freeWAIS-sf

The enhanced freeWAIS distribution
Edition 0.5, for freeWAIS-sf 2.0
October 1995

by Ulrich Pfeifer

1 Overview

The freeWAIS-sf distribution contains clients, servers, and auxiliary programs for the TCP/IP protocol known as *WAIS*. See Section 1.1 [Protocol], page 1. The distribution is focussed on the WAIS server `waisserver` and the indexer `waisindex`. Clients are distributed only for convenience. You can use any WAIS client to query the freeWAIS-sf server.

1.1 The WAIS Protocol

The WAIS protocol is based on the ANSI standard Z39.50 Version 1 (aka Z39.50 V1 or Z39.50 88). It uses only a subset of features of the Z39.50. Namely no requests which require the server to preserve some state are used. The WAIS protocol extents this basic features with elements for relevance feedback. The WAIS protocol is sometimes slightly incorrect also referred to as Z39.50 V1.

This ad hoc extension found their way back to the Z39.50 standard when the next version Z39.50 V2 (aka Z39.50 92) was defined. For implementation of WAIS-like (remember: WAIS is the name of the original protocol) the exist a definition of subset called the "WAIS Profile". Often Z39.50 V2 is used as synonym for this profile. freeWAIS-sf currently does **not** support Z39.50 V2.

1.2 Client Server Architecture

A WAIS System consists of clients talking to a server via an TCP/IP network using the WAIS protocol. Servers answer search requesta using auxiliary data structures called the *index*. These are created for the original documents by the `waisindex` program. Retrieve-requests are answered by the server fetching (parts of) the original files. As set of documents together with their associate index is called a *WAIS database*.

```
                +------+
                |Client|
                +------+
                    |
                 TCP/IP
                 network
                    |
                +------+
          +---->|Server|<----+
          |     +------+     |
          |                  |
       +-----+           +---------+
       |Index|           |Documents|
       +-----+           +---------+
          ^                  |
          |     +-------+    |
          +-----|Indexer|<---+
                +-------+
```

2 Getting freeWAIS-sf and related stuff

2.1 Mirror Sites

Most of the ftp accessable Stuff is mirrored on different sites. Please check archie or one of the sites below. The load on our server is growing out of bounds. We had to close it for the public - for the big files at least. Since all Packets are routed over the EUnet site, it will be faster to get the stuff from there as from ls6-www.

`ftp.wsct.wsc.com`
: '`/pub/freeWAIS-sf/`'

`ftp.sysadmin.com`
: '`/pub/admin/www/misc/wais-sf/`'

`ftp.maxwell.syr.edu`
: '`/infosystems/wais/FreeWAIS-sf/`'

`ftp.germany.eu.net`
: '`/pub/infosystems/wais/Unido-LS6/`'

`ftp.switch.ch`
: '`/mirror/wais/servers/sf-alpha/`'

`ftp.leo.org`
: '`/pub/comp/infosys/wais/freeWAIS/`'

`ftp.egg.com`
: '`/pub/infosystems`'

If you want to become a mirror site for freeWAIS-sf, send a mail to '`wais@ls6.informatik.uni-dortmund`' containing:

- The ip-address and ip-name of the host you mirror from.
- The ip-name of the serveryou want to redistribute and the directory where the distribution will be available.
- The frequency you will mirror the distribution.

2.2 Documentation

freeWAIS-sf-faq

The freeWAIS-sf FAQ. Information which did not found its way in this manual.

- '`ftp://`*mirror-site*`/`*mirror-dir*`/freeWAIS-sf-faq`'
- '`http://www.cis.ohio-state.edu/hypertext/faq/usenet/wais-faq/\ freeWAIS-sf/faq.html`'

comp.infosystems.wais FAQ

Very old c.i.w FAQ. Some Pointers to clients.

'`http://www.cis.ohio-state.edu/hypertext/faq/usenet/wais-faq/\`
`getting-started/faq.html`'

This Manual

- '`ftp://`*mirror-site*`/`*mirror-dir*`/freeWAIS-sf-2.0/fwsf.texi.gz`'
- '`ftp://`*mirror-site*`/`*mirror-dir*`/freeWAIS-sf-2.0/fwsf.dvi.gz`'
- '`ftp://`*mirror-site*`/`*mirror-dir*`/freeWAIS-sf-2.0/fwsf.ps.gz`'
- '`ftp://`*mirror-site*`/`*mirror-dir*`/freeWAIS-sf-2.0/fwsf.info.gz`'

WWW95 Paper

'`http://www.igd.fhg.de/www/www95/papers/47/fwsf/fwsf.html`'

2.3 Sources

freeWAIS-sf-2.0

'`ftp://`*mirror-site*`/`*mirror-dir*`/freeWAIS-sf-2.0/freeWAIS-sf-2.0.tar.gz`'

waisperl For linking Perl with the freeWAIS-sf libraries. Used with SFgate. '`ftp://`*mirror-site*`/`*mirror-dir*`/freeWAIS-sf-2.0/waisperl/Wais-2.0.tar.gz`'

SFgate The WWW Gateway for freeWAIS-sf.

