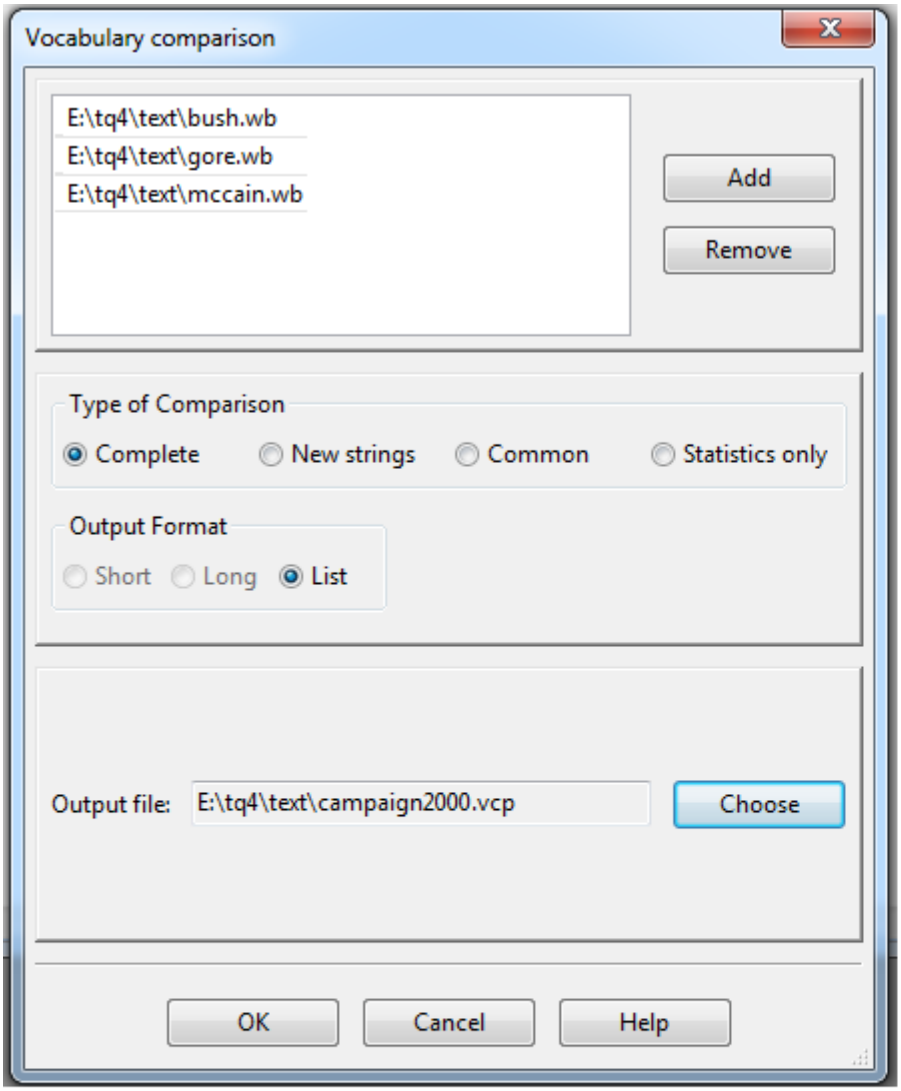
Length statistics of vocabulary

length	frequency	percentage
1	1	0,003
2	6	0,020
3	77	0,252
4	274	0,895
5	674	2,202
6	1447	4,728
7	2305	7,531
8	2912	9,515
9	3306	10,802
10	3422	11,181
11	3286	10,737
12	3031	9,904
13	2560	8,365
14	1989	6,499
15	1594	5,208
>15	387	1,264

8.3.2 Printed results of word permutations

1	!	1	! case	4	! It	4	! public	1	! U
17	! ”	2	! cases	1	! just	1	! re	1	! unborn
6	! ’	1	! Catholic	2	! know	1	! re-election	1	! unconstitutional
29	! ,	1	! certain	1	! L	1	! reasons	2	! under
30	! .	1	! challenged	1	! last	1	! recently	1	! United
2	! 000	1	! chances	3	! law	1	! requested	1	! Unlike
1	! 130	1	! church	1	! lawmakers	1	! restrictions	1	! unlikely
1	! 14th	2	! citizens	1	! laws	1	! returned	1	! up
1	! 212-square-mile	1	! clear	1	! lawyer	2	! right	1	! upheld
1	! 22	1	! co-exist	1	! legislation	3	! Roe	1	! v
1	! 90	1	! colony	1	! limit	1	! Roman	1	! vigorous
8	! a	1	! consideration	1	! major	1	! rule	1	! vocal
3	! abortion	1	! contrast	1	! make	4	! s	1	! Wade
2	! abortions	1	! could	1	! Making	7	! said	5	! was
1	! actually	8	! Court	1	! Marianas	1	! San	2	! way
1	! Ada	1	! daughter	1	! may	1	! seeking	5	! We
1	! advocacy	1	! decided	1	! members	2	! Senator	1	! welcome
1	! affected	1	! defend	2	! might	2	! she	1	! what
1	! after	1	! dependents	1	! military	1	! sign	2	! which
1	! against	1	! disturbing	1	! Minnesota	1	! signed	1	! While
1	! Air	1	! either	2	! Missouri	1	! silence	3	! who
4	! all	1	! end	2	! more	1	! sites	1	! whom
1	! allow	1	! enough	1	! Mr	1	! six	1	! winning
1	! allowed	1	! expected	2	! Ms	1	! so	1	! with
1	! Amendment	1	! expense	1	! Naval	1	! solely	1	! without
3	! Among	6	! for	1	! no	1	! southern	3	! would
1	! an	1	! Force	2	! not	1	! Spanish	1	! year
6	! and	1	! forces	1	! notification	1	! spoke	1	! years
1	! Anita	1	! Francisco	1	! now	1	! spring	293	”
1	! anonymity	1	! from	12	! of	2	! state	21	” !
1	! Another	1	! Government	2	! official	1	! States	999	” ”
1	! Anthony	10	! Guam	1	! Ohio	1	! strikes	37	” &
1	! appeal	1	! guaranteed	2	! on	1	! stupid	999	” ’
1	! Apuron	1	! handful	1	! opponent	1	! summer	20	” (
1	! Archbishop	3	! has	1	! opposition	1	! supporting	20	”)
1	! Archipelago	1	! have	1	! oppressive	5	! Supreme	999	” ,
3	! are	1	! hearing	3	! or	1	! t	285	” -
1	! armed	1	! heart	1	! organized	2	! take	9	” —
3	! Arriola	1	! held	1	! Originally	1	! teen-agers	4	” -related
1	! as	1	! her	1	! ornery	1	! territory	26	” -rights
5	! at	2	! here	2	! other	2	! than	999	” .
1	! bases	2	! History	3	! out	4	! that	124	” 000
1	! basic	1	! hoped	1	! overturned	37	! the	16	” 10
2	! be	1	! how	1	! parental	1	! their	81	” 100
3	! been	1	! I	1	! people	1	! them	6	” 10th
1	! Benshoof	2	! if	1	! percent	1	! then	5	” 12
6	! bill	6	! In	1	! political	1	! there	5	” 13
2	! but	1	! installation	1	! possible	1	! They	34	” 130
4	! by	1	! instance	1	! premise	1	! think	5	” 136
1	! C	7	! is	1	! privacy	3	! this	6	” 14

8.4 Comparison of vocabularies



Two or more vocabularies can be compared in one analysis. All strings that do not occur in the first file but in the other files can be written to an output file. The statistical information messages include inclusive and exclusive strings of all files. Umlauts are processed correctly, because the sort order table `sort.def` is used. All vocabularies must be sorted ascending by alphabet using the same sort order table.

The features for comparing two or more vocabularies are different, if you only compare two vocabularies you have more choices how to present the results.

The program compares (two or more) vocabularies in four types of analyses:

- complete: comparison of only two vocabularies with the differences of the strings
- new strings: output of the strings, that occur in all vocabularies but not in the first vocabulary
- common: comparison of strings that occur in all vocabularies
- statistics only, the comparison of the vocabularies is suppressed

The complete comparison can be written in three formats if two vocabularies are compared, otherwise the list format is the default format.

- short format: output are the frequencies of the first file, the second file, the differences between the two frequencies and the string. The frequencies are formatted in 9 digits. If a string occurs only in one file, the frequency field of the other file is left blank, the difference is not computed.

column	contents
1 - 9	frequency of the word in the 1. file
10 - 18	frequency of the word in the 2. file
19 - 27	difference of the frequencies
28	free
29 -	word

- long format: the first 39 characters of the strings of each file followed by its frequency displayed in 7 digits. Between the two columns the differences of the frequencies are shown in 7 digits.

columns	contents
1 - 7	frequency of the word in the 1. file
8 - 46	word in the 1. file
47 - 53	difference of the frequencies
54 - 60	frequency of the word in the 2. file
61 - 99	word in the 2. file

- list format: the frequencies of all strings that occur in all files are output as well as their difference and the string. The frequencies and the differences use 9 digits.

columns	contents
1 - 9	frequency of the word in the 1. file
10 - 18	frequency of the word in the 2. file
19 - 27	difference of the frequencies
28	free
29 -	word (unlimited length)

8.4.1 Parameters of the program

- **add:** the name of a file is required and included in the comparison. The name may contain drive and/or directory specifications. You can add up to 20 files.
- **remove:** the selected file is removed from the vocabulary comparison.
- **type of vocabulary comparison:**
  - complete: comparison of only two vocabularies with the differences of the strings
  - new strings: all strings that occur in all vocabularies but not in the first vocabulary are written to a file
  - common: comparison of strings that occur in all vocabularies
  - statistics only, the comparison of the vocabularies is suppressed
- **output file of vocabulary comparison:** The file name of the results is specified here: either containing the complete comparison or the new strings.
- **format of the vocabulary comparison:**
  - short: the frequencies of the strings in the first file, the second file, the difference and the string are written. The counters have 9 digits.
  - long: for each file the frequencies and the strings are written, both parts are separated by the difference. The counters have 7 digits, the strings are truncated after 39 characters.
  - list: the frequencies (9 digits) of all files and strings are written in one line.

8.4.2 Information messages

```
TextQuest (tm) Text Analysis Software 13.04.2007 22:19
     application: vocabulary comparison
Input file 1     D:\texts\bush.wb
Input file 2     D:\texts\gore.wb
Input file 3     D:\texts\mccain.wb
Statistics
                    File 1                  File 2                  File 3                  All Files
                Types   Tokens    TTR   Types   Tokens    TTR   Types   Tokens    TTR
strings read:
words            1293     4117  0.314    1031     3235  0.319    1334     4285  0.311
digits             12       12  1.000      15       19  0.789       4        4  1.000
other              10      613  0.016       9      437  0.021      10      382  0.026
sum              1315     4742  0.277    1055     3691  0.286    1348     4671  0.289

exclusive strings:
words             624      727  0.858     520      673  0.773     700      865  0.809
digits              9        9  1.000      13       17  0.765       2        2  1.000
other               2        2  1.000       0        0  0.000       1        4  0.250
sum               635      738  0.860     533      690  0.772     703      871  0.807

sum of common strings:
words             266     2643  0.101     266     2094  0.127     266     2661  0.100      798     7398  0.108
digits              1        1  1.000       1        1  1.000       1        1  1.000        3        3  1.000
other               8      611  0.013       8      433  0.018       8      376  0.021       24     1420  0.017
sum               275     3255  0.084     275     2528  0.109     275     3038  0.091      825     8821  0.094
```

8.4.3 Different outputs of vocabulary comparison

Output of a comparison of 3 word lists

Texts: Speeches on foreign affairs in the 2002 US-Presidental Campaign, geographic entries

Bush	Gore	McCain	word (some were grouped)
	1		African
	6		Albanian Albanians
60	27	26	America American Americans
7		3	Asia Asian
3		4	Beijing
1		2	Berlin
	1		Bosnia
21		21	China
6			Eurasia
5	12	4	Europe European
2	2	2	German Germans Germany
2		1	Gulf
	1		Herzegovina
3			India
		2	Iraq Iraqis
1		10	Israel Israelis
3		2	Korea Korean
	15	1	Kosovar Kosovars Kosovo
	1		Montenegro
1	6	7	NATO
1	1	1	Pacific
1			Pakistan
1			Philippines
34		25	Russia Russian Russians
	7	4	Serb Serbia Serbian Serbs
	1		Slovenia
4		5	Taiwan
1			Thailand
		1	Tibetan
1			Ukraine
		2	UN
		1	Uruguay
		1	Warsaw

Vocabulary comparison in long format

104	A	30	74	A
			2	abandon
1	abandons			
			1	ability
1	able			
			1	ABM
			1	abolish
1	abortion			
4	about	-3	7	about
1	above			
1	abroad	-1	2	abroad
			1	abrogate
			1	absence
			1	abundantly
			1	abuses
			1	accelerated
			2	accept
			1	access
1	accidental			
			1	accommodate
1	accomplished	0	1	accomplished
			1	accomplishment
			1	accord
1	accountable			
1	accounted			
1	accounting			
1	accounts	0	1	accounts
1	accurate			
1	achieve			
1	achievement			
			1	acknowledge
4	across	3	1	across
2	act	0	2	act
			1	acting
2	action	0	2	action
			1	actions
1	active			
1	actively	0	1	actively
1	activity	-1	2	activity
			1	adapts
			3	add
1	address	-2	3	address

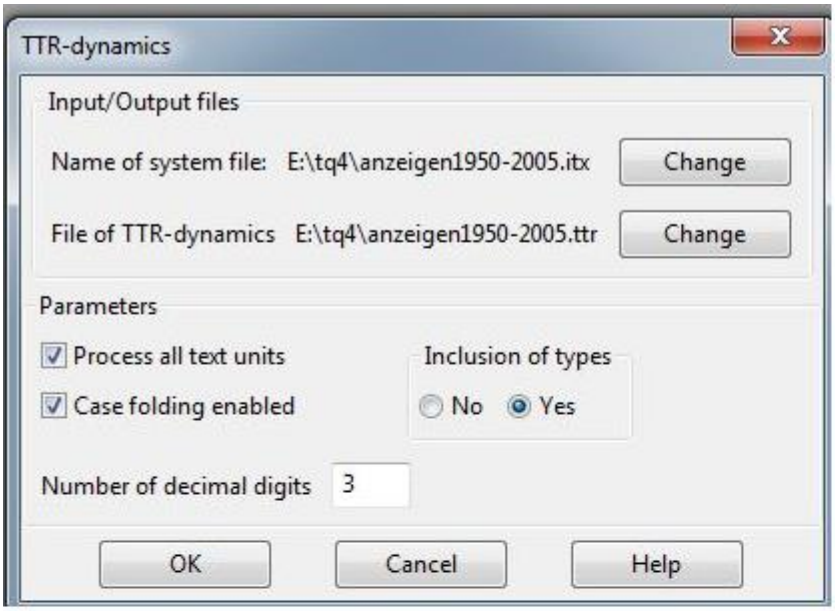
Vocabulary comparison in short format

104	74	30	A
	2		abandon
1			abandons
	1		ability
1			able
	1		ABM
	1		abolish
1			abortion
4	7	-3	about
1			above
1	2	-1	abroad
	1		abrogate
	1		absence
	1		abundantly
	1		abuses
	1		accelerated
	2		accept
	1		access
1			accidental
	1		accommodate
1	1	0	accomplished
	1		accomplishment
	1		accord
1			accountable
1			accounted
1			accounting
1	1	0	accounts
1			accurate
1			achieve
1			achievement
	1		acknowledge
4	1	3	across
2	2	0	act
	1		acting
2	2	0	action
	1		actions
1			active
1	1	0	actively
1	2	-1	activity
	1		adapts
	3		add
1	3	-2	address

Vocabulary comparison list format

104	74	A
	2	abandon
1		abandons
	1	ability
1		able
	1	ABM
	1	abolish
1		abortion
4	7	about
1		above
1	2	abroad
	1	abrogate
	1	absence
	1	abundantly
	1	abuses
	1	accelerated
	2	accept
	1	access
1		accidental
	1	accommodate
1	1	accomplished
	1	accomplishment
	1	accord
1		accountable
1		accounted
1		accounting
1	1	accounts
1		accurate
1		achieve
1		achievement
	1	acknowledge
4	1	across
2	2	act
	1	acting
2	2	action
	1	actions
1		active
1	1	actively
1	2	activity
	1	adapts
	3	add
1	3	address

8.5 Vocabulary growth – TTR-dynamics



The TTR is the type-token ratio, where *types* is the number of different strings and *token* is the number of all strings. The ratio is between 1 and 0: 1 means, that each word occurs only once in the text, 0 is never reached. The larger the text is, the lower the TTR will become, it is dependent on the length of the text.

After each word the current value for the TTR is calculated and written to an output file. These data can be processed with other programs, e.g. those written by Gabriel Altmann. The types can be supressed in the output file to save disk space.

TTR dynamics are calculated only for strings that have a letter or a digit as first character. After each token the value of the TTR is recalculated. Sampling is supported. The output file consists of the token, the cumulated values for types, tokens and the TTR after each token. TTR dynamics show the growth of the vocabulary of a text. The value of the TTR starts with 1 and decreases in general, sometimes it increases. Useful for interpretation are the increases and the number of tokens if certain values are reached, especially for the comparison of texts. The number of increases and decreases of the value and the values after 100, 200, 300, 400, 500, ..., 1000, 2000, ..., 10000, 20000, ..., 100000 tokens are computed also.

name of system file: accept the generated file name or click change to open the file dialog.

file of TTR-dynamics: the name of the file where the TTR-dynamics are written to. Accept the generated file name or click change to open the file dialog.

process all text units If you check this box, the complete text will be processed, otherwise the defined sample will be processed (see chapter 4 on page 37).

case folding enabled Letters can be treated as the same, if they are different only in their case

(lower or upper case).

inclusion of types: If not checked, the types are not written to the output file, which reduces its size dramatically. Also the data can be processed more easily with Gabriel Altmanns programs, e.g. Altmann Fitter to analyse the distribution.

number of decimal digits: the precision of the TTR-values can be specified between 1 and 5, default is 3 digits.