'`ftp://`*mirror-site*`/`*mirror-dir*`/SFgate-4.0/SFgate-4.0.tar.gz`'

Example Format Files

'`ftp://`*mirror-site*`/`*mirror-dir*`/fmt-examples`'

Rainer Klute's Imake extension to CERN httpd 3.0

'`ftp://ftp.germany.eu.net/pub/infosystems/www/cern/WWW-Imake.tar.gz`'

3 Building and Installation

3.1 Unpacking the Distribution

The file '`freeWAIS-sf-2.0.tar.gz`' is a `tar` archive compressed with GNU `gzip`. Unpack it with the following command:

```
gunzip -cd freeWAIS-sf-2.0.tar.gz | tar xvf -
```

The directory structure is as follows:

```
                  |-FIELD-EXAMPLE
                  |
                  |-bin
                  |
                  |-ctype
                  |               |-CNIDR
                  |               |
                  |               |-SF
                  |-doc-----------|
                  |               |-original-TM-wais-|-manl
                  |
|-freeWAIS-sf-2.0-|-ir
                  |
                  |-lib
                  |
                  |-regexp
                  |
                  |-ui
                  |
                  |-x
```

The directories contain

'`FIELD-EXAMPLE`'
: some test data and procedures, which check the system once you build it.

'`bin`'
: some shell scripts. Most of them for displaying non text documents after retrieval. These scripts are not maintained. They are included for convenience and drawn form freeWAIS-0.2.

'`ctype`'
: the code for the 8-bit character sets.

'`doc`'
: the documentation which accompanied freeWAIS-0.2 in the subdirectories '`CNIDR`' and '`original-TM-wais`'. In '`SF`' the documentation written for freeWAIS-sf can be found.

'`ir`'
: the main code parts for the indexer and the server.

'`lib`'
: some replacement of C functions which might be missing on some systems.

'`regexp`'
: the code for regular expression matching.

'ui' some standard clients

'x' the standard X11 client.

3.2 Running Configure

The distribution comes with a `Configure` script generated by `metaconfig` which you should run next. It determines various system properties and asks you some questions.

3.2.1 Files generated

`Configure` generates some files:

'config.h, confmagic.h'
: This is where most of the option settings go. You may edit 'config.h' and change some of them. This file is referenced via `#include` statements by most C files.

'config.sh'
: Here most setting for your directory layout and location of programs end up. This file is is sourced by all `*.SH` files when producing the corresponding files.

  You may change settings in 'config.sh' and rerun `Configure` with the options `-der`.

'Makefile'
'FIELD-EXAMPLE/Makefile'
'ctype/Makefile'
'doc/original-TM-wais/manl/Makefile'
'install'
'ir/Makefile'
'lib/Makefile'
'regexp/Makefile'
'ui/Makefile'
'bin/Makefile'
: Makefiles for bulding and installing the distribution.

'x/Imakefile'
: Imakefile for building and installing the X11-Clients.

'bin/catalog'
'bin/dictionary'
'bin/inverted_file'
'bin/makedb'
'bin/mkfmt'
'bin/ws'
: Some scripts which depend on the path to `perl`.

3.2.2 Questions

Run `Configure` by submitting the following to your shell:

```
./Configure
```

`Configure` will display a startup message and begin to check the required system properties.

Intermixed with questions about variaus system properties there are some questions about the desired properies if the freeWAIS-sf system itself. Theese are explained in little more detail below.

Finally `Configure` prints the name of the files it generates and some short advice how to proceed.

3.2.2.1 Regular Expressions

If your system has support for regular expression, the first question will be:

```
...
Checking if your systems regexp implementation works ...
Yes, it works
Do you want to use your systems regexp.h? [n]
```

It is usually save to accept the default '`no`' and use the regular expressions included in the distribution. If that does not work (this maybe the case with IRIX) try your systems regular expressions by answering '`yes`'. In fact every answer other that '`no`' will be interpreted as '`yes`'.

3.2.2.2 Huge Headlines

The next question refers to the length of the headline file. If you will not have large databases, answer '`no`' here. To estimate the size of your headline file, multiply the number of documents you want to index with the medium headline length. If this size fitz in three bytes, it is save to accept the default.

```
Will you have HEADLINE files greater than 16 MB [n]
```

`bool` **MYREGEXP** '`config.h`'
: If the answer is different from '`no`', a `#define` for `MYREGEXP` is added to '`config.h`'.

3.2.2.3 Proximity vs. String Search

Next decision selects one of two possible query extensions.

```
This version of freeWAIS-sf supports new proximity operators by Tom
Snee.  He also fixed the string search code.  You can now enable them
at the cost of dropping the string search capability. There is
currently no description of the proximity operators.  Have a look in
ir/query_l.l to learn about them;

Use proximity instead of string search? [n]
```

bool **PROXIMITY** 'config.h'

If the answer is different from 'no', a #define for PROXIMITY is added to 'config.h'.

3.2.2.4 Character Set

Configure now asks for the character set, you want to use. All characters for which your isalnum(c) returns true are assumed to be legal within words.

You can augment this character set by additional chars. If you want 'C++' to be a valid word, you must make '+' an allowed character. This mechanism is usually used to add country specific ISO characters to the default ASCII set.

```
You can compile freeWAIS-sf with it's own ctype package. You should do
this, if you want to use special (country specific) chars, which are
not supported by your systems ctype.

Use your systems ctype? [n]
```

bool **MYCTYPE** 'config.h'

If the answer is different from 'no', a #define for MYCTYPE is added to 'config.h'. This also leads to the inclusion of the 'ctype' directory in the building process. So a option -I../ctype is added to the compiler flags.

The following question is asked only if the default is accepted.

```
I will now ask for your special letters.  If you do not want to give
the now, edit config.h after this Configure run. Input your upper case
letters in the same order than your lower case letters. toupper() and
tolower() depend on this order. Input letters which are upper and
lower case in both strings.


what are your upper case letters [äöüß]?⋆[RETURN]


what are your lower case letters [ÄÖÜß]?⋆[RETURN]
```

string **LCHARS** 'config.h'

string **UCHARS** 'config.h'

The character strings are added as extension of LCHARS rsp. UCHARS. Note that the order of character matters, since the index position is used for the macros toupper(*c*) and tolower(*c*).

3.2.2.5 LOCAL_SEARCH

You can compile and link the clients with the capability to search index files directly. So you need not to install a server, for local searches. The clients will be greater, but faster with local searches.

You will have to run the test (see Section 3.3.1 [Tests], page 10) manually without the switch!

```
Do you want to compile with -DLOCAL_SEARCH? [y]
```

bool **LOCAL_SEARCH** 'config.h'

Server code is linked to the clients when true.

3.2.2.6 Modified URL handling

You can modify the URL document type, to put the URL of the indexed document in the document id instead of the headline. If you use this modification you can customize the headline (e.g. with the '-t fields' option). Also it is not required to keep a copy of the documents for retrieval with the WAIS server. But currently only SFgate can handle this modified document-ids. Normal clients will not correctly interpret the document-ids and try to retrieve the document from a WAIS- instead of the corresponding HTTP server. See section "Overview" in *The SFgate Manual*.

```
Do you want to use the modified URL handling? [y]
```

bool **URLDOCID** 'config.h'

The DocumentId is used to carry the URL to the client. The server will be unable to access the document. You can remove it after indexing.

3.2.2.7 Installation Prefix

```
Installation prefix to use? (~name ok) [/usr/local/ls6/wais]
```

3.2.2.8 Synonym Files

I added a patch from Alberto Accomazzi for speeding up usage of synonym files. He writes about his patch:

> For those of you who have fairly large synonym files (> 10Kb) and are running the software on a machine that supports shared memory (your machine does) enabling this feature will speed up the waisserver response time by a significant factor.
>
> For those of you who do not have shared memory, I have rewritten the memory allocation part of 'synonym.c' so that bigger memory chunks are allocated and used rather than allocating memory for each word and synonym, so the code should be a little faster for you too.
>
> ```
> Do you want to use shm cache? [n]
> ```

3.2.2.9 Registration

As distributed, freeWAIS-sf Release 2.0 will send an UDP packet to my workstation every time waisserver reindexes his info database, containing your (numeric) UID, your operating system, your compiler version and the freeWAIS-sf version.

This is *just* because I would like to get an idea of to which systems/compilers freeWAIS-sf has be ported and how many people use it, and keep on using it (rather than tried it once folks). It will *never* become a licensing scheme or some crazy thing like that. But, you can disable it by answering '`yes`' to the following question. If you do that, please let me know, if you are running freeWAIS-sf on a system/compiler, which is not mentioned in the '`README`'.

```
Disable the UDP packet sending? [n]
Ok. Thank you for your trust.
```

3.3 Compiling the system and running the tests

When you have generated the makefiles using `Configure` (see Section 3.2 [Configure], page 6), you can compile the system by just entering '`make all`'. This will also index a test database and run some tests. Here is the list of the make targets:

`make` Build the system and run the tests.

`make install`
: Install binaries and scripts.

`make install.man`
: Install the manual pages.

`make clean`
: Remove object files, libraries, backups, ...

`make realclean`
: Remove files generated by `flex`, `bison`, `dvips`, `latex`, `waisindex`.