8.5.1 Information messages

```
TextQuest (tm) Text Analysis Software 19.04.2007 20:57
         program: WORDBOOK
     application: TTR-dynamics
input file        D:\texts\bush.itx
output file       D:\texts\bush.ttr
options: upper-/lower case ignored
statistics:
strings (token) read:
- I 01:       493 text units
- I 03:      4117 words
- I 04:        12 numbers
- I 05:       613 other
- I 06:      4742 total
- I 31:      1288 TTR-values ascending
- I 32:      2824 TTR-values descending
- I 33:        17 TTR-values unchanged
- I 34:    0,3161 TTR raw value
- I 35:    0,4561 TTR-quotient
- I 36:    0,6600 TTR value at 100 token
- I 37:    0,5950 TTR value at 200 token
- I 38:    0,5767 TTR value at 300 token
- I 39:    0,5250 TTR value at 400 token
- I 40:    0,5140 TTR value at 500 token
- I 41:    0,4983 TTR value at 600 token
- I 42:    0,4843 TTR value at 700 token
- I 43:    0,4725 TTR value at 800 token
- I 44:    0,4589 TTR value at 900 token
- I 45:    0,4510 TTR value at 1000 token
- I 46:    0,3850 TTR value at 2000 token
- I 47:    0,3407 TTR value at 3000 token
- I 48:    0,3198 TTR value at 4000 token
WORDBOOK start:  20:57:07
WORDBOOK end:    20:57:07
WORDBOOK needed 0 seconds CPU time
```

8.5.2 Results of TTR dynamics

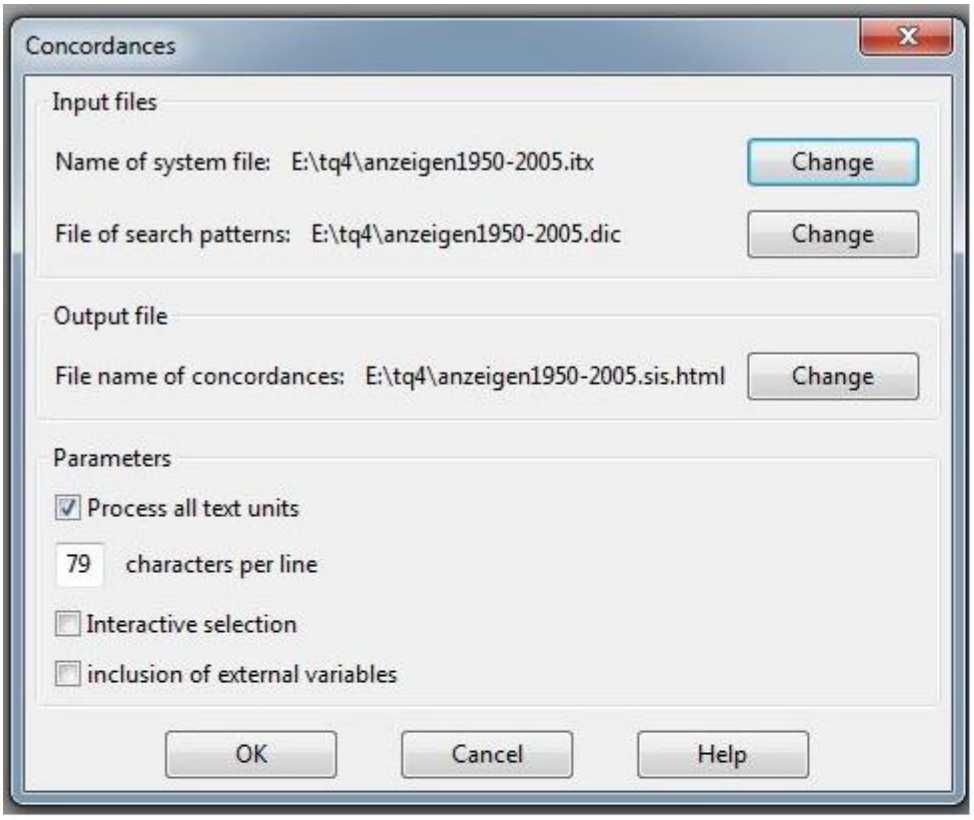
type	token	TTR	strings
1	1	1.000	Under
2	2	1.000	Pressures
3	3	1.000	and
4	4	1.000	Stigma
5	5	1.000	More
6	6	1.000	Doctors
7	7	1.000	Shun
8	8	1.000	Abortion
8	9	0.889	Under
9	10	0.900	siege
10	11	0.909	from
11	12	0.917	protesters
11	13	0.846	and
12	14	0.857	largely
13	15	0.867	isolated
13	16	0.813	from
14	17	0.824	medical
15	18	0.833	colleagues
15	19	0.789	Doctors
16	20	0.800	who
17	21	0.810	perform
18	22	0.818	abortions
19	23	0.826	say
20	24	0.833	they
21	25	0.840	are
22	26	0.846	being
23	27	0.852	heavily
24	28	0.857	stigmatized
24	29	0.828	and
25	30	0.833	fewer
25	31	0.806	and
25	32	0.781	fewer
25	33	0.758	Doctors
25	34	0.735	are
26	35	0.743	willing
27	36	0.750	to
28	37	0.757	enter
29	38	0.763	the
30	39	0.769	field
31	40	0.775	Reflecting
31	41	0.756	the
32	42	0.762	public
33	43	0.767	s
34	44	0.773	ambivalence
35	45	0.778	about
35	46	0.761	Abortion
36	47	0.766	many
36	48	0.750	Doctors

8.6 Concordances – KWIC

Concordances show the context of search patterns and are used as a tool for lexicology or to examine ambiguity. The search pattern is centered in the middle of the line, and its context flows around it. The size of the context can be specified in characters per line.

The concordances are written to an output file. Words, word sequences and word co-occurences can be search patterns, details are described in chapter 5 on page 39.

The following picture shows the parameters:



name of system file: you can accept the generated file name or click the change button to open the file dialog.

file of search patterns: the name of the file where the search patterns are stored (`*.dic`-file). The number of search patterns is discussed in chapter 5 on page 39. You can accept the generated file name or click the change button to open the file dialog.

process all text units If you check this box, the complete text will be processed, otherwise the defined sample will be processed (see chapter 4 on page 37).

file name of concordances: the file name the concordances are written to. You can accept the

generated file name or click the change button to open the file dialog.

line length: The default value for concordances ist 79 characters. The value is dependent on the output medium (screen or printer). The context can be enlarged if external variables are not included.

interactive selection: yes means, that each occurence requires an answer whether it is to be written to the output file or not. No means, that all occurrences are included.

inclusion of external variables If this question is denied, the concordances are only written together with their codes, the external variables are suppressed.

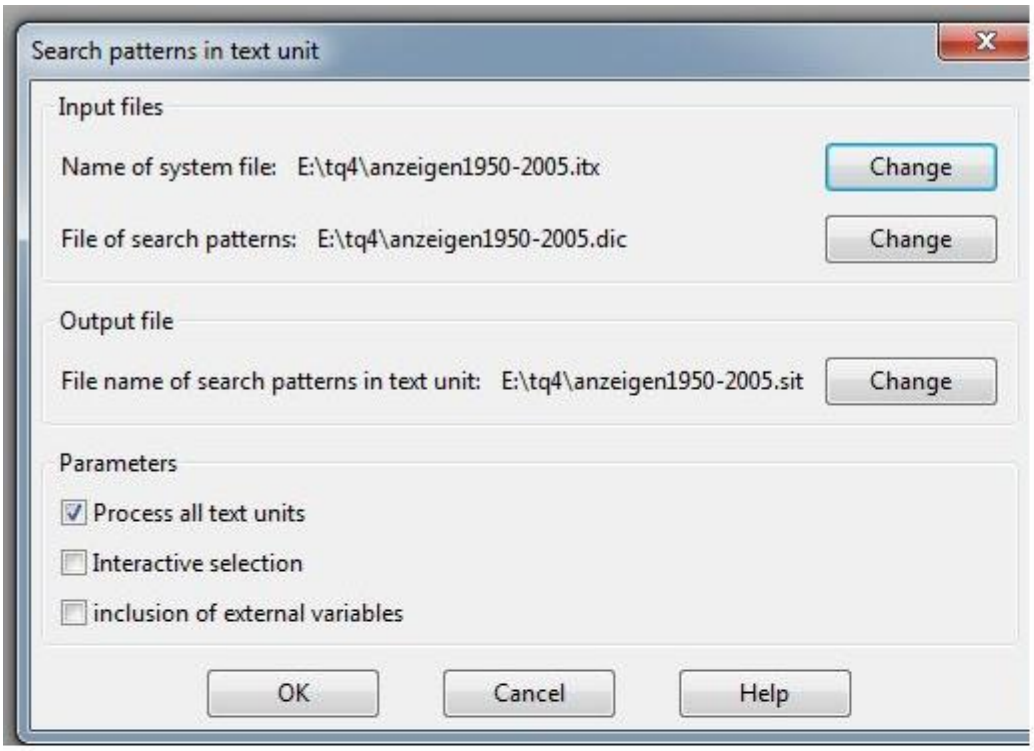
8.6.1 Information messages

```
TextQuest (tm) Text Analysis Software 11.11.2006 11:41
         program: SUWACO
     application: concordance
input file        D:\texts\CONTAKT.itx
category file     D:\texts\CONTAKT.dic
concordance file  D:\texts\CONTAKT.sis

- C 01:      1363 search patterns processed
- C 06:       194 with option C marked search patterns
- C 07:      1148 with option U marked search patterns
- C 09:         8 with option D marked word root chains
- C 10:         8 with option F marked word root chains

- I 01:      6315 text units read
- I 03:     62513 words read
- I 17:     16281 output records in SIC file
- I 26:         0 negation(s)
SUWACO start:  11:41:38
SUWACO end:    11:41:48
SUWACO needed 10 seconds CPU time
```

8.6.2 Printed output of a concordance in KWIC-format

7	from being perfect and that a selfish	**minorit**y in every walk of life farmin
3	ich Americans have demanded since the	**white** man first came to these shores
7	most always loses . Only a very small	**minorit**y of the people of this countr
1	practices ; every profession has its	**black** sheep , but long experience in
7	kinds of unfair practices by selfish	**minorit**ies which unfortunately did mo
8	ies must be without stint and without	**discrimina**tion . No sectional , no po
7	pinion is at war with a power-seeking	**minorit**y . That is no new thing . It
7	of partisan politics . They seek-this	**minorit**y in business and industry to
8	the principles of bettering the human	**race** by peaceful means . Within those
8	an be wholly alien to you . The human	**race** now passes through one of its gr
8	osing this plan have sought to arouse	**prejudic**e and fear by crying that I a
8	make it dependent upon the desire or	**prejudic**e of any individual Justice ?
8	ifies a national approach , free from	**prejudic**e of partisanship and warrant
8	ion must be removed from the field of	**prejudic**e to the field of logic . We
7	nd to be good citizens . Only a small	**minorit**y have displayed poor citizens
7	purpose on the part of the condemned	**minorit**y to distort the criticism int
8	ation , special interests or economic	**prejudic**es in whatever program may be
8	hat immoderate statement , appeals to	**prejudic**e , the creation of unkindnes
7	and would no longer oppose that small	**minorit**y which , in spite of its own
8	tion , personal attack and appeals to	**prejudic**e . It would be a lot better
8	f ties of good will , the stirring of	**prejudic**e and the incitement to disun
8	y self-distrust , endangered by class	**prejudic**e , by dissension between cap
8	n among them be terminated , that the	**race** of armaments cease and that comm
8	fferences of occupation , geography ,	**race** and religion no longer obscure t
8	r-glass tells us that we are off on a	**race** to make democracy work , so that
1	y power to prevent , there will be no	**black**-out of peace in the United Stat
8	d the radio use the utmost caution to	**discrimina**te between actual verified
8	the people of this country will also	**discrimina**te most carefully between n
8	areas from restricted transportation	**discrimina**tions , the extension of th
8	against group , faith against faith ,	**race** against race , class against cla
8	, faith against faith , race against	**race** , class against class , fanning
7	e some among us who were persuaded by	**minorit**y groups that we could maintai
7	pecialized group , which represents a	**minorit**y of the total employees of a
8	olitical is encouraged to exploit its	**prejudic**es through false slogans and
8	rge- a group that may be sectional or	**racial** or political is encouraged to
7	, expressed the attitude of the small	**minorit**y who want to see no evil and
8	leading to the most gigantic armament	**race** and the most devastating trade w
8	elf to dominate and enslave the human	**race** . The British people and their a
8	They try to reawaken long slumbering	**racial** and religious enmities which s
8	less than from Washington to Denver ,	**Color**ado five hours for the latest ty
3	you hold your fire until you see the	**whites** of his eyes , you will never k
5	eir plans do not stop there , for the	**Indian** Ocean is the gateway to the fa
8	mmunists , and every group devoted to	**bigot**ry and racial and religious into
8	nd every group devoted to bigotry and	**racial** and religious intolerance . It
8	ther to make war upon the whole human	**race** . Their challenge has now been f
8	majority of the members of the human	**race** are on our side . Many of them a
1	vere until the destruction of all his	**black** designs upon the freedom and sa
8	particularly vigilant against racial	**discrimina**tion in any of its ugly for

8.7 Search patterns in the text unit



Search patterns in text unit are similar to concordances, the context however is not limited by a number of characters but is the whole text unit. All kinds of search patterns are possible, details are described in the chapter on the definition on search patterns on page 39. The results are written to the output file and can be processed by other programs. The output file consists of lines that start with the search pattern. After a blank the whole text unit follows. Displaying the results the search patterns can be underlined, **bold face** or in *italics*.

name of system file: the name of the file where the system file is stored. You can accept the generated file name or click the change button to open the file dialog.

file of search patterns: the name of the file where the search patterns are stored (`*.dic`-file). The number of search patterns is discussed in chapter 5 on page 39. You can accept the generated file name or click the change button to open the file dialog.

file name of search patterns in text unit: the name of the output file that contains the search patterns in text unit. You can accept the generated file name or click the change button to open the file dialog.

process all text units If you check this box, the complete text will be processed, otherwise the defined sample will be processed (see chapter 4 on page 37).

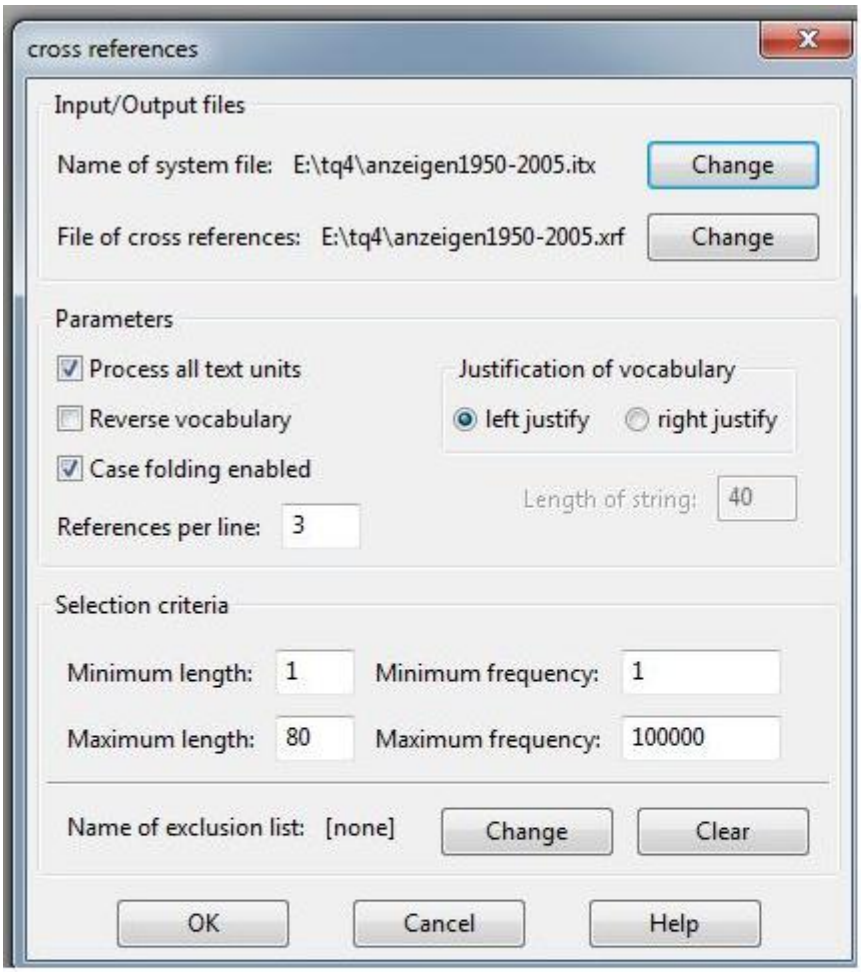
interactive selection: yes means, that each occurence requires an answer whether it is to be written to the output file or not. No means, that all occurrences are included.

8.7.1 Information messages

The result window looks like this:

```
TextQuest (tm) Text Analysis Software 24.11.2006 00:43
         program: SUWACO
     application: search units in text unit
negation test enabled
input file        D:\texts\CONTAKT.itx
category file     D:\texts\CONTAKT.dic
SIT file          D:\texts\CONTAKT.sit

- C 01:      1363 search patterns processed
- C 06:       194 with option C marked search patterns
- C 07:      1148 with option U marked search patterns
- C 09:         8 with option D marked word root chains
- C 10:         8 with option F marked word root chains

- I 01:      6315 text units read
- I 03:     62513 words read
- I 17:     16343 output records in SIC file
- I 26:         0 negation(s)
SUWACO start:  00:43:00
SUWACO end:    00:43:10
SUWACO needed 10 seconds CPU time
```

8.8 Cross references



A cross reference of a text consist of all occurences of each string together with its external variables and the position of the string (number of the string in the text unit), sorted by alphabet.

In texts with hierarchical external variables cross references should be unique, that means, that no string should have the same external variables and the same position. If this is the case, there maybe incorrect external variables in the text.

Samples of the text units can be drawn. Words, digits and other types of strings are counted, and the average length of a text unit (in words, not in strings) is computed. If the sort order table `sort.def` exists, it will be used (see page **??** for details). Also case folding can be enabled or disabled. The frequency of the string is written after the last reference into a separate line. Strings can be excluded from processing if they occur in an exclusion list.

name of system file: the name of the file where the system file is stored. You can accept the generated file name or click the change button to open the file dialog.

file name of cross references: the name of the file where the cross references are written to. You

can accept the generated file name or click the change button to open the file dialog.

process all text units If you check this box, the complete text will be processed, otherwise the defined sample will be processed. Details are described in chapter 4 on page 37.

case folding enabled Letters can be treated as the same, if they are different only in their case (lower or upper case).

format of vocabulary: normal form or reverse form.

justification of vocabulary: left justified or right justified.

number of references per line: Here the number of cross references per line are to be specified. The minimum value is 1, every cross reference starts with a new line. It consists of the external variables separated with tildes. The cross references are separated by a blank.

minimum length: the minimum number of characters a string must have to be included in the vocabulary.

maximum length: the maximum number of characters a string may have to be included in the vocabulary.

name of exclusion list: If you enter a valid file name, all strings that are in the exclusion list will not be processed.