If you need the x clients `cd` to the x subdirectory and enter `xmkmf` and `make depend` prior to `make all` and `make install`

3.3.1 Running the tests

After successful compilation the `make` process indexes a test database and runs some queries. You will have to start the tests manually, if you did not chose the `LOCAL_SEARCH` option. See Section 3.2.2.5 [LOCAL_SEARCH], page 8.

```
cd FIELD-EXAMPLE
../ir/waisserver -d . -p 9565
make test
...
kill waisserver-pid
```

The results of the queries are compared to expected results contained in the distribution. Depending on your choices during the configuration (see Section 3.2 [Configure], page 6) some of the tests will fail. These are mentioned below. The `imake` makefiles will avoid the test which are expected to fail.

Here is the list of tests:

PLAIN ‘`Probabilistic Indexing`’

BOOLEAN
: ‘`au=(pennekamp or fuhr) and processing`’

COMPLEX
: ‘`py==1995 and (ti=(Retrieval freeWAIS) or au=pfeifer)`’

FIELD ‘`au=pfeifer`’

DATE ‘`ed>930101`’

GERMAN
: ‘`au=großjohann`’
: This will only work if you made ‘`ß`’ a legal chracter. See Section 3.2.2.4 [Character Set], page 8.

LITERAL ‘`'Enhanced Retrieval'`’
: This test will fail, if you chosed the proximity operators during configuration. See Section 3.2.2.3 [Proximity Search], page 7.

NUMERIC
: ‘`py==1995`’

PARTIAL ‘`Pfeif*`’

PROXIMITY
: ‘`(Searching w/2 Documents) or (Development pre/3 Linear)`’
: This test will succeed only, if you chosed the proximity operator during configuration. See Section 3.2.2.3 [Proximity Search], page 7.

If some of the tests supposed to work fail, have a look at the ‘`FIELD-EXAMPLE`’ directory. For each test there is a pair of file: ‘*testname*`.is`’ and ‘*testname*`.should`’. ‘*testname*`.should`’ is what the query result was on my machine and ‘*testname*`.is`’ was produced running on your system. Compare the files and decide, if you can live with the differences, e.g. if only the weighting differs, you should be able to use the system.

3.4 Installation

If the tests where successful, you can install the system using ‘`make install`’. See Section 3.3 [Compiling], page 10.

3.4.1 Installation Directories and Files

This is how your installation directory should look like after installation. See Section 3.2.2.7 [Installation Prefix], page 9.

```
                            |-SFproxy
                            |-catalog
                            |-check-sources
                            |-dictionary
                            |-getaddrs
                            |-inverted_file
                            |-makedb
                            |-mkfmt
                            |-server_stats
                            |-stats.awk
                            |-stringtoany
                            |-swais
                            |-trunc
                            |-wais-gif-display
              |-bin---------|-wais-html-display
              |             |-wais-jfif-display
              |             |-wais-jpeg-display
              |             |-wais-pict-display
              |             |-wais-ppm-display
              |             |-wais-tiff-display
              |             |-waisindex
              |             |-waisping
              |             |-waisq
       prefix-|             |-waisretrieve
              |             |-waissearch
              |             |-waisserver
              |             |-ws
              |             |-xwais
              |             |-xwaisq
              |
              |             |-X11-------------|-app-defaults----|-Xwais
              |-lib---------|-XwaisHELP
              |             |-XwaisqHELP
              |             |
              |             |-emacs-----------|-lisp------------|-wais.el
              |
              |                               |-catalog.1
              |                               |-dictionary.1
              |                               |-inverted_file.1
              |                               |-makedb.1
              |                               |-mkfmt.1
              |             |-man1------------|-waisindex.1
              |-man---------|                 |-waisq.1
                            |                 |-waissearch.1
                            |                 |-waisserver.1
                            |                 |-xwais.1
                            |                 |-xwaisq.1
                            |
                            |-man3------------|-ftw.3
                                              |-scandir.3
```

- The '.../bin' directory must be added to your path.
- The file '.../lib/app-defaults/Xwais' must be found by your X11 server. You can safely move it to some directory (e.g. '/usr/local/X11/lib/X11/app-defaults') where your server already searches or add the '.../lib/app-defaults/Xwais' directory to the search path e.g. by adding '.../lib/app-defaults/Xwais,%N' to the environment variable XUSERFILESEARCHPATH.
- The file '.../lib/emacs/lisp/wais.el' must be found by Emacs (see section "Overview" in *The GNU Emacs Manual*). You can safely move the file to your site lisp directory or add '.../lib/emacs/lisp' to your emacs load-path.
- You should also add '.../man' to your MANPATH. Note that the manuals are out of date!

3.4.2 Installing the Server

To install the server for being started by inetd proceed as follows. The description applies to SUNOS 4.1.3 but should be similar on other *NIX systems.

You do not have to install a server, if your clients are compiled and linked with the LOCAL_SEARCH (see Section 3.2.2.5 [LOCAL_SEARCH], page 8) option on. Without a server only clients which have access to the database files can do searches.