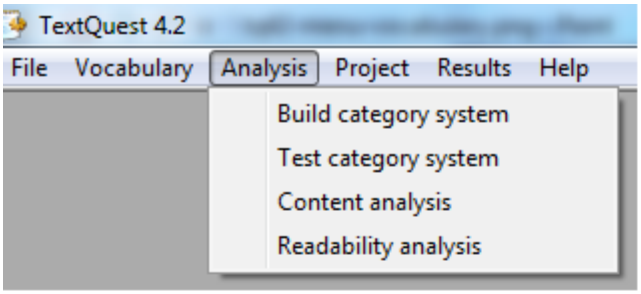
8.8.1 Information messages

```
TextQuest (tm) Text Analysis Software 27.11.2006 13:37
         program: WORDBOOK
     application: cross references
input file        D:\texts\CONTAKT.itx
output file       D:\texts\CONTAKT.xrf
options: sort table SORT.DEF used upper-/lower case ignored
statistics:
strings (token) read:
length    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15  >15
freq  62242  122    2    1    1    0    0    0    0    0    0    0    0    0    0    0

- I 01:      6315 text units
- I 03:     41097 words
- I 04:      3805 numbers
- I 05:     17599 other
- I 06:     62501 total
- I 07:     6.508 words/text unit
- I 08:     0.603 numbers/text unit
- I 09:     2.787 other/text unit
- I 10:     9.897 total/text unit
- I 11:       126 words with identical external variables
            types     token   TTR   type of string
strings written:

- I 21:     41064     41097   0.999 words
- I 22:      3805      3805   1.000 numbers
- I 23:     17499     17599   0.994 other
- I 24:     62368     62501   0.998 total
WORDBOOK start:  13:37:30
WORDBOOK end:    13:37:32
WORDBOOK needed 2 seconds CPU time
```

8.8.2 Printed results of cross references

9. The menu: analyses of texts

The analyses available are either a content analysis or a readability analysis. The category manager supports the creation and maintenance of category systems. The readbility analysis is based on readability formulas.



9.1 Content analysis

A content analysis can be regarded as a rule based system to transform text information into numeric information. Categories have to be defined with numeric codes, and each category must consist of at least one search pattern. Each search pattern must be a valid indicator for the category that it belongs to. An example: you have a category for animals, so search patterns can be *dog, cat, cow, bird, monkey, elephant, crocodile* etc.

The results of a content analysis are written to files and can be processed by other programs. The same regulations for search patterns described in the last chapter apply.

The results are both numeric and text: the coding results can be calculated as frequencies for each category or as sequence of categories within a text unit. The rapport files allow you to validate the results of the coding process and show you coded, uncoded, negated and/or ambiguous text units as well as a complete coding control. The interactive coding mode can be used to handle potentially ambiguous and/or negated search patterns.

The most common case is that you create a category system with search patterns according to your hypotheses. However, you can also use one of the standardised category systems that are included.

Their use is easy: all the files you need are already there. It can be necessary to extend a category system, because words occur in the text that should belong to the category system, but are not part of the category system.

The following category system are delivered with **T**ext**Q**uest:

file	Name and author/translator	cat.	entries
	English		
rid-eng	Regressive Imagery Dictionary: Martindale	65	3939
harvard	Harvard Psychological Dictionary: Stone	105	16810
liwc-eng	Linguistic word count: Pennebaker	68	5714
	German		
rid-ger	Regressive Imagery Dictionary: Martindale	43	4577
liwc-ger	Linguistic word count: Pennebaker, Mehl	68	12238
hkw	Hamburg communication-sociological dictionary: Deichsel	86	5972
daw2003	Dresden anxiety dictionary: Berth	6	1493
kontakt	personal ads: Giegler, Klein	38	1363
nfaktor	television news factors: Klein	16	889
	Spanish		
liwc-esp	Linguistic word count: Pennebaker	68	7460
	French		
harvardf	Harvard Psychological Dictionary: Hogenraad	105	27139
	Italian		
liwc-it	Linguistic word count: Pennebaker	69	7610

9.1.1 Build category systems

Before a content analysis can be performed, a category system has to be selected of a new one constructed. **T**ext**Q**uest needs a file of search patterns and a file of category labels, both are dependent on each other.

Since version 3.0 there are two ways to do this:

using an editor or a word processor to create the files You can use any editor or word processor for these files, a good idea is to take the sample files as a template. The files have to be saved unformatted as UTF-8-formatted or text format with carriage return/line feed (CR/LF).

using the category manager of TextQuest (since version 3.0) This manager allows the adding, change, and deletion of categories or search patterns. Its use makes constructing of a category system much easier than before, because errors in the syntax of search patterns and parameters are not possible. The category manager creates both the files for the category labels and the file of search patterns.

9.1.1.1 Category labels

The category labels support the documentation of codes and their meaning. Definitions of category labels are compulsory, the definitions must be stored in a file. The category manager will create this file automatically. However, if you use a text editor, each line of this file contains – starting on column 1 – the number of the category. A blank follows, and after the blank you write the category label for the category. The maximum length is 60 characters. If it is longer, it will be truncated. Up to 999 different category labels are possible.

Example for a file with category labels:

```
1 character
2 inner values
3 attractiveness
4 intellectual mobility
```

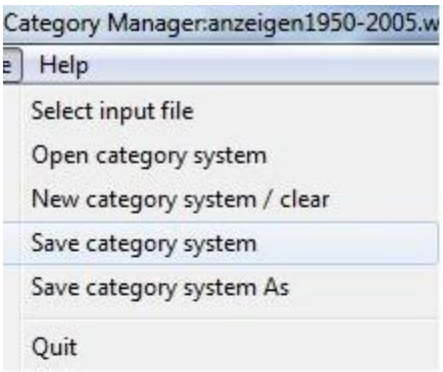
The file `contakt.lab` contains another example with more categories. If you use the category manager, the label file is created automatically. There are other files with the file extension `*.lab` that show you how the label file looks like. Of course you can use these files as templates for your own project.

9.1.1.2 Category system

A content analysis is based on a category system that consist of search patterns stored in a file (`*.dic`-file). The category manager is a tool that lets you construct a category system and maintain it. One strategy for the generation of search patterns is to load the word list and look for single meaningful words. You can also generate word sequences and/or word permutations and use for the construction of categories based on word combinations and thus including the context of a single word. The category manager is explained in detail in the next section.

Alternatively you can use a text editor to create the files for a category system, but this is cumbersome.

9.1.2 The category manager



Although you can construct a category system by using a simple text editor and create the file of search patterns (`.dic`) and the file of category labels (`*.lab`), this manager makes this task much easier. The first column shows the files that can be used and/or created.

select input file: you select a file that can be used as a basis for the construction or maintenanance of a category system. The typical file to select is a word list. If you are coding answers of open ended questions, it is also possible to load the file of uncoded text units.

open category system: you load the files of a category system, these files are shown in the second column of the category editor: the file of category labels (`*.lab`) and the file of search patterns (`*.dic`).

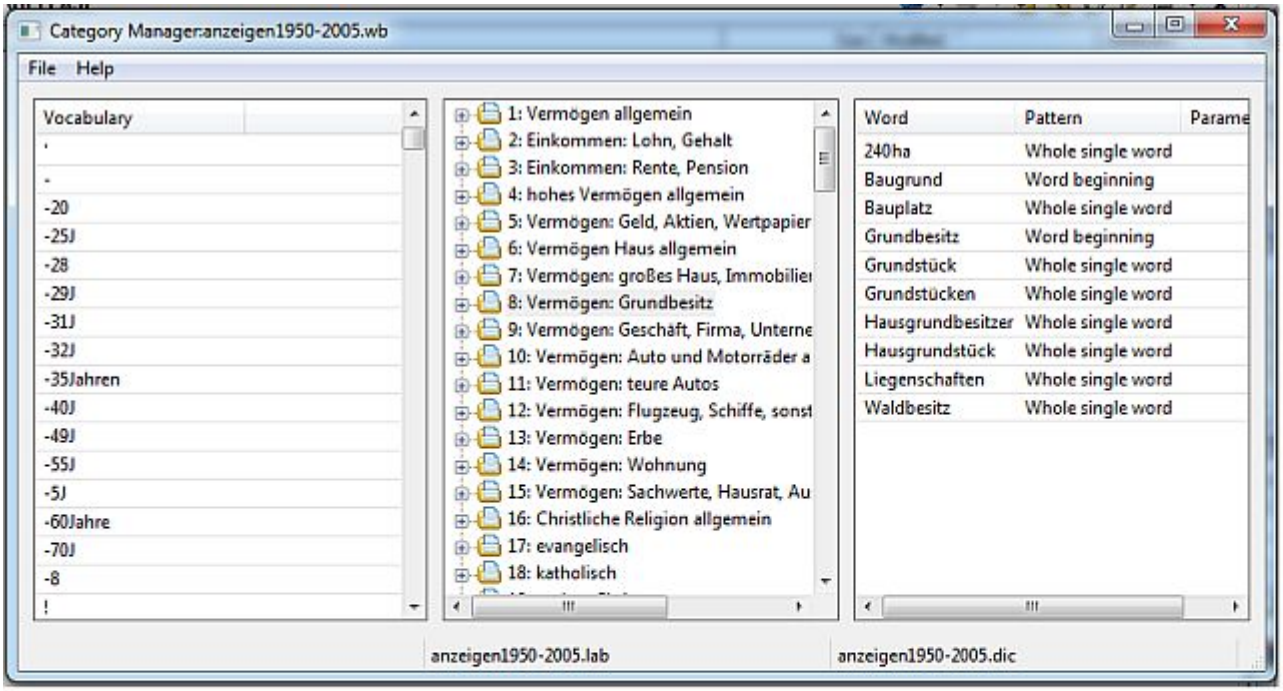
new category: you can create a new category system, at first you create the categories, afterwards you can create search patterns for each category.

save category system: you save the category system on the current drive in the current folder/directory.

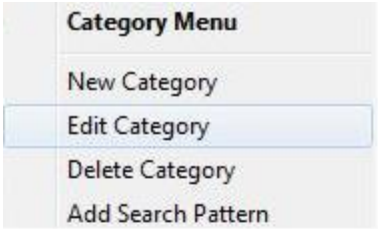
save category system as: you save the category sytem and select the drive and/or folder/directory.

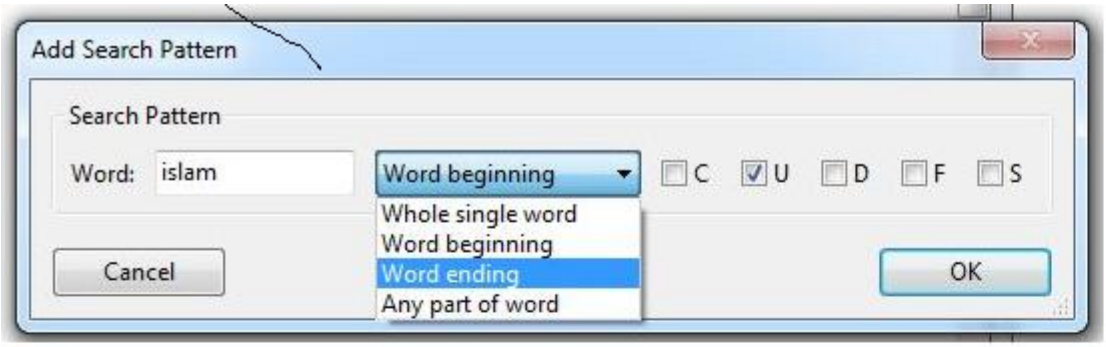
quit: you leave the category manager without saving the results. Warning: all unsaved material is lost and will not be saved.

You usually load a word list and a category system and can start working, the following picture shows you the environment.



- the left column shows the input file
- the middle column shows the categories and its search patterns
- the right column shows all options of a search pattern that can be specified in the parameter field. You have to tick or untick the boxes.





If you want to add a new search pattern to the categoy system, proceed as follows:

- move the mouse to a category in the middle column
- right click and choose *add search pattern*
- move the mouse to an entry in the left column, click and hold the left mouse button and drag it into the word field of the search pattern in the middle column
- you can change the parameter field options right to the entry

The parameter field can be used to control the treatment of each search pattern. The following parameters are possible:

C coding control All text units that contain the search patterns are written to the file of coded text units. If interactive coding is enabled, the text unit, the search pattern, the category number and the corresponding label are displayed. The coding decision (yes or no) and the code can be specified.

U Uppercase All characters of the search pattern are translated into uppercase, so that lower case and upper case are treated as the same. This is useful with words that are capitalised because they are at the beginning of a sentence.

N negation The search pattern is checked for negation. If an odd number of indicators before and after the search pattern occurs, the search pattern is not coded. The search pattern is coded when an even number (e.g. double negation – litotes) of indicators occurs. The number of words before and after the search pattern were indicators are searched can be specified (default: 2), also the list of indicators – separately for before (`neg-pre.def`) and after (`neg-post.def`) the search pattern.

9.1.3 Test category system

This test checks whether a search pattern is a part of another one or if it occurs more than once. If this is the case, the danger of multiple coding arises which leads to weighting and biasing the results. This time consuming test is done with the category system and also tests, whether parts of word roots occur in other search patterns. The results are written to a file.

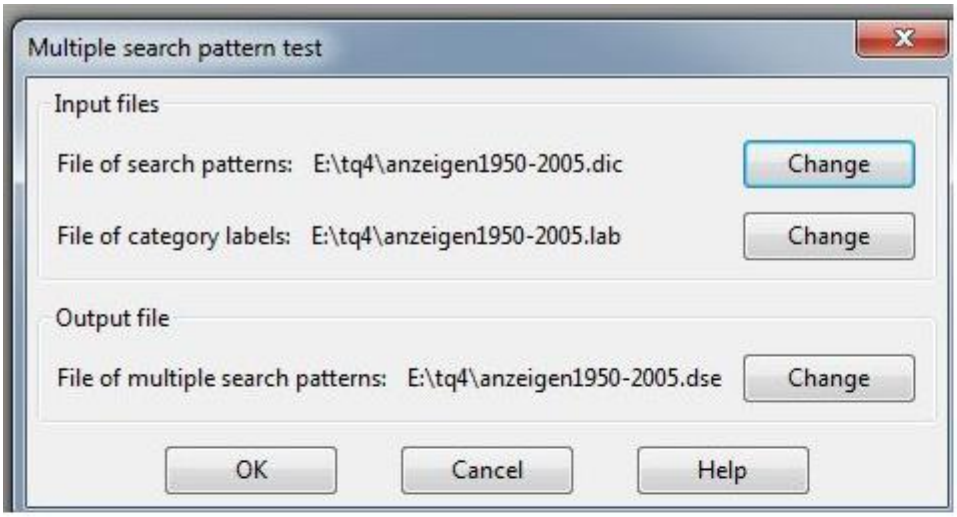
The first test is whether search patterns that are no word co-occurences are part of another such search pattern, e.g. *men* is a substring of women. The code of the multiple search patterns are compared, because the same code influences the weighting of the results, whereas different codes influences the selectivity of the category system.

The second test compares each search pattern that is not a word co-occurence whether it is a part of a word co-occurence, also considering the codes. This test is more a warning, because one might find a lot of the results neither affect the weighting or the selectivity. E.g. if a search pattern is *men* and a word co-occurence of a search pattern is *women of today*>, the test finds that *men* is part of *women* and so pointing to a text passage where a false coding can occur if this word co-occurence is found.

The results consist of:

type of search pattern	match	code
word (sequences)	substring	same
word (sequences)	substring	different
word (sequences)	string	same
word (sequences)	string	different
word in word co-occurence	string	same
word in word co-occurence	string	different

The following picture shows the parameters:



file of search patterns: the name of the file where the search patterns are stored (`*.dic`-file). The number of search patterns is discussed in chapter 5 on page 39. You can accept the generated file name or press the change button to open the file dialog.

file of category labels: this file contains the category system with codes and their labels. You can accept the generated file name or press the change button to open the file dialog.

file name of multiple search patterns: this file contains all search patterns with their codes and labels, that occur more than once within the category system (`*.dic`-file) or are part of another search pattern. You can accept the generated file name or press the change button to open the file dialog.

9.1.4 Results of the multiple entry test

line		search pattern	code	category label
519	<	WITW>	16	[Lebenseinschnitte]
311	<	BEAMTENWITW>	8	[hoher ökonomischer Status]
445	<	UFO>	15	[Metaphorik]
395	<	UFO>	13	[Metaphysik]
614	<	BIENE>	17	[Sex]
411	<	BIENE>	15	[Metaphorik]
537	<	BRIEFMARK>	17	[Sex]
413	<	BRIEFMARK>	15	[Metaphorik]
861	<	KATER >	26	[part. Verhalten - erotisch getönt]
425	<	KATER >	15	[Metaphorik]
862	<	KATZE >	26	[part. Verhalten - erotisch getönt]
426	<	KATZE >	15	[Metaphorik]
543	<	KOKOSNüSSEKNACKEN >	17	[Sex]
427	<	KOKOSNüSSEKNACKEN >	15	[Metaphorik]
612	<	MISSIONAR>	17	[Sex]
429	<	MISSIONAR>	15	[Metaphorik]
1294	<	NACHTEULE>	41	[gesellschaftliche Aktivität]
432	<	NACHTEULE>	15	[Metaphorik]
151	<	RHEIN>	3	[örtliche Gebundenheit]
440	<	RHEINLäNDER>	15	[Metaphorik]

The first column contains the line where the search pattern occurs within the (`*.dic`-file). The second columns contains the search patterns itself, the third column contains the category number followed by its meaning in square brackets. In the first block you see that the first search pattern is a part of the second one, both having different categories. You have to decide whether it makes sense to change this.

The second block shows you that one search pattern is a member of two different categories, so if this happens, you have to decide which category fits best. If you do not do that you violate one of the rules of a category system: categories have to be selective. The next blocks are examples of that phenomen, too. Only the last block is like the first block: one search pattern is part of another one.

9.1.5 Results of the coding

The content analysis is based upon the fact that search patterns are looked for in each text unit. This is called coding. If a search pattern is found, its code will be processed further on. The possibilities to define search patterns are described in chapter 5 on page 39. The results are written into the appropriate output files and can be analysed with statistical software; a setup for SAS, SPSS, ConClus, or SimStat (called script) can be generated.

The coding results can be written to the output file in two modes:

- vector file: the codes are written to the output file in the order they occur within the text unit.
- tabulation file: for each code there is a counter that holds the frequency for the code in the text unit. These counters are written to the output file after each text unit. The size of the tabulation file is calculated from the number of categories of the category system, each counter must not exceed 999 within a text unit.

The codes of the files may have up to three digits (values 1 to 999). If this limit is exceeded, an error message is displayed providing more information. The coding does not take the context into account, so that ambiguties of search patterns or negations are not recognised and can result in errorneous codings. Therefore it is possible that potential ambiguous and/or negated search patterns can be coded interactive.

The validity of the coding process can be controlled by interactive coding and/or by rapport files:

- file of the coded text units: all text units containing at least one search pattern of the category system is written to this output file. Category labels can be written behind each coded part of the text, this is useful for the validation of the coding process.
- file of uncoded text units: all text units that do not contain a search pattern of the category system are written to this output file.
- file of negated text units: all text units containing at least one search pattern of the category system where negation indicators before or after the search pattern occured in the specified distances are written to this output file.
- file of coded search patterns: each coded search pattern is written to the output file with external variables, code, text, start and end position (column) and category label. This file can become very large.
- file of overlapping text segments: text segments where at least one character is part of at least two search patterns. This causes problems with the vector file, not all codes can be displayed. The reasons maybe technical or caused by the category system.

9.1.6 Interactive coding

The screen shot shows the current text unit with the external variables and the red (bold) search pattern, at the bottom the search pattern, its code and category label are displayed. At the bottom the command buttons are on the left, the category system is on the right. The codes are clickable and can change the code, then you must press one of the following buttons to code the search pattern:

- yes – search pattern is coded with the selected code.
- yes+stop – search pattern is coded, after the last search pattern the results are written to the output files, and the coding will be terminated.
- no – search pattern is not coded.
- no+stop search pattern is not coded, after the last search pattern the results are written to the output files, and the coding will be terminated.

Interactive coding can last a long time. Therefore it is possible to terminate the coding and continue later. After the appropriate command was issued (+stop), the remaining search patterns are coded and the results written to the output files. After a restart the coding is continued where it was stopped, the results are appended to the appropriate files. Another termination is possible.

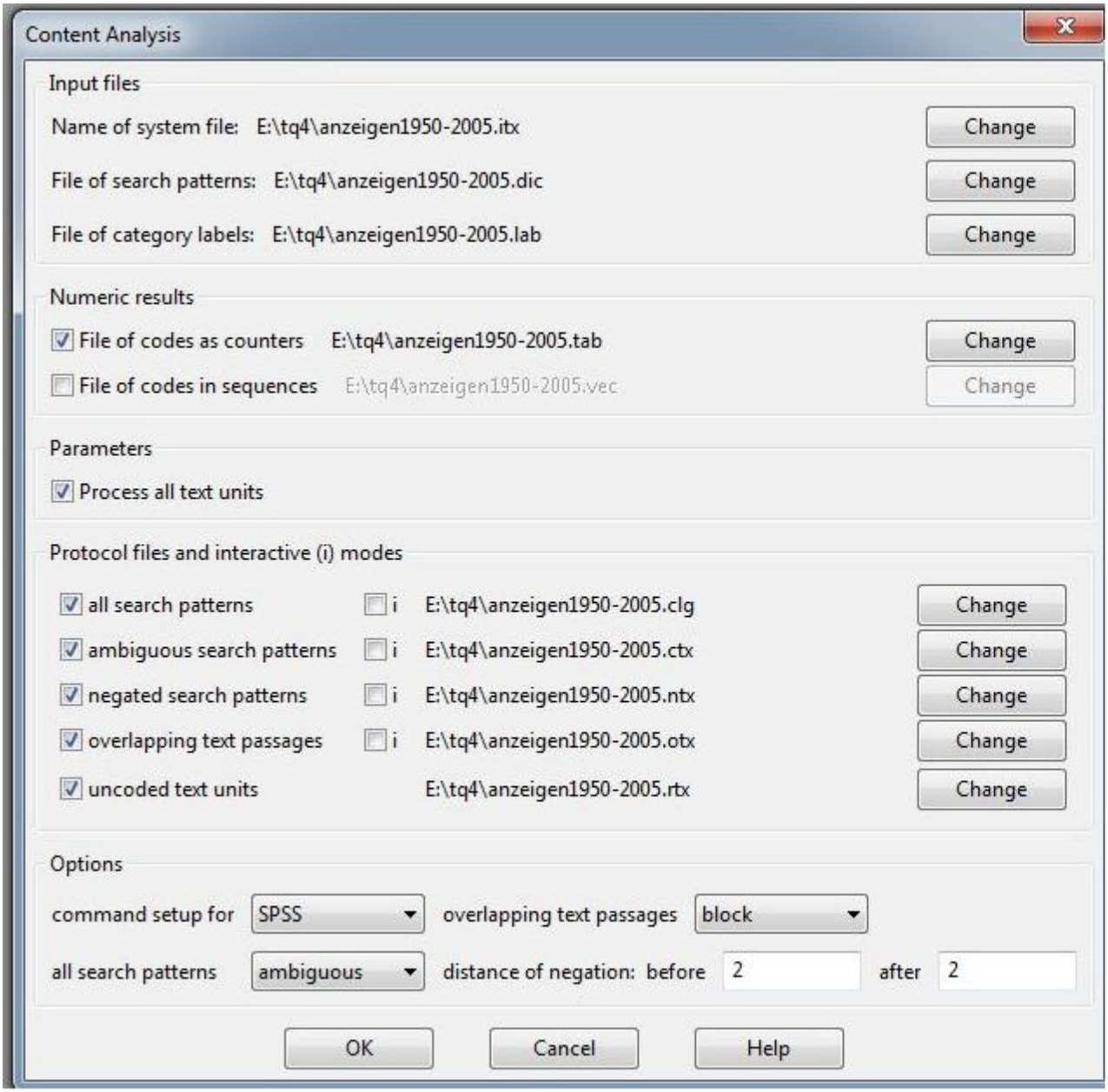
The coding suggestion does not consider negation. Also an extension of the category system with new codes is **not** possible.

Considering uncoded and not as suggested as orignally intended from the category system coded search patterns a coeffient (Interactive coding reliability coeffient) is computed:

$$ICRC = \frac{coded\ search\ patterns - rejected\ search\ patterns - changed\ search\ patterns}{coded\ search\ patterns + rejected\ search\ patterns}$$

The range is between 0 and 1. The higher it is, the better the reliability is.

The following picture shows the parameters:



name of system file: the name of the file where the system file is stored. Accept the generated file name or press the change button to open the file dialog.

process all text units If you check this box, the complete text will be processed, otherwise the defined sample will be processed (see chapter 4 on page 37).

file of search patterns: the name of the file where the search patterns are stored (DIC-file). The number of search patterns is discussed in chapter 5 on page 39. Accept the generated file name or press the change button to open the file dialog.

file of category labels: this file contains the category system with codes and their labels. Accept the generated file name or press the change button to open the file dialog.

file of codes as counters: the name of the file where the counters of the categories are stored. Accept the generated file name or press the change button to open the file dialog. If you want to process the file with SimStat, you must change the file extension to CSV.

file of codes in their sequence: the name of the file where the codes in their sequence are stored. Accept the generated file name or press the change button to open the file dialog. If you want to process the file with SimStat, you must change the file extension to CSV.

number of codes within a text unit: If a file of codes in their sequences is requested, this number specifies how many search patterns are coded within an text unit. The value is dependent on the longest text unit. If this number is exceeded, a warning is given. Coding continues without storing in this file. The statistics concerning the found search patterns are correct although the VEC-file is incorrect. Accept the generated file name or press the change button to open the file dialog.

setup for: A setup for further processing of the raw data matrix with statistical packages is generated for SPSS, SAS, or ConClus. It contains the reading specifications (data list), the labels for the codes (var labels) and the commands for frequency tables. The file is named `*.sps` for SPSS, `*.sas` for SAS, and `*.stk` for ConClus, where * stands for the project name.

coding parameters For each type of search patterns coding parameters can be specified:

- I: yes: these search patterns are coded interactive; no: automatic coding.
- protocol files: protocol files for these search patterns are requested.
- options: here parameters for the types of search patterns can be defined.
  - all search patterns: unique or ambiguous.
    - unique: the ambiguity of all search patterns is treated as specified in the parameter field. If a protocol file for all search patterns was requested, only the text units containing at least one potential ambiguous search patterns are written to the output file. If interactive coding is enabled, only the marked search patterns are coded interactive, all other are coded automatic.
    - ambiguous: if interactive coding is enabled, all search patterns are coded interactive, useful for teaching purposes or pretests. This feature smallens the difference between convential and computer aided content analysis. If a protocol file was requested, all text units that contains at least one search pattern are written to this file.
  - ambiguous search patterns; with or without labels.
    - with labels: labels are useful for the coding control of all or of the ambiguous search patterns, after them code and category label follow.
    - without labels: if the file is used for further processing (e.g. generate a word list) category labels disturb.
  - negated search patterns: distance of negation. Two values can be specified: the first one specifies the number of strings before the search pattern is searched for negation indicators, the second one specifies the number of strings after the search pattern is searched for negation indicators. The negation indicators are counted. If the number is odd, a negation exits, even numbers indicate a double negation (litotes). 0 means to disable negation control for all search patterns.

- overlapping search patterns: modes for their treatment
  - block: the first search pattern in the category system is used for coding.
  - overwrite: the last search pattern in the category system is used for coding.
  - longest: the longest search pattern in the category system is used for coding.

9.1.7 Information messages

```
TextQuest (tm) Text Analysis Software 03.03.2007 18:55
         program: SUWACO
     application: content analysis
input file        D:\texts\CONTAKT.itx
category file     D:\texts\CONTAKT.dic
tab file          D:\texts\CONTAKT.tab
vector file       D:\texts\CONTAKT.vec
CODED file        D:\texts\CONTAKT.ctx
REST file         D:\texts\CONTAKT.rtx
NEG file          D:\texts\CONTAKT.ntx
label file        D:\texts\CONTAKT.lab
Job for SPSS D:\texts\CONTAKT.sps

- C 01:      1363 search patterns processed
- C 02:        38 different categories
- C 06:      1363 with option C marked search patterns
- C 07:      1148 with option U marked search patterns
- C 09:         8 with option D marked word root chains
- C 10:         8 with option F marked word root chains

- I 01:      6315 text units read
- I 03:     62513 words read
- I 11:     16341 coded text passages in TAB file
- I 12:     16341 coded text passages in VEC file
- I 13:      5518 coded text units (87,38 %) in CODED file
- I 14:       797 not coded text units (12,62 %) in REST file
- I 15:         0 negated text units in NEG file
- I 16:     16341 search patterns in CLOG file
- I 21:      6315 output records in VEC file
- I 22:      6315 output records in TAB file
- I 23:       884 characters in overlapping search patterns
- I 24:         3 overflows in vec file
- I 25:         0 text units with negation(s)
SUWACO start:  18:55:15
SUWACO end:    18:55:25
SUWACO needed 10 seconds CPU time
```

In a content analysis every search pattern from a category system stored in a `*.dic`-file is searched in the system file. Case folding can be ignored (umlauts and characters with diacritics are treated correct) or not. Single and multiple negations in front of the search pattern are recognised. Also an interactive coding of potential ambiguous search patterns including several rapport files are possible.

9.1.8 Coded text units

This is a rapport file that allows to check the validity of the coding. It is a **T**ext**Q**uest system file that can be used for other analyses, e.g. a word list. It may contain category numbers and their labels for the checking of the validity of the coding process. Without categories and their labels this file can be regarded as the result of a filtering process.

9.1.9 Uncoded text units

This is a rapport file that allows to check the validity of the coding. It is a **T**ext**Q**uest system file that can be used for other analyses, e.g. a word list that allows the inspection of all words that are currently not used as search patterns for a content analysis category system. This is useful for open ended questions to see which responses stil have to be coded.

9.1.10 Negated text units

This is a rapport file that allows to check the validity of the negation algorithm and its coding. It is a **T**ext**Q**uest system file that is useful to test the number of words before and after the search pattern that are searched for negation indicators, and also these indicators themselves.

9.1.11 Coding control

This is a rapport file that allows to check the validity of the coding prcoess. It shows the search pattern, its code, its label, and its context.

9.2 Readability analysis

The readability analysis computes many different formulas that are based on syntactic criteria. Implications of the most formulas are that they are language and/or text genre specific, so the results have to be interpreted carefully. In opposite to the literature mentioned in the footnote, **T**ext**Q**uest doesn't work with a sample of 100 words, but with the whole text or parts of it (see chapter 4 on page 37). Dependent on that, raw values for the whole text and standardised values are computed. The results can be reading age, reading grade, or values of a certain range, often standardised between 0 and 100. See the notes for each formula of the implications are met by your text, otherwise you get invalid results.

The text unit must be the sentence, all formulas need the number of sentences.

9.2.1 Special word lists

Some readability formula require the counting of words, e.g. the vocabulary of 10year old children, foreign words, prepositions, or conjunctions. All these files are text files (extension `*.def`) and must be stored in the same directory where the system file of the current project is stored. These files can be adopted to one's own purposes. **T**ext**Q**uest comes with the following word lists:

file name	formula	language
dalechal.def	Dale-Chall 1948	English
dale.def	Dale 1983	English
spache.def	Spache 1	English
fworte.def	Kuntzsch's TRI	German
bamvan.def	Bamberger/Vanecek's 1000 most used words of 10 year old children	German
praepos.def	Dickes/Steiwer prepositions	German
konjunkt.def	Tränkle/Bailer conjunctions	German
spauld.def	Spaulding 1958	Spanish

For the TRI index it is possible to specify strings as indicators for foreign words. The regulations are the same described for inclusion lists (see example file fworte.def). The number of indicators for foreign words is dependent on the available memory (RAM). An indicator must not be longer than 80 characters.

The word lists of Bamberger and Vanecek were developed for the Vienna fiction and non-fiction formulas and consist of the 1000 most used words of 10 year old children. The list contains more words because some words are also included in new German orthography. Another reason is that all words and their flexions and cases were added (new in version 4.1):

- nouns: all singular and plural forms in all four grammatical cases (nominative, genetive, dative, accusative)
- verbs: all forms of present and past tense, male/female forms for all four grammatical cases, participle form

- adjective: in all four grammatical cases

All other word lists contain whole words that are searched in the text and counted. Validity features are not implemented (yet).

The file `refo.def` contains the patterns for the counting syllable algorithm, `refod.def` is for German, `refoe.def` for English. With the file `refo.def` the algorithm for counting the syllables is controlled, and it can be adapted to other languages quite easily. The following regulations must be considered:

- The longest patterns must occur at the beginning of the file.
- Up to 200 patterns are allowed.
- The maximum length of a pattern is 4 characters.
- Only capital letters are allowed in diphtongs, umlauts must also be lowercase and uppercase.
- Within the patterns only ? (representing any character) as a wild card may be used (important for the English language). The * (asterisk) as a wild card character is not allowed.

In general these patterns are an enumeration of the diphtongs and vowels of a language. In languages with big differences between spoken and written language (e.g. English), whole syllables must be entered. The provided `refo*.def` files show how that is done for German and English. The algorithm of syllable counting can be controlled by a protocol file that contains the number of syllables and the string.

Another function is the control of style. The journal file (`*.jou`) contains all sentences that are too long, have too many backets, or are too complex. Also too long words or too many foreign words are written to this file. The number of items can be specified, and with the journal file an inspection of the source file is easy.

9.2.2 Definitions

- syllables: number of syllables in the text. The rules for spoken language are valid for counting the syllables (one chin move = one syllable). Numbers are counted like they were spoken, e.g. 10 is one syllable, 21 is three syllables. Words without vowels are counted as monosyllables (words with one syllable), numbers are an exception. Depending on the language of the text, numbers are counted differently. **T**ext**Q**uest tries to recognise the language of the text for this purpose. Currently counting syllables of numbers works for English and German with a low error rate.
- strings: number of strings in the text, a string consists of all characters between two blanks or another delimeter (e.g. end of line or beginning of line). **T**ext**Q**uest separates some characters from the words, e.g. punctuation marks or brackets. These characters are specified within the generation of a system file and can be changed in the appropriate menu.

  The classification of the strings is done as words, numbers, and other. Words start with a letter, numbers with a numeral, and other are the rest category.

- sentences: number of sentences in the text.
- monosyllables: number of words with one syllable.
- 2syllables: number of words with one or two syllables.
- 3syllables: number of words with at least 3 syllables.
- characters: number of characters in the text, all characters count, not only letters.
- punctuation marks: number of punctuation marks in the text. punctuation marks are ,.;:!?
- rare words: words in the text not contained in a word list. The word lists contain well known word which are counted. This number is subtracted from the total number of words, the result is the number of rare words.
- 3charwords: number of words with 3 and more characters.
- 6charwords: number of words with 6 and more characters.
- 7charwords: number of words with 7 and more characters.

The English word lists are converted into uppercase internally before the comparison. Only whole words are counted, not parts (substrings) of it. This procedure is not possible for German, because there are differences in word meaning due to differences in upper-/lower case (e.g. Würde and würde).

The formulas can be found in the literature mentioned for each formula in the bibliography of this manual. **T**ext**Q**uest uses the formulas give in the original literature, because other authors did not copy the formula correctly or even worse, wrote the sample size into the formula.

9.2.3 Language independent formulas from Tuldava

Tuldava's suggestion for a language idependent formulas is based on sentence length multiplied by the logarithm of word length, where the sentence length can be measured in words per sentence, characters per sentence or syllables per sentence. The word length can be measured in characters or syllables per word. The implementation of **T**ext**Q**uest combined these suggestions in four different formulas:

1. tuldava1 is based on words and characters
2. tuldava2 is based on characters
3. tuldava3 is based on syllables
4. tuldava4 is based on words and syllables

9.3 English

Many formulas calculate the reading grade, the following table shows the reading grades and their corresponding age groups both for the USA and the UK. Reading class and reading grade mean the same.

age	US grade	UK grade
3-4	pre school	Nursery School
5-6	kindergarten	1
6-7	1	2
7-8	2	3
8-9	3	4
9-10	4	5
10-11	5	6
11-12	6	7
12-13	7	8
13-14	8	9
14-15	9	10
15-16	10	11
16-17	11	12
17-18	12	13
18-22	college	university

The formulas are ordered by language, and then by the name of the authors, whereas the results in **T**ext**Q**uest are ordered by language and the type of result:

- index value, mostly for the formula of Rudolf Flesch and derivates, often between 0 and 100
- reading grade or reading class
- reading age

At first some of the most popular readability formulas are introduced, because these were often recalculated or served as a basis for new formulas. However, there are some problems using readability formulas. At first some variables are difficult to count, e.g. syllables. Also splitting text into grammatical sentences are difficult, because characters that end a sentence like .? or ! maybe ambiguous, e.g. a . can mean the end of a sentence but also a decimal point or within an abbrevation. Some formulas also require a special kind of counting words, syllables or other variables, these are currently not implemented.