Add a line for the WAIS protocol to your '/etc/services':

```
wais            210/tcp
```

You can use any term instead of 'wais' if you want. But 'z3950' is misleading (see Section 1.1 [Protocol], page 1). You can choose a port different from 210 if you want. Make sure, that your database descriptions contain the right port then (see Section 4.3.1 [Database Description], page 20).

To '/etc/inetd.conf' add a (one!) line:

```
wais   stream  tcp    nowait  wais    \
       installdir/bin/waisserver waisd.d -d databasedir  -e logfile
```

Send the inet daemon a 'HUP' signal:

```
kill -HUP inetd-pid
```

4 Building Databases

If you successfuly compiled and installed the system, you already have build your own database. This database will serve as an example throughout this section.

Let's first define some terms.

Document The entities you want to search for.

Source The files containing the documents you want to make searchable.

Index A set of files generated by `waisindex` which are used by the server to perform searches. See Section 4.3 [Index Files], page 19.

Database The combination of source and index.

Format The layout of the documents including the separation of documents in the files.

Field The Format description might associate regions in the documents with certain concept types. The sender of an email might be associated to the concept '`author`'. Searches might be constrained to such an concept type or *field* for short.

Type A type is attached to each document in the database. The possible types roughly correspond to mime content types. Possible types are '`TEXT`', '`HTML`', '`URL`', ...

In general you have two options when indexing a new database: You can use a build-in format or use the '`-t fields`' option. The first alternative is not covered by this manual since the latter is more general and you can emulate the build-in formats with the field indexing.

4.1 `waisindex` Options

The following is mostly drawn from the `waisindex` manual.

`-d` *index-filename*
This is the base filename for the index files. Therefore if '`/usr/local/foo`' is specified, then the index files will be called '`/usr/local/foo.dct`' etc. The index should be stored on the local file system of the machine running waisindex. It works over NFS, but it is much slower.

`-a` Append this index to an existing one. Useful for incremental additions or updates. This will only add onto an index, so that if a file has changed, it will get reindexed, but the old entries will not be purged. Therefore, to save space, it is a good idea to reindex the whole set of files periodically.

`-r` Recursively index subdirectories.

`-mem` How much main memory to use during indexing. This variable will have a large effect on how fast indexing is done.

`-export` This causes the resulting source description file to include the host-name and tcp-port for use by the clients. Otherwise the file contains no connection information, and is expected to be used only for local searches.

`-nocat` Inhibits the creation of a catalog. This is useful for databases with a large number of documents, as the catalog contains 3 lines per document.

`-T` *type* Sets the type of the document to *type*.

`-t` *format* This is the format of files that are handled by waisindex. To find out the list of currently known types, execute the waisindex command with no arguments and it will list them. Use '`-t fields`' to use the field indexing. The file '*database*`.fmt`' must contain the format description.

`-stop` This option refers to the list of otherwise valid words which will be excluded from indexing and therefor searching.

If '`-stop`' is given the build-in stop list is *not* used. In any case the file '*database*`.stop`' might contain stopwords separated by newlines. If fact `waisindex` might add some words, if they occur to often.

filename ...

These are the files that will be indexed according to the arguments above. To insure the files are registered in the filename list correctly, it is advised that these are full paths (beginning with a '`/`'). If the database is used from a machine other than the machine on which the index is created, this should be a machineindependent path.

`-stdin` Read the names of the files to index from standard input instead from the argument line.

`-line` Makes the proximity operators use the lines aus units instead of words.

4.2 Bulding a Format Description

The trickiest part when building a new database is to write a format description — the horrible '*database*`.fmt`' files.

An important thing to know when writing format descriptions is that `waisindex` parses sources files *line by line*. All of the nice regular expressions have to match a single line. So including multiple linefeeds in an expression will cause the matcher to fail.

4.2.1 Regular Expressions

All regular expressions in the format description must be included between two '`/`'. Therefore a '`/`' in the regular expression must be escaped with a '`\`'. To match '`TCP/IP`' write '`/TCP\/IP/`'. This in consequence means, that a '`\`' must be escaped too ;-). Other escape sequences interpreted by the parser are:

`\n` A newline character. See Section 4.2 [Format Description], page 16.

`\r` The carriage return character.

`\f` The formfeed character

`\`*x* The characer `C-`*x* (control *x*) where *x* ranges from '`A`' to '`Z`'.

After escaping is resolved, the string is passed to a standard regular expression package (or your systems one see Section 4.2.1 [Regular Expressions], page 16).

Operator Meaning

x	the character '*x*'
"*x*"	an '*x*', even if *x* is an operator
x	an '*x*', even if *x* is an operator. *Remember that the parser eats one '\'. So use '`/\\*/`' to match a '`*`' in the text.*
`[`*xy*`]`	the character '*x*' or '*y*'
`[a-c]`	the characters '`a`', '`b`' or '`c`'
`[^`*x*`]`	any character but '*x*'
`.`	any character but newline
`^`*x*	an '*x*' at the beginning of a line
x`$`	an '*x*' at the end of a line
x`?`	an optional '*x*'
x`*`	0,1,2, ... instances of '*x*'
x`+`	1,2,3, ... instances of '*x*'
x`|`*y*	an '*x*' or a '*y*'
`(`*x*`)`	an '*x*'
x`{`*m*`,`*n*`}`	*m* through *n* occurrences of '*x*'

4.2.2 Structure of the Format Descriptions

A format description consist of three parts. The first part — usually just one line — describes how the files should be split into documents. The second part, called the *Layout Section* defines how a headline for each document should be computed.

The last and usually largest part defines which fields should be generated and how the indexing should be done.

4.2.2.1 Separation of Documents

A format description begins with a '`record-sep:`' directive.

```
record-sep: regexp
```

Here *regexp* might be '`/\f/`' to match a line containing a formfeed character. Note that the line matching the '`record-sep:`' regular expression is never indexed!

4.2.2.2 Layout Section

The layout section must be embraced by '`layout:`' and '`end:`'. It can contain multiple '`headline:`' and one '`date:`' directive. Each '`headline:`' keyword defines a region in the document which should be copied to a section of the headline. The region, defined by the regular expressions which start and end it can occur any number of times in the document. The order of the directives defines the order of the sections in the headline.

`headline:` *start end width [skip]*

The above line advises the indexer to copy the text between the matches for *start* and *end*, optionally skipping the text matched by *skip* to the next *width* characters of the headline.

The '`date:`' directives defines how a date to be associated with the document can be extraced. Note that this date is not usable for searches. It ist part of the headline and displayed to the user by some clients.

`date:` *start scanfarg d-m-y d-m-y d-m-y skip*

The *start* and *skip* work like above. The *scanfarg* parameter is passed to `scanf` to read in year, month, and day. The order of the arguments is indicated by the three *d-m-y* where each might be '`year`', '`month`', and '`day`'. The '`month`' might be followed by '`string`' to indicate that the month will be given as a standard three-letter abreviation.

Here is complete example:

```
layout:
headline: /^PY: / /^[A-Z][A-Z]:/ 5 /^PY: */
headline: /^AU: / /^[A-Z][A-Z]:/ 21 /^AU: */
headline: /^TI: / /^[A-Z][A-Z]:/ 41 /^TI: */
date:     /^ED: / /%d-%3s-%d/ day month string year /^ED: [^ ]/
end:
```

4.2.2.3 Definition of Field and Index Types

The field specification part of the format description is made up of multiple '`region:`' ... '`end:`' groups. Each is mapping a region of text to a set of query categories aka fields.

`region:` *start [skip]*
   fieldlist options indexspecs
`end:` *end*

The *start*, *skip*, and *end* expressions define the region of the document for which the index specification applies.

fieldlist is the list of fields each optionaly followed by a description string enlosed in double quotes ('`"`'). The description is entered in the database description (see Section 4.3.1 [Database Description], page 20).

Options include the directives '`numeric:` *skip width*', '`date:` *d-m-y d-m-y d-m-y*', and '`stemming`'. The first option advises the indexer to allow numeric values only and makes sure that numeric atomic search expressions will work with the categories. The second advices the indexer to convert the date in the region to the '`yymmdd`' format prior to index it numerically. The last option tells the indexer to stemm the words in the region before entering them in the index. When searching, the server also stemms search terms prior to the index lookup.

indexspecs is a list of index types along with a keyword indicating if the region should be mapped to the designated categories (`LOCAL`), the default category (`GLOBAL`) or to both (`BOTH`). The default category is used when none is specified in the query. See Appendix B [Query Syntax], page 41.

Index types currently supported are `TEXT`, `SOUNDEX` and `PHONIX`.

Consider the following example:

```
region: /^AU: /
        au "author names" SOUNDEX LOCAL TEXT BOTH
end: /^[A-Z][A-Z]:/
```

To the indexer this means:

For all words starting with '`AU: `' at the beginning of a line up to a line which starts with two capital letters followed by a colon and a blank, put the word in the default and the '`au`' category and its soundex code only in the '`au`' category.

Thus an author name can be found in the created database in the default category or the '`au`' category if the exact spelling is known. If the name is misspelled, it might be found using the query '`au=(soundex` *misspelled-name*`)`'. See Appendix A [Sample Format], page 37, Appendix B [Query Syntax], page 41.

4.3 The Index Files

`waisindex` generates a number of index files. If called with '`waisindex -d test -t fields ...`' it uses:

'`test.fmt`'
: The format definition. See Section 4.2 [Format Description], page 16. This file is not necessary if the option '`-t fields`' is not used.

'`test.fde`'
: The *optional* format description. Plain text, which is added to the database description.