9.3.1 Flesch 1948: Reading Ease Index REI

The Flesh Reading Ease Formula was developed by Rudolf Flesch in 1948 and it is based on school texts covering grades three to twelve. The index is usually between 0 (hard) and 100 (easy). This orientation contrasts with some of the other readability measurements since higher scores mean easier reading. This test is often used to assess adult reading materials; in fact it is used by some United States government agencies and the United States Department of Defence as an indicator of readability. There are two formulas citied, but only one fits published

genre of text: US school texts covering grade 3 to 12, prose

sample size: 100 words

result: standardised value between 0 (difficult) and 100 (easy)

standardisation: yes

The following table serves as an interpretation aid for Flesch's REI and all derivates the three derived formulas from Powers, Sumner, Kearl as well as Farr, Jenkins, Paterson and Kincaid, Fishburne, Rogers, Chissom.

90-100	5. class
80-90	6. class
70-80	7. class
60-70	8. und 9. class
50-60	10.-12. class (high school)
30-50	college
0-30	college graduate

9.3.2 Dale-Chall 1948: DC

This formula was revised several times, because the underlying tests (McCall-Crabbs lessons from 1925) changed in 1951 and 1960, in 1958 by Powers, Sumner, and Kearl and 1968 by Holmquist.

The Dale-Chall formulas work with two words lists, one with 2946 and another one with 920 words (Bamberger/Vanecek 1984, p. 56). **T**ext**Q**uest uses the longer list with 2946 words. The result is the reading grade of a reader who can comprehend a text at 3rd grade or below. The adjustment if more than 5 % of difficult words occur in the text is implemented. However, known words like proper names, grammatical forms like 3rd person singular of verbs, plural of nouns, progressive forms of verbs and the like are currently not implemented.

genre of text: children between 5 and 10 years of age

sample size: 100 words

result: reading grade

standardisation: no

DC	class	age
0 - < 5.0	4	5-10
5.0 - < 6.0	5-6	10-12
6.0 - < 7.0	7-8	12-14
7.0 - < 8.0	9-10	14-16
8.0 - < 9.0	11-12	16-18
9.0 - < 10.0	13-15	18-21
>= 10.0	15 (college)	22

The second last line age value was changed from 18-20 to 18-21 because otherwise 21 year old persons would be excluded.

9.3.3 McElroy 1950: Fog count

John McElroy, author of Techniques For Clear Informative Writing (1950), developed the Fog Count (FC) to measure reading ease. In a single-sentence sample, Easy words of one or two syllables are counted once and hard words of three or more syllables are counted thrice.

The formula does not treat all polysyllables as hard words. Names of persons, places, months and days are considered to be easy. Michelangelo, Mesopotamia, September and Wednesday, are each counted once. Abbreviations (e.g. UNESCO, UNICEF) or numbers (e.g. 3.1417 or 50,000,000) are also treated as easy words. Moreover, compound names of persons with common titles are treated as single names. So, President Barack Obama gets a count of only one. All these exception are currently not implemented.

Linsear write is another name for Fog count.

genre of text: general

sample size: 1 sentence

result: 25 is the average

standardisation: yes

McElroy's Fog Count may be converted into a grade level by dividing by 2.

9.3.4 Andersson 1983: RIX

The Rix index (Anderson 1983, 1994) owes its origins to a readability index developed by Björnsson (1968) in Sweden and called Lix. Anderson reports that he discovered the Rix index when he was based at the University of Stockholm in Sweden; he translated it and brought it back to Australia. Students at Flinders University applied it to French, German, Greek and English texts using the Swedish norms. When these applications were reported at a conference (Anderson 1981), reading teachers wanted English norms. The result was a new index which was even quicker than Lix to use. To acknowledge its origins, the new index was named Rix.

Rix has been applied in remedial and special education settings (Anderson 1986), in adult literacy programs (Malmquist 1985), and across a range of professional areas (e.g. law, commerce, medicine) to assess comprehensibility of printed texts and publications.

Like Lix, Rix is based on two factors, a word factor based on the length of words, and a sentence factor based on sentence length. These two factors appear in virtually every readability formula produced over the past 70 years. However, Rix measures the two factors differently from other formulae and weights them differently. Validity studies demonstrate that Rix gives very similar results to perhaps the most widely used measures, Flesch and Fry, but is simpler to use and may be applied over a greater range of texts. Full details may be found in Anderson (1983).

genre of text: general

sample size: 10 samples with 10 sentences each

result: reading age

standardisation: no

Rix	grade
7.2	college
6.2	12
5.3	11
4.5	10
3.7	9
3.0	8
2.4	7
1.8	6
1.3	5
0.8	4
0.5	3
0.2	2
0-0.2	1

9.3.5 Björnsson 1968: LIX

genre of text: general

sample size: 100 words

result: reading age

standardisation: no

Lix	grade
56 +	college
52-55	12
48-51	11
44-47	10
40-43	9
36-39	8
32-35	7
28-31	6
24-27	5
20-23	4
15-19	3
10-14	2
-10	1

9.3.5.1 Bormuth 1981: B-G

The Bormuth Readability Index outputs a number that correlates to a U.S. grade level. For example, a result of 10.6 means students in 10th grade and above can read and comprehend the text. Unlike the new Dale-Chall Readability Formula which outputs an adjusted number that you must match to a number on an adjusted grade level table, the Bormuth Readability Index does not require you to use a table to determine an adjusted grade level.

genre of text: academic documents and school textbooks

result: grade

readership: above 4th grade

9.3.6 Caylor, Stich, Ford: Forcast 1973

The FORCAST Readability Formula is the result of The Human Resources Research Organization of Alexandria, Virginia, to study the reading requirements of military occupational specialties in the US Army. John S Caylor, Thomas G Sticht, and J Patrick Ford were assigned this responsibility in 1973.

The subjects for the research were Vietnam draftees entering basic training and job-specific training. The FORCAST Readability Formula evolved from their study. The formula was first published in 1973 as an article in a journal called Literacy Discussion, published by UNESCOÆs International Institute for Adult Literacy.

The FORCAST Readability Formula is the only test not designed for running narrative. Therefore, it is considered perfect for multiple-choice quiz contests, applications, entrance forms, and so on. The FORCAST Readability Formula is strictly not prescribed for assessing primary age reading materials. It was tested against a comprehension level of 35 % only.

genre of text: technical manuals, notes, multiple-choice-questions

sample size: 150 words

result: reading grade, 5-12 class

standardisation: yes

9.3.7 FC-A: Forcast 1973

genre of text: technical manuals, notes, multiple-choice-questions

sample size: 150 words

result: reading age. 5-12 class

standardisation: yes

9.3.8 Coleman 1965: CM1

genre of text: general

sample size: 100 words

result: forecast on the percentage basis of the right answers using a cloze test

standardisation: yes

9.3.9 Coleman 1965: CM2

genre of text: general

sample size: 100 words

result: forecast on the percentage basis of the right answers using a cloze test

standardisation: yes

9.3.10 Coleman-Liau 1975: CL-I

Percentage of the correct answers of a college undergraduate. 36 text passages with 150 words each with gap test with 5 words.

genre of text: textbook for the public school system of the USA

sample size: 100 words

result: cloze

standardisation: yes

cloze value	grade
> 80.5	1
76.9 - <=80.5	2
73.2 - <=76.9	3
69.6 - <=73.2	4
65.9 - <=69.6	5
62.3 - <=65.9	6
58.6 - <=62.3	7
55.0 - <=58.6	8
51.3 - <=55.0	9
47.7 - <=51.3	10
44.0 - <=47.7	11
40.4 - <=44.0	12
36.7 - <=40.4	13
33.1 - <=36.7	14
29.4 - <=33.1	15
25.8 - <=29.4	16
<25.8	17+

9.3.11 Coleman-Liau 1975: CL-G

genre of text: textbook for the public school system of the USA

sample size: 100 words

result: reading grade derived from the table above

standardisation: yes

9.3.12 Dale-Chall 1995: DC2

genre of text: children between 5 and 10 years of age

sample size: 100 words

result: cloze percentage

standardisation: no

cloze	grade
>= 58	1
57-54	2
53-50	3
49-45	4
44-40	5-6
39-34	7-8
33-28	9-10
27-32	11-12
21-16	13-15
15-0	16+

9.3.13 DB1: Danielson/Bryan 1963

The formulas were derived from an analysis of 383 McCall-Crabs Standard Test Lessons in Reading (from 1950).

genre of text: general

sample size: 100 words

result: index value

standardisation: no

9.3.14 DB2: Danielson/Bryan 1963

genre of text: general

sample size: 100 words

result: reading grade ?? standardised value between 0 (difficult) and 100 (easy)

standardisation: no

90-100	very easy, class level 3
80-89	relatively easy, class level 4
70-79	easy, class level 5
60-69	standard, class level 6
50-59	medium, junior high school level
30-49	difficult, high school level
0-29	very difficult, college level

9.3.15 Farr, Jenkins, Paterson 1951: recalculation of Flesch's Reading Ease Index REI

genre of text: general, new REI

sample size: 100 words

result: standardised value between 0 (difficult) and 100 (easy)

standardisation: yes

9.3.16 FK-G: Flesch-Kincaid 1953

The US Government Department of Defense uses Flesch-Kincaid Grade Level formula as a standard test.

genre of text: prose

sample size: 100 words

result: reading grade

standardisation: no

9.3.17 FK-A: Flesch-Kincaid 1953

genre of text: prose

sample size: 100 words

result: reading age

standardisation: no

9.3.18 Fry 1968

$FRY = \frac{words}{sentences} \qquad \frac{syllables}{sentences}$

You need Fry's graphic for the interpretation of the results. The sample size is 100 words.

9.3.19 Gunning 1952: Gunning's FOG

genre of text: general

sample size: ca. 100 words, whole sentences

result: reading grade

standardisation: no

FOG	Grad
6	6. class
7	7. class
10	8. class
11	high school
12	college
13	freshman
16	B.A. level
18	Dr. Level, insurance texts

9.3.20 Kincaid, Fishburne, Rogers, Chissom 1975 - recalculated ARI

Kincaid, Fishburne, Rogers, Chissom recalculated several readability formulas that are described here. The authors work for the US Navy and recalculated 5 formulas especially for technial texts of the US Navy.

genre of text: technical texts

sample size: ca. 100 words, whole sentences

result: reading grade

standardisation: no

9.3.21 Kincaid, Fishburne, Rogers, Chissom 1975 - recalculated FOG count

genre of text: technical texts

sample size: ca. 100 words, whole sentences

result: reading grade

standardisation: no

9.3.22 Kincaid, Fishburne, Rogers, Chissom 1975 - recalculated Flesch REI

genre of text: technical texts

sample size: ca. 100 words, whole sentences

result: reading grade

standardisation: no

9.3.23 Kincaid, Fishburne, Rogers, Chissom 1975 - recalculated Farr, Jenkins, Paterson

genre of text: technical texts

sample size: ca. 100 words, whole sentences

result: reading grade

standardisation: no

9.3.24 Kincaid, Fishburne, Rogers, Chissom 1975 - recalculated Forcast

genre of text: technical texts

sample size: ca. 100 words, whole sentences

result: reading grade

standardisation: no

9.3.25 McAlpine 1997: EFLAW

genre of text: general, English as a foreign language

sample size: 100 words

result: difficulty level table

standardisation: no

The Microsoft vesion uses another table with 20.49, 25.49 and 29.49 as boundaries.

EFlaw	grade
1− <= 20	easy
20− <= 25	quite easy
25− <= 30	a little difficult
> 30	confusing

9.3.26 McLaughlin 1969: SMOG-G

SMOG (Simplied Measure Of Gobbledygook) values are based on samples of 30 sentences, best choice is to take 10 sentences from the beginning, 10 sentences from the mniddle, and another 10 sentences from the end of the text. The criterion is not the 50 or 75 percentage value for understanding, but the complete understanding.

The standard error of the estimated grade level is 1.5159 grades, comparable to that of other readability formulae. You may have seen SMOG conversion tables compiled by one Harold C. McGraw. They are slightly inaccurate because they are based on the approximate formula. Furthermore tables for texts of fewer than 30 sentences are statistically invalid, because the formula was normed on 30-sentence samples.

genre of text: general

sample size: 30 sentences

result: grade

standardisation: yes

9.3.27 SMOG-A: McLaughlin 1969

genre of text: general

sample size: 30 sentences

result: reading age

standardisation: yes

9.3.28 SMOG-G: McLaughlin 1969

genre of text: general

sample size: 30 sentences

result: grade level

standardisation: yes

Harold C. McGraw developed a re-calculation table:

3syl	class
1 -2	4
3-6	5
7-12	6
13-20	7
21-30	8
31-42	9
43-56	10
57-72	11
73-90	12
91-110	13
111-132	14
133-156	15
157-182	16
183-210	17
211-240	18

9.3.29 Powers, Sumner, Kearl 1958: recalculation of Dale-Chall

The Powers-Sumner-Kearl Readability Formula is one of the best formulas to calculate the US grade level of a text sample based on sentence length and number of syllables. This formula is suited for primary age children (age 7-10) and, usually, is not considered ideal for children above the age of 10 years.

genre of text: children between 5 and 10 years of age

sample size: 100 words

result: grade

standardisation: yes

9.3.30 Powers, Sumner, Kearl 1958: recalculation of Gunning's Fog

genre of text: general

sample size: ca. 100 words, whole sentences

result: reading age

standardisation: no

9.3.31 Powers, Sumner, Kearl 1958: recalculation of Flesch's REI

genre of text: general, modified Flesch-formula

sample size: 100 words

result: class level for 7 to 10 years old

standardisation: yes

9.3.32 Powers, Sumner, Kearl 1958: recalculation of Flesch's REI

genre of text: general

sample size: 100 words

result: reading age for 7 to 10 years old

standardisation: yes

9.3.33 Powers, Sumner, Kearl 1958: recalculation of Farr-Jenkins-Paterson's Modified new reading ease index

genre of text: general, modified new REI

sample size: 100 words

result: standardised value between 0 (difficult) and 100 (easy)

standardisation: no

9.3.34 Smith/Senter 1967: ARI

The Automated Readability Index (ARI) is an approximate representation of the U.S. grade level needed to comprehend the text. It relies on characters per word instead of syllables per word and has the advantage this measurement that couting characters is easier than counting syllables. The ARI-value is typically higher than the Kincaid and Coleman-Liau measures, but lower than the Flesch.

genre of text: technical texts of the US forces, here US Army

sample size: 20 pages of 24 books each, no non-fictional texts

result: US reading grade

standardisation: no

9.3.35 Smith/Senter 1970: ARI

genre of text: technical texts of the US Airforce

sample size: 20 pages of 24 books each, no non-fictional texts up to seventh grade

result: index

standardisation: no

9.3.36 Solomon 2006: Direct Dale-Chall Grading (DDCG)

formula: $DDCG = 0.3 * \frac{rare\ words * 100}{words} + 0.1 * \frac{words}{sentences} + 3$

genre of text: 1.-4. class ??

sample size: 100 words

result: reading grade

standardisation: no

9.3.37 Solomon 2006: Stain index

formula: $SI = \frac{syllables}{10} * \frac{sentences}{3}$

genre of text: general

sample size: 3 sentences

result: index

standardisation: yes

9.3.38 Spache 1953

Both Spache-formulas take the vocabulary into account, like the Dale-Chall formula. The Spache wordlist contains 1040 words.

genre of text: 1.-4. class

sample size: 100 words

result: reading grade

standardisation: no

9.3.39 Spache 1978

genre of text: 1.-4. class

sample size: 100 words

result: reading grade

standardisation: no

9.3.40 WSI: Wheeler-Smith 1954

genre of text: general

sample size: 100 words

result: reading grade

standardisation: no

WSI	grade
26.6-34.5	5
19.1-26.5	4
11.6-19.0	3
8.1-11.5	2
4.0- 8.0	1

9.3.41 German

9.3.41.1 AVI: Amstad 1978

text genre: general, based on Flesch's formula

result: standardised value between 0 (difficult) and 100 (easy)

9.3.41.2 DS: Dickes-Steiwer 1977

This formula is the most simple formula, the computer formula contains much more criteria like number of prepositions, conjunctions, personal pronouns for third person and first and second person. The complete formula contains criteria like names, living verbs, reflexive verbs, and concrete nouns.

text genre: fiction for 13 year old students

result: estimation for cloze, Flesch derivate

9.3.41.3 FDK (Fasse dich kurz): Schirm 1971

genre of text: unknown

sample size: unknown

result: index between 0 (easy) and over 50 (difficult)

standardisation: no

FDK	style
up to 10	short, e.g. telegram, notices
11 - 25	modern: short, precise
26 - 50	diffuse, prolix
over 50	unclear, inflated

9.3.41.4 Fucks 1955

text genre: general

result: unknown, result not usable. Square root results in reading grade.