'`test.syn`'
: The *optional* synonym file contains multiple lines with synonym terms separated by spaces.

```
wais freewais
programming hacking
```

'`test.stop`'
: The *optional* stopword file contains words which should be ignored when indexing, each in one line.

The following files are generated or modified by `waisindex`:

'`test.src`'
: The database description. See Section 4.3.1 [Database Description], page 20.

'`test.fn`' The filename list. One entry for each file in the database. See Section 4.3.2 [Filename List], page 22.

'`test.hl`' The headline list. One entry for each document in the database. See Section 4.3.3 [Headline List], page 22.

'`test.doc`'
Document table. One entry for each document in the database. Contains pointers to the filename list and the headline list. See Section 4.3.4 [Document Table], page 22.

'`test.cat`'
The catalog file. One entry for each document in the database. A human readable combination of document table with headline list and filename list.This file may be very space consuming. You can avoid generating this file if you use `waisindex` with the '`-nocat`' option. See Section 4.1 [waisindex Options], page 15. See Section 4.3.5 [Catalog Files], page 23.

'`test.dct`'
The global dictionary. One entry for each term in the default field. See Section 4.3.6 [Dictionary Files], page 23

'`test.inv`'
The inverted file for the default field. For each term in the database, there is a list of *postings* giving the documents and positions in the documents where the term occurs. See Section 4.3.7 [Inverted Files], page 24.

'`test.stop`'
`waisindex` might add words to the stopword file, if they occur to often and would break the index.

For each field in the format description a dictionary and a inverted file is generated. This applies only to `waisindex` run with the '`-t fields`' option.

'`test_field_`*name*`.dct`'

'`test_field_`*name*`.inv`'

4.3.1 The Database Description

The database description ('*database*`.src`') serves for two purposes:

- The description is used by some clients to extract connection information. This might be hostname/ip-address and port or the location of the index files for local searches. See Section 3.2.2.5 [LOCAL_SEARCH], page 8.
- A number of descriptions might be used to make up a database of descriptions also referenced to as *directory of servers*. Use '`waisindex -t server ...`' to build such a directory.

This is how a database descriptions looks like (when called without the '`-export`' option):

```
(:source
   :version  3
   :database-name "/usr/local/ls6/src+data/src/freeWAIS-sf-2.0/FIELD-EXAMPLE/test"
   :cost 0.00
   :cost-unit :free
```

```
    :maintainer "pfeifer@buster"
    :keyword-list (
                   fuhr
                   pfeifer
                   )
    :fields (
                   (:field
                    :name "ck"
                    :description "Citation Code")
                   (:field
                    :name "py"
                    :description "Publication Year (numeric)")
                   (:field
                    :name "au"
                    :description "Author")
                   (:field
                    :name "ti"
                    :description "Title")
                   (:field
                    :name "jt"
                    :description "Journal Title")
                   (:field
                    :name "ed"
                    :description "Date of insertion (date)")
            )
    :description "Server created with freeWAIS-sf 2.0 PL 30 on Oct  6 18:09:24 1995 by pfeifer@buster
 The files of type fields used in the index were:
    /usr/local/ls6/src+data/src/freeWAIS-sf-2.0/FIELD-EXAMPLE/TEST
 Here goes the contents of the format description file (database.fde)
 "
 )
```

With the '-export' option the first part changes to:

```
 (:source
    :version  3
    :ip-address "129.217.20.190"
    :ip-name "buster"
    :tcp-port 210
    :database-name "test"
 ...
```

You can modify the database description, especially if you want to enter it in a directory of servers. Notice that waisindex might overwrite the description when re-indexing or adding to the database when the list of significant terms (keywords) changes. Therefore, use the format description file (*database.fde*) to add your comments on the database.

4.3.2 The Filename List

The filename list ('*database*.fn') contains for each source file of the database its filename, modification time, and the type of the documents.

The files starts with four zero bytes. Then for each file follows it's name as zero terminated string, its modification time a four byte integer and the display type as zero terminated string.

Byte	Contents
0-3	0x00000000
4-52	'/usr/local/ls6/src/freeWAIS-sf/FIELD-EXAMPLE/TEST'
53	0x00
54-57	0x2EEC461D (-> *13:46 Dec 12 1994*)
58-61	'TEXT'
62	0x00

4.3.3 The Headline List

The headline list contains the headline; a null terminated string for each document of the database . The first 4 for bytes are zero.

Byte	Contents
0-3	0x00000000
4-73	' 1990 Ait-Kaci, Hassan; Na Implementing a Knowledge-Based Library I'
74	0x00
74-144	' 1990 Shepard, Michael A.; Transient Hypergraphs for Citation Netwo'
145	0x00
...	