9.3.41.5 G-LIX: LIX for German Bamberger, Vanecek 1984: 62

text genre: books for juveniles

result: range: 15 (very easy) - 80 (very difficult)

9.3.41.6 G-RIX: RIX for German Bamberger, Vanecek 1984: 64

text genre: books for juveniles

result: reading age

9.3.41.7 G-SMOG: SMOG for German: Bamberger, Vanecek 1984

text genre: general

result: reading grade

9.3.41.8 G-WSI: WSI for German Bamberger, Vanecek 1984

text genre: general

result: reading grade

9.3.41.9 QU: Bamberger, Vanecek 1984

text genre: general

result: reading grade

9.3.41.10 TB1: readability index Tränkle, Bailer 1984

text genre: general

result: optimisation of Dickes-Steiwer, Flesch derivate

9.3.41.11 TB2: readability index Tränkle, Bailer 1984

text genre: general

result: optimisation of Dickes-Steiwer, Flesch derivate

9.3.41.12 TRI: Text-Redundanz-Index Kuntzsch 1981

text genre: political comments in newspapers

result: index between 0 (very difficult) and 100 (very easy)

9.3.41.13 WSTF1: 1. Wiener Sachtextformel: Bamberger, Vanecek 1984

text genre: non fiction texts

result: reading age

9.3.41.14 WSTF2: 2. Wiener Sachtextformel: Bamberger, Vanecek 1984

text genre: non fictional children's books

result: reading age

9.3.41.15 WSTF3: 3. Wiener Sachtextformel: Bamberger, Vanecek 1984

text genre: non fictional children's books

result: reading age

9.3.41.16 WSTF4: 4. Wiener Sachtextformel: Bamberger, Vanecek 1984

text genre: non fictional children's books

result: reading age

9.3.41.17 WLTF1: 1. Wiener Literaturtextformel: Bamberger, Vanecek 1984

text genre: fictional children's books

result: reading age

9.3.41.18 WLTF2: 2. Wiener Literaturtextformel: Bamberger, Vanecek 1984

text genre: fictional children's books

result: reading age

9.3.41.19 WLTF3: 3. Wiener Literaturtextformel: Bamberger, Vanecek 1984

text genre: fictional children's books

result: reading age

9.3.42 Spanish

9.3.42.1 CSRI: Childrens Spanish Reading Index Crawford 1984

text genre: children's books for primary age

result: reading age, 1. - 6. reading grade

9.3.42.2 Huerta: Huerta 1959

text genre: general

result: Flesch derivate

9.3.42.3 Gutierrez: Gutierrez 1972

text genre: general

result: only for 6. grade, percentage of right answer using cloze

9.3.42.4 SMOG-S: Contreras a.o. 1999

The authors offer a formula based on the SMOG value for Spanish texts. They use the precise version of the SMOG (with decimals).

text genre: general

results: SMOG for Spanish

9.3.42.5 Spaulding: Spaulding 1958

Textgenre: general

results: Index from 20 to 200

There are some additional rules that include the extension of the original word list. Numbers are counted as monosyllables in general.

Spaulding	meaning
0 - 40	texts for primers
40 - 60	very easy
60 - 80	easy
80 - 100	somehow difficult
100 - 120	difficult
120 - 200	extraordinary difficult

9.3.42.6 IFSZ: Flesch-Szigriszt 1993

The authors developed a version of Flesch's REI for Spanish texts.

text genre: general

results: index and grade

value	meaning
0 - 40	very difficult
40 - 55	somewhat difficult
55 - 65	normal
65 - 80	quite easy

9.3.43 Danish

9.3.43.1 DK-LIX: Jakobsen 1971

text genre: general

result: reading age

9.3.44 Dutch/flamish

:

9.3.44.1 Brouwer: Brouwer 1963

This formula was developed on the basis of 25 children's books.

text genre: children's books

result: reading age

9.3.44.2 Dourma: Dourma 1960

text genre: general

result: reading age

9.3.44.3 Staphorsius: Staphorsius und Krom 1985

The authors developed several formulas, the following one is the one for computers.

text genre: non fictional texts for 3.-6. class

result: reading grade: 3-6

9.3.45 French

9.3.45.1 KM: Kandel and Moles 1958

text genre: general, Flesch derivate

result: reading age

9.3.45.2 SMOG-F: Contreras a.o. 1999

The authors offer a formula based on the SMOG value for French texts. They use the precise version of the SMOG (with decimals).

text genre: general

result: SMOG for French

9.3.46 Swedish

9.3.46.1 S-LIX: Lix for Swedish, Björnsson 1968, 1983

text genre: general

result: value range: 20 (simple) up to 60 (difficult)

Another function is the control of style. In a journal file (*.jou) all sentences that are too long, have too many backets, or are too complex. Also too long words or too many foreign words are written to this file. The number of items can be specified, and with the journal file an inspection of the source file is easy.

9.3.47 Italian

There are two GULP (Gruppo Universitario Linguistico Pedagogico) formulas published, and another two based on Flesch's formula by Roberto Vacca.

9.3.47.1 GULPease: Flesch derivate for Italian, Tonelli, et al. 2012

There are two different formulas! Gruppo Universitario Linguistico Pedagogico. It is unknown what table is for what formula, the literature is a complete chaos.

genre of text: general

result: value range: 0 not understandable up to 100 difficult

GULPease	meaning
0 - 35	not understandable
35 - 50	very difficult
50 - 60	difficult
60 - 80	easy
> 80	very easy

9.3.47.2 GULPease2: Flesch derivate for Italian

genre of text: general

result: value range: 0 not understandable up to 100 difficult

GULPease2	meaning
0 - 40	not understandable
40 - 60	difficult
60 - 80	medium
80 - 100	easy

9.3.47.3 Vacca1972: Flesch derivate for Italian, Franchina-Vacca 1972

genre of text: general

result: value range: 0 very difficult to 100 very easy

9.3.47.4 Vacca1986: Flesch derivate for Italian, Franchina-Vacca 1986

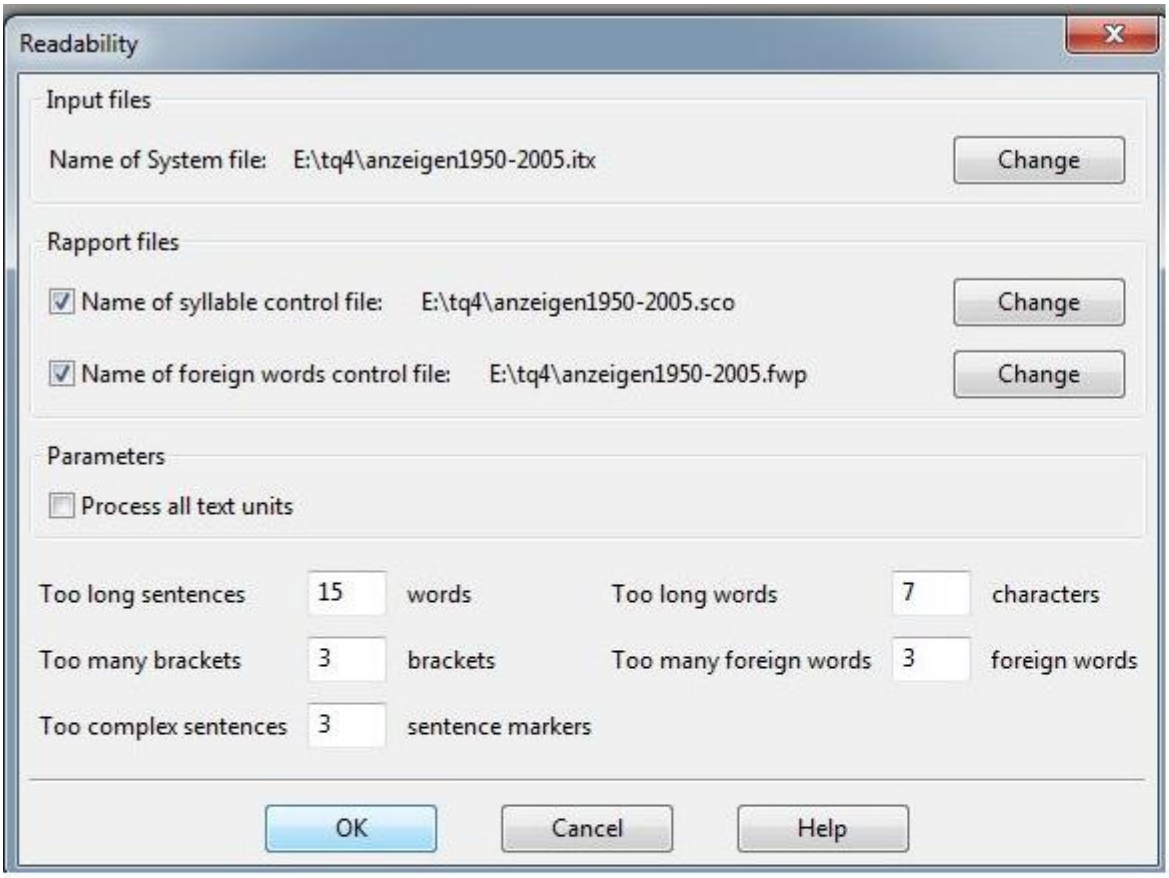
Warning: there are two versions of this formula, the coefficients were interchanged and result in invalid results.

genre of text: general

sample size: 100 words

result: value range: 0 very difficult to 100 very easy

9.3.48 Parameters of the program



name of system file: the name of the file where the system file is stored. The name may contain drive and/or directory specifications.

file of foreign words: For the calculation of TRI indicators for foreign words are counted, the indicators can be validated with this file, it contains the words being recognised as foreign words. The name may contain drive and/or directory specifications.

process all text units If you affirm this question, the complete text will be processed, otherwise the defined sample will be processed (see chapter 4 on page 37).

protocol file for syllable counting: enter a file name if you want to validate the syllable counting algorithm, otherwise leave it empty.

protocol file for foreign words: enter a file name if you want to validate the foreign words recognising, otherwise leave it empty.

too long sentences: sentences with more than the specified number of words are written to the journal file.

too many brackets: sentences with more than the specified number of brackets (round and braced brackets) are written to the journal file.

too complex sentences: sentences with more than the specified number of sentence markers (.;!?) are written to the journal file.

too long words: sentences with more than the specified number of words are written to the journal file.

too many foreign words: sentences with more than the specified number of foreign words are written to the journal file.

```
TextQuest (tm) Text Analysis Software 15.05.2007 14:41
         program: REFO
     application: Readability
input file        D:\texts\bush.itx
automatic language detection: English
file of foreign words control  D:\texts\bush.fwp
file of syllable count control D:\texts\bush.sco
- I 01:       493 text units read
- I 03:      4117 words read
- I 04:        12 numbers read
- I 05:       637 other read
- I 06:      4766 character strings read
- I 07:     15600 characters read
- I 08:       329 sentence structure markers read
- I 09:       303 sentence end markers read
- I 10:         2 brackets read
- I 11:      7200 syllables read
- I 12:      2205 (53,56 %) words with 1 syllable read
- I 13:      3137 (76,20 %) words with 2 and less syllables read
- I 14:       921 (22,37 %) words with 3 and more syllables read
- I 15:      2907 (70,61 %) words with 3 or less characters read
- I 16:      1040 (25,26 %) words with 6 and more characters read
- I 17:       682 (16,57 %) words with 7 or more characters read
- I 18:       141 ( 3,42 %) words from Bamvan word list read
- I 19:       105 ( 2,55 %) foreign words read
- I 20:      2808 (68,21 %) words from Dale-Chall word list read
- I 21:      2454 (59,61 %) words from Spache word list read
- I 22:      2717 (65,99 %) words from Dale word list read
- I 23:        90 ( 2,19 %) words from preposition list read
```

```
- I 24:         0 ( 0,00 %) words from conjunction list read
- I 25:       133 ( 3,23 %) words from Spaulding word list read
REFO start:   14:41:06
REFO end:     14:41:18
REFO needed  12 seconds CPU time

standardisation factor:    47,660

values of readability formulas

          raw    %  standard  formula
- R 01:  2806,000    21,232 Coleman 1
- R 02:  3249,490    31,027 Coleman 2
- R 03: -2673,417    82,771 Coleman-Liau: 99 1
- R 04:     3,238     3,238 Danielson-Bryan 1
- R 05:  3484,466    32,649 Farr-Jenkins-Paterson: New Reading Ease Index:  11
- R 06: -5894,177    69,217 Flesch's Reading Ease Index:   7
- R 07:     9,810     9,810 LIX:   1. class
- R 08:   326,149     5,423 Power's Modified Reading Ease Index:  12
- R 09:     6,328     9,263 Power's Modified New Reading Ease Index:  12
- R 10:     1,383     1,383 RIX:   5. class
- R 11:    18,682    18,682 Wheeler-Smith-Index:   3

reading level

- R 12:    -1,180    -1,180 Automated Reading Index
- R 13:     3,479     3,479 Coleman-Liau
- R 14:     4,181     4,181 Dale-Chall 1949: 4. class and below
- R 15:     4,184     4,184 Dale-Chall 1983: 4. class and below
- R 16:     3,893     3,893 New Dale-Chall
- R 17:   104,400   104,400 Danielson-Bryan 2: too high
- R 18:    15,564    15,564 McAlpine's EFLAW: very easy
- R 19:     6,007     6,007 Flesch-Kincaid
- R 20:     3,944     3,944 Gunning's FOG: very easy
- R 21:     5,817     5,817 New Gunning's FOG
- R 22:  -200,000    13,060 Forcast
- R 23:     4,984    -0,874 Linsear Write
- R 24:   326,159     5,433 Powers-Sumner-Kearl
- R 25:    33,348    10,486 SMOG
- R 26:    34,917     7,714 p-SMOG
- R 27:    33,267    10,682 SMOG2
- R 28:     2,619     2,619 Spache 1
- R 29:     2,227     2,227 Spache 2

reading age

- R 30:    39,126    39,126 Automated Reading Index
- R 31:    11,007    11,007 Flesch-Kincaid
- R 32:  -195,000    18,060 Forcast
- R 33:     9,667    14,604 FRY (words/sentence syllables/sentence)
- R 34:   330,642     9,915 Powers-Sumner-Kearl
- R 35:    38,348    15,486 SMOG
```

German

- R 36: 81,957 81,957 Amdahls Verstõndlichkeitsindex (AVI)
- R 37: 65,142 65,142 Dickes/Steiwer
- R 38: 68,704 68,704 Traenkle/Bailer 1
- R 39: 94,480 94,480 Traenkle/Bailer 2
- R 40: 2436,387 37,006 Textredundanz Index (TRI): 7.-8. class
- R 41: 31,643 5,625 Fucks
- R 42: 18,682 18,682 FDK
- R 43: 28,348 28,348 G-SMOG
- R 44: 18,682 5 G-WSI
- R 45: 9,810 1 G-LIX Prosa
- R 46: 9,810 3 G-LIX Sachtext
- R 47: 1,383 2 G-RIX Prosa
- R 48: 1,383 3 G-RIX Sachtext
- R 49: -0,916 5,486 Bamberger Qu
- R 50: 5,798 5,798 1. Wiener Sachtextformel (WSTF1)
- R 51: 5,722 5,722 2. Wiener Sachtextformel (WSTF2)
- R 52: 6,453 6,453 3. Wiener Sachtextformel (WSTF3)
- R 53: 253,596 6,176 4. Wiener Sachtextformel (WSTF4)
- R 54: -61,269 -61,269 1. Wiener Literaturtextformel (nWL1)
- R 55: -77,780 -77,780 2. Wiener Literaturtextformel (nWL2)
- R 56: 6,308 6,308 3. Wiener Literaturtextformel (nWL3)

Spanish

- R 57: 248,328 1,875 Crawford Spanish Reading Index (CSRI)
- R 58: -4616,020 105,647 Huerta
- R 59: 85,396 85,396 Gutierrez
- R 60: 46,814 46,814 Spaulding: very easy
- R 61: 6,664 15,582 SMOG

French

- R 62: -5102,012 86,000 Kandel-Moles
- R 63: 37,633 12,607 SMOG

Dutch

- R 64: -5346,150 81,525 Douma
- R 65: -4624,335 74,952 Brouwer
- R 66: 0,696 0,696 Staphorsius-Krom

Danish

- R 67: 9,810 9,810 LIX

Swedish

- R 68: 9,810 9,810 LIX

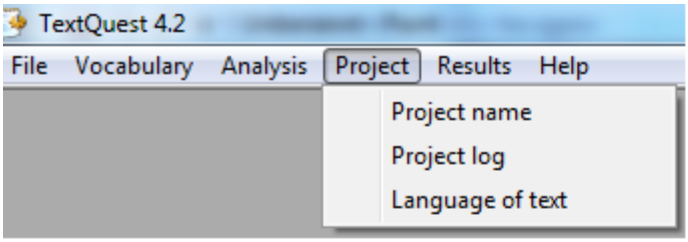
```
statistics:

- S 01:     9,667 words/text unit
- S 02:     3,273 characters/character strings
- S 03:    31,643 characters/text unit
- S 04:     0,213 foreign words/text unit
- S 05:    14,604 syllables/text unit
- S 06:     1,511 syllables/character strings



              stylistic criteria

              raw    %

- S 07:         4  0,81  too long sentences
- S 08:        16  3,25  too complex sentences
- S 09:         0  0,00  too many brackets
- S 10:       435  9,13  too long words
- S 11:         0  0,00  too many foreign words
```

The output also consists of frequency tables for words, syllables, sentence structuring characters, sentence ending characters, and their means.

```
frequency table


frequency syllables words foreign words SS chars SE chars brackets
      0       708       0         393       258       226       491
      1      2205       0          95       163       240         2
      2       932      16           5        56        20         0
      3       623      10           0        11         5         0
      4       241       4           0         4         2         0
      5        44       9           0         1         0         0
      6        13      12           0         0         0         0
      7         0      24           0         0         0         0
      8         0      37           0         0         0         0
      9         0      82           0         0         0         0
     10         0      99           0         0         0         0
     11         0      87           0         0         0         0
     12         0      63           0         0         0         0
     13         0      40           0         0         0         0
     14         0       6           0         0         0         0
     15         0       2           0         0         0         0
```

16	0	2	0	0	0	0
17	0	0	0	0	0	0
18	0	0	0	0	0	0
19	0	0	0	0	0	0
mean	1,51070	8,35091	0,21298	0,66734	0,61460	0,00406

10. The menu: project



Each project can have its own settings. The name of the project is important, because it is used for the generation of the file names. Also you can define what filters/samples to use, which exclusion lists, sort order tables, negation indicator tables, and how the external variables are described, and what language to chose.

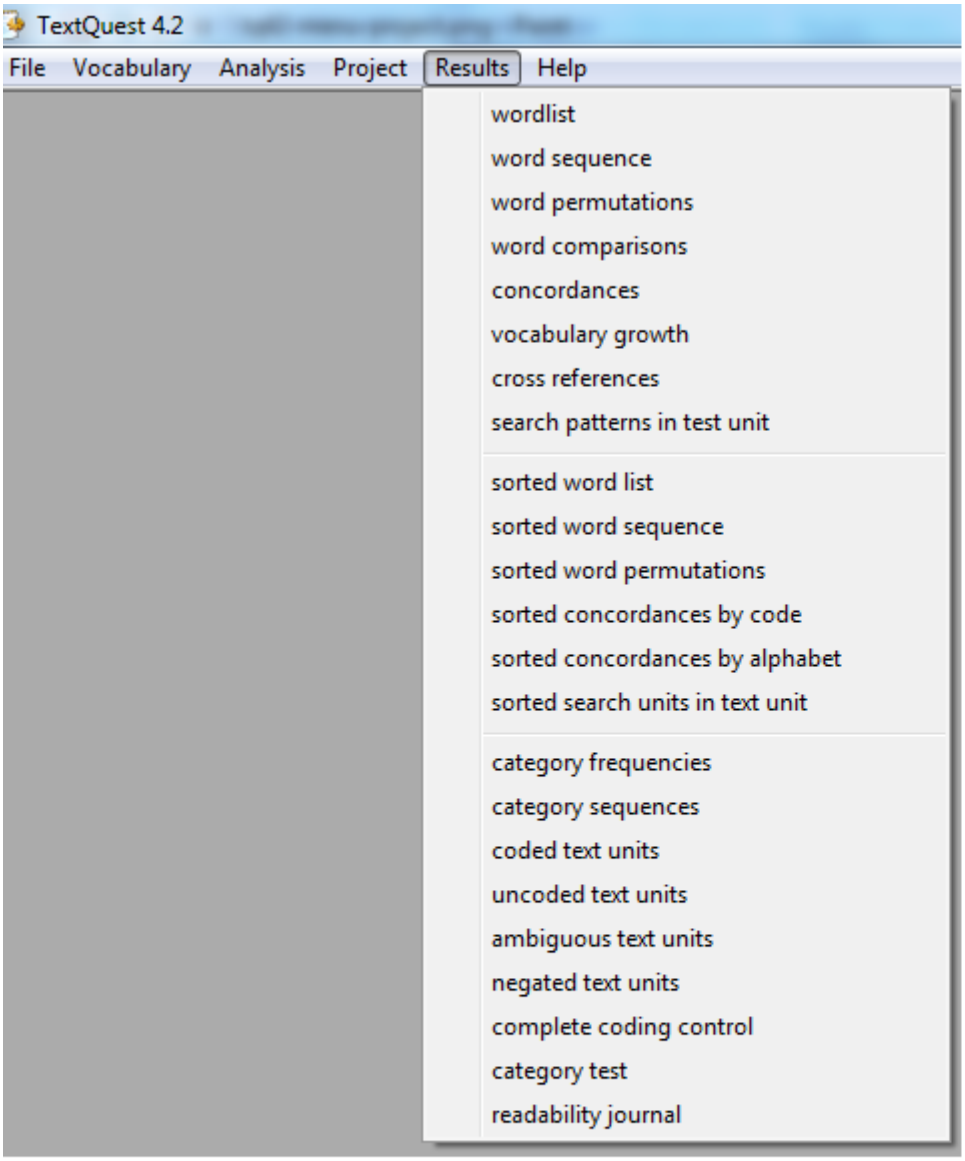
10.1 Project name

Here the name of the project can be defined. The project name points to the files, so it may contain drive and/or directory specifications. This feature frees the user to go through file selection menus.

10.2 Project log

The log stores all the results of the analyses you did, together with the time and the files. One can use the log file for documentation, and one can include it into an editor or word processor easily.

11. The menu: Results



The result menu allows you to browse through the results of the analyses, and the file belonging to the menu topic is loaded. However, you must stick to the naming conventions of **T**ext**Q**uest, because otherwise you cannot use this menu. Results from the vocabulary menu, sorted files, and analyses menu are separated by a line.

- word list. sorted ascending by alphabet
- word sequence: sorted ascending by alphabet
- word permutation: sorted ascending by alphabet
- word comparison: sorted ascending by alphabet
- concordances: unsorted
- vocabulary growth: unsorted
- cross references: unsorted
- search patterns in text unit: unsorted

- sorted word list: sorted descending by frequency
- sorted word sequence: sorted descending by frequency
- sorted word permutation: sorted descending by frequency
- sorted concordance by code: sorted ascending by code
- sorted concordance by alphabet: sorted ascending by alphabet
- sorted search patterns in text unit: sorted ascending by alphabet using search pattern as sort key

---

- category frequencies: the raw data matrix with the results of the content analysis as frequencies for each category
- category sequences: the sequence of codes for each text unit as a result from a content analysis
- coded text units: the text units that contain at least one search pattern
- uncoded text units: the text units that do not contain a single search pattern
- ambiguous text units: the text units that containt ambiguous search patterns
- negated text units: the text untis that contain negated search patterns
- complete coding control: a log file that shows the complete coding of a content analysis
- category test: the results of the category test that shows search patterns that occur more than once in a category system
- readability journal: shows the sentences and words that are too long, too complex, or contain too many foreign words

12. The structure of the TextQuest files

12.1 TextQuest-file: system file

The external variables are separated by tildes (~) and may consist up to 10 characters. Up to 50 external variables are possible. After the last external variable a vertical bar (|) follows, after that the number of words (5 digits) the length of the text (in characters), and a number sign (#). The text follows (maximum 100000 characters).

12.2 DIC file: search patterns

1	-	3	code (optional)
4	-	6	parameter field
7	-	200	search pattern

12.3 W?? file: word lists, word sequences, word permutations

1	-	6	frequency of the string
		7	free
8	-	1000	string

12.4 XRF file: cross references

			1. line
1	-	80	word
			following lines
			external variables, separated by tildes (~)

12.5 VEC file: sequence of codes

1	-	x	External variables: x=number*10
x+1	-	x+5	strings in the text unit, 5 digits
x+6	-	x+10	codes in the text unit, 5 digits
x+11	-	x+14	counter 1. category, 3 digits
x+15	-	x+17	counter 2. category, 3 digits
x+18	-	x+20	counter 3. category, 3 digits

12.6 TAB file: code counter

1	-	x	External variables: x=number*10
x+1	-	x+5	strings in the text unit, 5 digits
x+6	-	x+10	codes in the text unit, 5 digits
x+11	-	x+14	1. code, 3 digits
x+15	-	x+17	2. code, 3 digits
x+18	-	x+20	3. code, 3 digits

12.7 SIC file: concordances

The following example assumes 131 characters in one line.

1	-	3	code
		4	free
20	-	70	text before the search pattern
71	-	132	search pattern and following text

12.8 TTR file: TTR-dynamics

1	-	9	cumulated value of the types
10	-	18	cumulated value of the token
19	-	24	TTR-value
26	-	80	token (if included, otherwise empty)

13. List of information messages

Here all information message that can occur in **T**ext**Q**uestare listed and explained.

I 01 number of lines or text units read

I 02 data errors occured while building the system file

I 03 suwaco and wordbook: skipped text units

I 04 control sequences

I 05 empty lines read

I 06 empty text units

I 07 comment lines read

I 08 length of longest string

I 09 sum of words

I 10 sum of bytes

I 11 longest word and its position, and its length

I 12 number of text units written

I 13 word per text unit

I 14 longest text unit

I 15 chars truncated before a string

I 16 chars truncated after a string

I 17 suwaco and wordbooks: words read

I 18 numbers read

I 19 other (strings) read

I 20 strings (total) read

I 21 words/text unit

I 22 numerics/text unit

I 23 other strings/text unit

I 24 strings/text unit

I 25 id errors in crossref

I 26 because of minimum length excluded

I 27 because of maximum length excluded

I 28 because of minimum frequency excluded

I 29 because of maximum frequency excluded

I 30 sum of excluded strings

I 31 excluded stop words

I 32 Tuldava T

I 33 Somers S

I 34 number of numeric strings

I 35 sum of numeric strings

I 36 number of other strings

I 37 sum of all strings

I 38 records in tab files

I 39 records in vec files

I 40 coded text units

I 41 uncoded text units

I 42 negated text units

I 43 coding control

I 44 number of SITs

I 45 number of records in vector file

I 46 number of records in tab file

I 47 number of overlaps

I 48 sum of overflow in vec file

I 49 number of negated text units

I 50 number of negations

I 51 uncoded text passages

I 52 text passages where the code changed from the original code

I 53 ICRC coeffizient or concordances rejected

I 54 TTR-values ascending

I 55 TTR-values descending

I 56 TTR-values unchanged

I 57 TTR raw value

I 58 TTR-quotient

I 59 TTR value at 100 token

I 60 TTR value at 200 token

I 61 TTR value at 300 token

I 62 TTR value at 400 token

I 63 TTR value at 500 token

I 64 TTR value at 600 token

I 65 TTR value at 700 token

I 66 TTR value at 800 token

I 67 TTR value at 900 token

I 68 TTR value at 1000 token

I 69 TTR value at 2000 token

I 70 TTR value at 3000 token

I 71 TTR value at 4000 token

I 72 TTR value at 5000 token

I 73 TTR value at 10000 token

I 74 TTR value at 20000 token

I 75 TTR value at 30000 token

I 76 TTR value at 40000 token

I 77 TTR value at 50000 token

I 78 TTR value at 100000 token

14. Bibliography

Alexa, Melina; Cornelia Züll (1999): A Review of Software for Text Analysis. Mannheim: ZUMA-Nachrichten Spezial, Band 5.

Anonymous (1989): A Short Guide to the General Inquirer. In: Bulletin de Méthodologie Sociologique 24, p. 6-8.

Ballstaedt, Steffen-Peter; Heinz Mandl; Wolfgang Schnotz; Sigmar-Olaf Tergan (1981): Texte verstehen, Texte gestalten. München, Wien, Baltimore.

Bausch, Karl Heinz (1973): Zur Umschrift gesprochener Hochsprache. In: IDS, Gesprochene Sprache. Mannheim.

Bierschenk, Bernhard (1977): A Computer-Based Content Analysis of Interview Texts: Numeric Description and Multivariate Analysis. In: Didakometry 53, p. 42.

Bierschenk, Bernhard (1978): Content Analysis as Research Method. In: Kompendieserien 25, p. 93.

Bierschenk, Inger (1977): Computer-Based Content Analysis: Coding Manual. In: Pedagogisk Dokumentation 52, p. 113.

Boot, N.N.M. (1978): Ambiguity and Automated Content Analysis. In: MDN, Methoden en Data Nieuwsbrief van de Sociaal-Wetenschappelijke Sectie van de VVS, 3/1, p. 117-137.

Boot, M.N.M. (1979): Homographie, ein Beitrag zur automatischen Wortklassenzuweisung in der Computerlinguistik. Utrecht.

Bos, Wilfried; Christian Tarnai (1989): Angewandte Inhaltsanalyse in Empirischer Pädagogik und Psychologie. Münster.

Chotlos, John W. (1944): A Statistical and Comparative Analysis of Individual Written Language Samples. Psych. Monographs 56/ Nr. 2, p. 75-111.

Clubb, Jerome M; Erwin K. Scheuch (eds. 1980): Historical and Process-Produced Data. Stuttgart.

Cuilenberg, Jan J. van; Jan Kleinnijenhuis; Jan A. de Ridder (1988): Artificial Intelligence and Content Analysis: Problems of and Strategies for Computer Text Analysis. In: Quality and Quantity 22/1, p. 65-97

Dasgupta, Atis K. (1975): A Note on Content Analysis. In: Sociological Bulletin 24/1, p. 87-94.

Deichsel, Alexander (1975): Elektronische Inhaltsanalyse. Zur quantitativen Beobachtung sprachlichen Handelns. Berlin.

DeWeese III, Carroll (1976): Computer content analysis of printed media. A feasibility study. In: Public Opinion Quarterly 40, p. 92-100.

DeWeese III, Carroll (1977): Computer content analysis of 'Day Old' Newspapers: A feasibility study. In: Public Opinion Quarterly 41, p. 91-94.

Dohrendorf, Rüdiger (1990): Zum publizistischen Profil der "Frankfurter Allgemeinen Zeitung". Computerunterstützte Inhaltsanalyse von Kommentaren der FAZ. Frankfurt/M, Bern, New York, Paris.

Drewek, Raimund (1985): LDVLIB – Textanalyse mit System. In: Lehmacher, Walter; Allmut Hörmann (eds.): Statistik-Software. 3. Konferenz über die wissenschaftliche Anwendung von Statistik-Software. Stuttgart, p. 283-296.

Fan, David P. (1988): Predictions of Public Opinion from the Mass Media: Computer Content Analysis and Mathematical Modeling. (s.l.): Greenwood Press.

Faulmann, Carl (1880): Das Buch der Schrift. Wien, Reprint Nördlingen 1985.

Fischer, Peter Michael (1982): Inhaltsanalytische Auswertung von Verbaldaten. In: Huber, Günter L.; Heinz Mandl: Verbale Daten. Weinheim and Basel, p. 179-196.

Fonnes I: (1974): TEXT: A General Program Package for Text Processing. In: Contributed Papers: ISSC-SCSSD Workshop on Content Analysis in the Social Sciences, Pisa CNUCE, August 1974, p. 77-83.

Franzosi, Roberto (1990): Computer-Assisted Coding of Textual Data. An Application to Semantic Grammars. In: Sociological Methods and Research 19/2, p. 225-257.

Frisbee, B.; S. Sudman (1968): The Use of Computers on Coding Free Responses. In: Public Opinion Quarterly 32, p. 216-232.

Früh, Werner (1984): Konventionelle und maschinelle Inhaltsanalyse im Vergleich: Zur Validierung computerunterstützter Bewertungsanalysen. In: Klingemann, Hans-Dieter (eds.): Computerunterstützte Inhaltsanalyse in der empirischen Sozialforschung. Frankfurt/Main, p. 35-53.

Frow, John (1989): Formal Method in Discourse Analysis. In: Journal of Pragmatics 13/3, p. 333-341.

Giegler, Helmut (1991): Zur computerunterstützten Analyse sozialwissenschaftlicher Textdaten. Quantitative und qualitative Strategien. In: Hoffmeier-Zlotnik, Jürgen (ed.): Analyse qualitativer sozialwissenschaftlicher Daten. Opladen, p. 335-388.

Heinrich, Horst-Alfred (1996): Traditional versus computer aided content analysis. A comparison between codings done by raters as well as by INTEXT. In: Faulbaum, Frank; Wolfgang Bandilla (eds.): SoftStat '95. Advances in statistical software 5. The 8th Conference on the Scientific Use of Statistical Software. March 26-30, 1995 Heidelberg. Stuttgart, p. 327-333.

Heinrich: Horst-Alfred (1996): Generationsbedingte zeithistorische Erinnerung in Deutschland. Ergebnisdokumentation einer computergestützten Inhaltsanalyse mit INTEXT. (= Nationale Identität. Arbeitsberichte aus dem DFG-Projekt "'Nationale Identität der Deutschen"'. Messung und Erklärung der Veränderungsprozesse in Ost und West. Nr. 10), Mannheim.

Heinrich, Horst-Alfred (1996): Zeithistorische Ereignisse als Kristallisationspunkte von Generationen. Replikation eines Messinstrumentes. In: ZUMA-Nachrichten 39, p. 69-94.

Herdans, Gustav (1964): Quantitative Linguistics. London.

Johnson, Wendell (1944): Studies in Language Behaviour. Psych. Monographs 56/ Nr. 2

Klein, Harald (1988): INTEXT - ein Programmsystem zur computerunterstützten Inhaltsanalyse. In: Faulbaum, Frank; Hans-Martin Uehlinger (eds.): Fortschritte der Statistik-Software 1. Stuttgart, p. 574-581.

Klein, Harald (1990): New Possibilities and Developments of Text Analysis with INTEXT/PC. In: Faulbaum, Frank, Reinhold Haux; Karl-Heinz Jöckel (eds.): Fortschritte der Statistik-Software 2. Stuttgart, p. 487-494.

Klein, Harald (1990): INTEXT/PC – A Program Package for the Analysis of Texts. In: Universität Siegen (ed.): ALLC – ACH 90 The New Medium. Book of Abstracts & Conference Guide, p. 133-136.

Klein, Harald (1991): INTEXT/PC – A Program Package for the Analysis of Texts in the Humanities and Social Sciences. In: Literary and Linguistic Computing 6/2, p. 108-111.

Klein, Harald (1992): Validity Problems and their Solutions in Computer-Aided Content Analysis with INTEXT/PC and Other New Features. In: Faulbaum, Frank; Reinhold Haux; Karl-Heinz Jöckel (eds.): Advances in Statistical Software 3. Stuttgart, p. 483-388.

Klein, Harald (1993): INTEXT/PC – A Program Package for the Analysis of Texts. In: Steyer, Rolf, u.a (eds.): Proceedings of the 7th European Meeting of the Psychometric Society in Trier, Stuttgart, p. 219-221.

Klein, Harald (1993): INTEXT – a program system for the analysis of texts. In: Hřebiček, Luděk; Gabriel Altmann (eds.): Quantitative Text Analysis, p. 297-307. Trier: Wissenschaftlicher Verlag.

Klein, Harald; Helmut Giegler (1994): Correspondence Analysis of Text Data with INTEXT/PC. In: Greenacre, Michael; Jörg Blasius (eds.): Correspondence Analysis in the Social Sciences, p. 283-301. London: Academic Press.

Klein, Harald (1996): Computerunterstützte Inhaltsanalyse mit INTEXT – dargestellt am Vergleich von Nachrichtenfaktoren des Fernsehens. Münster.

Kleinen, Günter (1994): Die psychologische Wirklichkeit der Musik. Wahrnehmung und Deutung im Alltag. Kassel: Gustav Bosse Verlag.

Klingemann, Hans-Dieter (ed. 1980): Computerunterstützte Inhaltsanalyse in der empirischen Sozialforschung. Anleitung zum praktischen Gebrauch. Frankfurt am Main.

Klingemann. Hans-Dieter; Klaus Schönbach; Bernd Wegener (1978): Nachrichtenwerte und computerunterstützte Inhaltsanalyse. In: ZUMA-Nachrichten 2, p. 3-11.

Klingemann, Hans-Dieter; Peter Ph. Mohler (1979): Computerunterstützte Inhaltsanalyse (CUI) bei offenen Fragen. In: ZUMA-Nachrichten 4, p. 3-19.

Klingemann, Hans-Dieter; Peter Ph. Mohler (1980): Deutsche Diktionäre für computerunterstützte Inhaltsanalyse (1). In: ZUMA-Nachrichten 6, p. 53-57

Kramer-Santel, Claudia (1995): Die Darstellung des Umweltproblems in der Presse unter besonderer Berücksichtigung anreizkonformer Instrumente. Dissertation, Münster.

Kuckartz, Udo (1988): Computer und verbale Daten. Zürich.

Laffal, Julius (1990): A Concept Dictionary of English with Computer Programs for Content Analysis. Essex, Ct.

Lavigne, Gilles; Joelles Martin; Elise Nantel (1989): L'analyse de contenu assistée par ordinateur: L'option LIAO. In: La Revue Canadienne de Sociologie et d'Anthropologie, 26/4, p. 596-616.

Lenders, Winfried; Gerd Willèe (1986): Linguistische Datenverarbeitung. Ein Lehrbuch. Opladen.

Mandelbrot, Benoit (1961): On the Theory of Word Frequencies and on Related Markovian Models of Discourse. In: Roman Jakobson (eds.): The Structure of Language. Providence, p. 190-219.

McGee, Victor E. (1986): The OWL: Software Support for a Model of Argumentation. In: Behavior Research Methods, Instruments & Computers 18/2, p. 108-117.

McTavish, Donald G.; Ellen B. Pirro (1990): Contextual Content Analysis. In: Quality and Quantity 24/3, p. 245-265.

Messelken, H. (1989): Computerunterstützte Textanalyse. In: Historical Social Research 14/4, p. 86-93.

Mochmann, Ekkehard (1974): Automatisierte Textverarbeitung. In: Koolwijk, Jürgen van; Maria Wieken-Mayser (eds.): Techniken der empirischen Sozialforschung. 3. vol: Erhebungsmethoden.

Beobachtungen und Analyse von Kommunikation. München, p. 192-202.

Mochmann, Ekkehard (1985): Inhaltsanalyse in den Sozialwissenschaften. In: Sprache und Datenverarbeitung 9/2, p. 5-10.

Mohler, Peter Ph. (1980): Deutsche Diktionäre für computerunterstützte Inhaltsanalyse (2). In: ZUMA-Nachrichten 7, p. 42-44.

Mohler, Peter Ph. (1981): Deutsche Diktionäre für computerunterstützte Inhaltsanalyse (3) In: ZUMA-Nachrichten 8, p. 51-53.

Mohler, Peter Ph. (1985): Computerunterstützte Inhaltsanalyse: Zwischen Algorithmen und Mythen. In: Sprache und Datenverarbeitung 9/2, p. 11-14.

Mohler, Peter Ph.; Cornelia Züll; Alfons Geis (1989): Die Zukunft der computerunterstützten Inhaltsanalyse (cui). In: ZUMA-Nachrichten 25, p. 39-46.

Mohler, Peter Ph. (1989): Die linguistischen Leistungen der computerunterstützten Inhaltsanalyse. In: Batori, Istvan; Wilfried Lenders; W. Putschke (eds.): Computerlinguistik: Ein Internationales Handbuch der Computerunterstützten Sprachforschung und ihrer Anwendungen. Berlin.

Mohler, Peter Ph.; Katja Frehsen; Ute Hauck (1989): CUI: Computerunterstützte Inhaltsanalyse. Grundzüge und Auswahlbibliographie zu neueren Anwendungen. Mannheim: ZUMA-Arbeitsbericht, Nr. 89/09.

Muskens, George (1985): Mathematical Analysis of Content. In: Quality and Quantity 19/1, p. 99-103.

Nath, Detlev W. (1979): COFTA – Compiler für Textanalysen (Einführung). St. Augustin.

Richardson, M.G. (1979): Verzeichnis Deutscher Diktionäre für computerunterstützte Inhaltsanalyse. In: ZUMA-Nachrichten 4, p. 20-22.

Roberts, Carl W. (1989): Other than Counting Words: A Linguistic Approach to Content Analysis. In: Social Forces 68/1, p. 147-177.

Roberts, Carl W.; Roel Popping (1993): Computer-supported Content Analysis: Some Recent Developments. In: Social Science Computer Review 11, p. 283-291.

Salton, G.; C.S. Yang; C.T. Yu (1975): A Theory of Term Importance in Automatic Text Analysis. In: Journal of the American Society for Information Science 26/1, p. 33-44.

Schnurr, Paula P.; Stanley D. Rosenberg; Thomas E. Oxman (1992): Comparison of TAT and Free Speech Techniques for Eliciting Source Material in Computerized Content Analysis. In: Journal of Personality Assessment 58/2, p. 311-325.

Schönbach, Klaus (1979): Elektronische Inhaltsanalyse in der Publizistikwissenschaft. In: Publizistik 24, p. 449-457.

Schönbach, Klaus (1982): "The Issues of the Seventies". Elektronische Inhaltsanalyse und die langfristige Beobachtung von Agenda-Setting-Wirkungen der Massenmedien. In: Publizistik 27, p. 129-139.

Sedelow, Walter A.; Sally Y. Sedelow (1978): Formalized Historiography, the Structure of Scientific and Literary Texts. Part 1: Some Issues Posed by Computational Methodology. In: Journal of the History of the Behavioral Sciences 14/3, p. 247-263

Sells, P. (1985): Lectures on Contemporary Syntactic Theories. Stanford.

Singh, Jaspal (1985): Content Analysis. In: Guru Nanak Journal of Sociology 6/1, p. 37-44.

Smith, Robert B.; Peter K. Manning (1982): A Handbook of Social Science Methods. Volume 2: Qualitative Methods. Cambridge

Spack, Jones K.; M. Kay (1976): Linguistik und Informationswissenschaft. München.

Stone, Philip J.: (1962): The General Inquirer: A computer system for content analysis and retrieval based on the sentence as a unit of information. In: Behavioral Science 7, p. 484-494.

Stone, Philip J. and Cambridge Computer Associates Inc. (1968): User's Manual for the General Inquirer. Cambridge, Mass..

Stone, Philip J.: (1969): Improved Quality of Content Analysis Categories: Computerized Disambiguation Rules for High-Frequency English Words. In: Gerbner, G. et al. (eds.): The Analysis of Communication Content: New York, p. 199-221.

Tiemann, Rainer (1973): Algorithmisierte Inhaltsanalyse: Prozeduren zur Inhaltsanalyse verbaler Verhaltensweisen. Hamburg.

Trappes-Lomax, H.R. (1974): A Computer Based System for Content Analysis, a Review of the Edinburgh 'New Tagger' Version of the General Inquirer. Edinburgh.

Trauth, Michael (1992): Quantifizierende Textanalyse. Mit der Hilfe des Computers auf der Suche nach dem anonymen Autor. In: Historische Sozialforschung 17/1, p. 133-141.

Weber, Heinz-Josef (1976): Automatische Lemmatisierung. In: Linguistische Berichte 44, p. 30-47.

Weber, Robert P. (1983): Measurement Models for Content Analysis In: Quality and Quantity 17/2, p. 127-149.

Weber, Robert P. (1984): Computer-Aided Content Analysis: A Short Primer. In: Qualitative Sociology 7/1-2, p. 126-147.

Weber, Robert P. (1986): Correlational Models of Content: Reply to Muskens In: Quality and Quantity 20, p. 2-3, 273-275.

Weber, Robert P. (1990): Basic Content Analysis. 2. ed., Newbury Park.

Weih, Markus; Reinhold A; Richter-Schmidinger T; Sulimma AK; Klein, Harald; Kornhuber J.: Unsuitable readability levels of patient information pertaining to dementia and related diseases: a comparative analysis. Int Psychogeriatr. 2008 (6), p. 1116-1123.

Wickmann, Dieter (1969): Eine mathematisch-statistische Methode zur Untersuchung der Verfasserfrage literarischer Texte. Durchgeführt am Beispiel der "Nachtwachen" von Bonaventura mit Hilfe der Wortartübergänge. Köln/Opladen (Forschungsberichte des Landes NRW Nr. 2019)

Wilde Kelly, Ann; A.M. Sine (1990): Language as Research Data: Application of Computer Content Analysis in Nursing Research. In: Advances in Nursing Science 12/3, p. 32-40.

Wood, Michael (1980): Alternatives and Options in Computer Content Analysis. In: Social Science Research 9/3, p. 273-286.

Woodrun, Eric (1984): Mainstreaming Content Analysis in Social Sciences: Methodological Advantages, Obstacles and Solutions. In: Social Science Research 13/1, p. 1-19.

Züll, Cornelia; Robert P. Weber; Peter Ph. Mohler (1989): Computer-aided Text Classification for the Social Sciences: The General Inquirer III. Mannheim.

Züll, Cornelia; Peter Ph. Mohler; Alfons Geis (1991): Computerunterstützte Inhaltsanalyse mit TEXTPACK PC Release 4.0 für IBM XT/AT und Kompatible unter MS/DOS ab Version 3.0. Stuttgart.

Züll, Cornelia; Peter Ph. Mohler (eds.) (1992): Textanalyse. Anwendungen der computerunterstützten Inhaltsanalyse. Opladen.

15. Glossary

The glossary explains the technical terms used in this manual.

ambiguity: This problem occurs while defining search patterns for a category system (dictionary). Because search entries have to be defined unique, ambiguity must not occur. Example: pot. This can mean the same as a cup, but it can also mean a certain drug. The search pattern ' pot ' is ambiguous. It makes sense that you examine the context by doing a concordance of the text unit.

analysis unit: in a content analysis the analysis unit is the case or the observation in a statistical sense. Often the text unit and the analysis unit are identical, the unit is dependent on what hypotheses are to be tested.

blank: another word for space. A word is formed by all characters between two blanks (or other delimiters like start or end of a line). If one does not follow the regulations for typing, e.g. if one does not leave a blank after a comma, words cannot be proper separated from one another.

case folding: enabling case folding means that strings (mostly words) that are only different because they differ in lower/upper case letters are treated as the same by some **T**ext**Q**uest programs. Disabling case folding means that all differences matter, also the one that are based on differences in upper/lower case. For example: **That** and **that** are treated as one word if case folding is enabled and as two words if case folding is disabled. If you generate vocabularies, this option is availble. In English only words in the beginning of a sentence are written with capital letter (uppercase), whereas in German each noun starts with a capital letter, and there are words that have totally different meanings if they are written with the first letter lower- or uppercase (e.g. *würde* and *Würde*)

category: operationalisation of a theoretical construct with one or more search patterns (see there). Search patterns can be single words, parts of a word, a word stem, sequences of words like names, or word co-occurences.

category system: a group of several categories. Every category consists of at least one search pattern. Categories are the basis for content analyses.

character string: all characters between two blanks (see there), usually a word, but it can also be a part of a word or a word sequence.

coding unit: the coding unit (see content analysis) is the definition of a case. A new coding units starts with every new text unit. Aggregation can only performed with statistics software (e.g. Aggregate within SPSS).

column format: a raw text format that uses columns. Each external variable occupies the same columns on each line. The column format is often used when reading data from databases or statistical programs.

concordance: search patterns in their context. This is an analysis that shows search patterns and their context in one line (similiar to KWICs). The search patterns are in the center of a line, the rest consists of the context before and after the search pattern. In **T**ext**Q**uest the length of the line is variable.

content analysis: in the social sciences a content analysis is an analysis of communication content, mostly texts. It is used to test hypotheses, and a content analysis can be regarded as a set of rules to transform textual information into numbers. The set of rules is the category system (see there).

control sequences: these are use to generate a system file using control sequence format. The control sequences separate the text units and assign the values for the external variables.

control sequence format: one of the many input formats of raw text (see there) that works with control sequences that start with $. It is best used if you have to type in the text yourself.

cross reference: a list of all positions of a string where it occurs. A cross reference consists of all external variables and their positions within the text unit. Another name for a cross reference list is index.

default: each parameter or option that can be changed by the user has a value that is taken if the user doesn't specify the parameter, this is called the default, e.g. file names have default names derived from the name of the project.

dictionary: another term for category system. A dictionary consists of all search patterns that form the categories. Sometimes the term dictionary is also used in the sense of a word list.

digit: all strings where the first characters is a digit (0-9).

external variable: these variables represent attributes of a text. They must be specified by the user, up to 50 external variables are possible, at least one is required.

file: a form how to organise data. A file consists of logical records, each record consists of at least one variable. Logical records of a file of text units (the **T**ext**Q**uest system file) consist of the external variables, the number of words, the numbers of characters and the text. Each file has its own structure, the details are described in chapter *Structure of the files*.

filter: each analysis can be performed on the whole text or parts of it that are defined by the values of external variables. This process is called filtering or sampling.

column format: a raw text format that uses columns. Each external occupies the same columns on each line. The column format is often used when reading data from databases or statistical programs.

floating text: text in the format of a floating text is organised in a file that consists of text units as a logical record. This is the format a system file is organised. Another form of organising text is the vertical text format where a logical records consists of the external variables and one word.

homonym: a string that has more that one meaning. In a content analysis homonyms have to be disambiguated (see ambiguity). Example: pot. Meaning: cup or drug.

hyphenation: the hyphenation of words in a raw text is not allowed. All hyphenated words have to be eliminated before the system file is generated.

infix: a string (see there) that may occurs in any position within a word (see there), and used as a search pattern in a content analysis. If an infix occurs in the beginning of a string, it's called prefix (see there), if it occurs at the end of a string, it's called suffix (see there). In a strict sense an infix may not occur at the beginning or end of a string).

justification: reverse vocabularies are useful for the examination of word endings. For this purposes one can change the justification of the text to right jsutification, so that the word endings of each string are in the same column and therefore better to read. Left justification (the default) is best for non-reversed vocabularies.

KWIC: key-word-in-context, the context of a search pattern is displayed within a line of text. The search patterns always start in the middle of the line. KWICs are used for the inspection of the context of potentially ambiguous search patterns. The KWIC-lines can be alphabetically sorted by the search patterns.

KWOC: key-word-out of-context. Like KWIC, but the context is not limited to a line, is can be more than one line. If the context is the whole text unit, this is called search pattern in the text unit.

line format: line format assumes that the raw texts consists of lines only and that external variables are not important. There is only external variable, the line counter.

negation: the negation of search patterns may bias the results in a content analysis. **T**ext**Q**uest can detect negations on the basis of negation indicators before and after the search pattern. The algorithm can be tested with the appropriate rapport file.

numeral: a number written as a word (e.g.: one, eleven).

OCR: optical character recognition. This software transforms images into texts. Scanners are used for this, they work like photo copiers. The image from the scanner is processed by OCR software that recognises the characters and writes them to a text file. The error rate of the recognition depends on the text and the condition of the paper. Paper of newspapers is often of bad quality and results in bad recognition results. Have in mind that a recognition rate of 99 % means that ca. 10-15 errors are still on the page (of 1500 characters).

page format: the page format is a special format of the line format. There are two external variables, the line counter and the page counter. After a certain number of lines the page counter is incremented by one, this number can be specified by the users.

paragraph format: each paragraph forms a text unit if one uses this format to build a system file. The paragraph counter is the only external variable.

prefix: a string (see there) that is in the beginning of a word (see there). A prefix is a special form of an infix (see there). In content analysis that can be a single letter (or another character).

project: a project contains all files necessary for the analysis of one system file. The project name can contain drive and/or directory specifications.

raw text: digitised form of a text that can be processed without editing or converting by **T**ext**Q**uest, so that a system file (see there) can be generated. The raw text must have specific formats, see the chapter of data preparation for details.

reverse word list: word list (see there) where the words are listed in reverse order (the first character becomes the last, the last character becomes the first). Example: `small` becomes `llams`.

search pattern: at least one operationalisation of a category (see there). There are two types of search patterns in **T**ext**Q**uest:

1. strings (words, part of words or sequences of words)
2. word co-occurences)

special characters: all characters that neither start with a letter or a number. These are e.g. punctation marks or other characters of the characters set (IBM EBCDIC, PCs ASCII, Windows ANSI).

special word: see foreign word.

STOP-word: a word list (see there) contains all types (see there) of a text. Many of them are not useful for the definition of search patterns. Using a STOP-word file these can be deleted from a word list. Such a file contains articles, pronouns, prepositions and conjunctions.

string: a set of characters that is delimited by a blank in the beginning and the end (or other delimiters).

suffix: that part of a string (see there) that forms the end of a string (see there). Search patterns can be defined as suffixes.

system file: a file of text units (see there) that is the basic file for all forms of text analyses. They consist of external variables and the text, the latter is stored with variable length. A system file consists of at least one text unit (see there).

text unit: a text unit is the unit of all further analyses and dependent what is to be researched. In readability analysis a text unit must be a sentence, in coding open ended questions a text unit is one answer to one open ended question. More details are described in *Preparation of the text.*

token: another term for a string (see there) in a text, used in linguistics.

truncate: a string can be truncated if it exceeds the maximum length of 80 characters in the following applications: cross references, sorting (if a sort order table is enabled, the maximum length of a string is 38 characters), some forms of output of comparisons of word lists.

TTR: Type-Token-Ratio. The ratio between all different strings (types, see there) and the sum of all strings (token, see there). The value of the TTR is between 0 and 1; the higher it is, the more heterogeneous is the vocabulary of the text. A value of 0 indicates an empty input file, a value of 1 means, that each word occurs only once. The value of the TTR is dependent on the length of the text (Zipf's law)

type: the sum of different strings (see there) in a text.

vertical text: The logical record of a text consists of a word together with its external variables. The opposite is called floating text (see there), each logical record consists of a text unit (see there).

vocabulary: a vocabulary in **T**ext**Q**uest can be a word list, word sequences, or word permutations. All use a system file as a basis, exclusion lists can be used, also other exclusion criteria like frequency and/or length of a string.

word: a word within a text unit are all characters, that are between two blanks (or another delimiter like start or end of a line). The more precise expression is string (see there), although most strings are words.

word co-occurence: several word or any parts of them that must occur within one text unit. Up to 6 word roots can be in a word co-occurence. These can be searched within a text unit in three different modes that vary the order and the distance of the word co-occurences how they must occur in the text (see chapter about the search patterns).

word list: a list of all types (see there) together with their frequency. Sometimes the term frequency table is also used. The word list is sorted alphabetically in ascending order, but the sort order can be changed. Using sort order tables is possible.

word length: the number of characters in a string. Usually the length of strings does not exceed 20 characters, but in languages like German compound words can become very long. In some analyses, e.g. vocabulary comparisons, words maybe truncated after the 39th character. The word length is also an exclusion criterion for vocabularies.

word permutation: are all combinations of two words within a text unit. Each word is combined with every word that follows up to the end of the text unit.

word root: A string (see there), that can be part of another string. Word roots can be in prefix, infix or suffix position (see there), they can also used in word co-occurences.

word sequence: a word sequence consists of a number of words that follow each other within a text unit, e.g. a phrase like "raining cats and dogs" or names like "Tony Blair" or "James Dean". The number of words that make up a word sequence can be varied, values between 2 and 9 make sense. Word sequences can be used to find phrases, and they can be the basis of word co-occurences in search patterns.

Index