The length of the headlines is variable with an upper limit given by the macro MAX_HEADLINE_LEN in 'config.h'.

int **MAX_HEADLINE_LEN** 'config.h'

Maximum length of a headline. It may cause difficulties when increasing the value above the default value of 300. Some of the problems are fixed; maybe not all ;-)

4.3.4 The Document Table

The document table contains one entry for each document in the database. Each entry contains a pointer to the filename list and the headline list. Also start and end of the document in the file is given. The other entries are the number of terms and lines and the date extracted from the document (see Section 4.2.2.2 [Layout Section], page 17).

Byte	Contents
0-1	0x0000
2-4	Three byte offset in the filename list.
5-7	Three byte offset in the headline list. This entry may be four bytes if you selected the `HUGE_HEADLINES` option during configuration. See Section 3.2.2.2 [Huge Headlines], page 7.
8-11	The first character of the document in the source file (four bytes).
12-15	The last character of the document in the source file (four bytes).
16-19	Number of term in the document (four bytes).
20-22	Number of lines in the document (three bytes).
23-26	Date extracted from the document (yymmdd: a long int year*10000+month*100+day).
27-29	Next document: Three byte offset in the filename list.
...	

The first 25 bytes entry might be zero filled as well. Therefor the entry for document *n* starts at *n**25+2.

4.3.5 The Catalog Files

Unless prohibited by the '`-nocat`' option (see Section 4.1 [waisindex Options], page 15), `waisindex` generates a human readable version of the document table (see Section 4.3.4 [Document Table], page 22) with the filenames and headline included from the filename list (see Section 4.3.2 [Filename List], page 22) and the headline list (see Section 4.3.3 [Headline List], page 22).

```
Catalog for database: /usr/local/ls6/src/freeWAIS-sf/FIELD-EXAMPLE/test
Date: Jun  4 13:32:46 1995
276 total documents

Document # 1
Headline:  1990   Ait-Kaci, Hassan; Na  Implementing a Knowledge-Based Library I
DocID: 0 244 /usr/local/ls6/src/freeWAIS-sf/FIELD-EXAMPLE/TEST

Document # 2
Headline:  1990   Shepard, Michael A.;  Transient Hypergraphs for Citation Netwo
DocID: 244 437 /usr/local/ls6/src/freeWAIS-sf/FIELD-EXAMPLE/TEST

...
```

4.3.6 The Dictionary Files

The dictionary file ('*database*`.dct`', '*database*`_field`*name*`.dct`' contains entries for all valid terms in the database. An entry contains:

Bytes	Contents
21	The term as a zero terminated string. Therefor the length of the terms is 20 chars maximum.
4	Pointer in the inverted file. See Section 4.3.7 [Inverted Files], page 24
4	Should be the number of documents the term occurs in. Is the total number of occurances instead.

The entries are sorted by the terms and grouped in blocks of 1000 entries each. This main file is preceeded by a directory containing the first entries of the main blocks. The bytes 21-24 of this directory entries contain the pointer to the main blocks, the last four bytes are zero. Entries in the index block are sorted too so that binary search on the index can work. The first two bytes of the file give the number of main blocks used. E.g., if we have two blocks the layout is as follows:

Bytes	Contents
0-1	0x0000
2-3	2
4-32	Directory entry for the first block (21 Bytes: Term, 4 Bytes pointer, 4 Bytes zero)
33-61	Directory entry for the second block
62-29061	First block
29062-58062	Second block

4.3.7 The Inverted Files

The inverted files are the heart of the index. They contain the references to the documents they occur in for each term. The first 4 bytes contain the number of all terms in the file. The rest of the file is made up of term entries. Note that the following is mainly *debugger knowledge*; Tungs documentation is not very good here. But a Perl script using this definition was able to parse my inverted files ;-).

Header1

Bytes	Contents
1	'`{`', the dictionary flag.
6	0x000000000000. Probably used during building?
2	Length of header 1
4	Total number of occurances of the term. Multiple occurances in documents count!
variable	The term terminated by a newline. *Variatio delectat.*

Header 2 This is where the entries in the dictionary point to. See Section 4.3.6 [Dictionary Files], page 23

Bytes	Contents
1	'`E`' the full flag, indicating a vaid entry.

4	Number of postings following this header.
4	Total size of the postings following this header.

Postings A list of postings, each referring to one document.

Bytes	Contents
4	The document id. This is the entry number in the document table. See Section 4.3.4 [Document Table], page 22.
4	The total size of the character position list including the first one(!). If you use the proximity operators (see Section 3.2.2.3 [Proximity Search], page 7) the word position is stored instead. This also applies to the rest of the posting list.
4	A single precision float which gives the weight of the term with respect to the document. See Section 5.2 [Query Weighting], page 28.
3	The position of the first occurance of the term in the document.
variable	If there is more than one occurance of the term in the document it will be followed by a list of *compressed integers*. From each byte bits 6 to 0 are used. If 7 bit are not enough to store the number, bit 7 of the higher order bytes is set. So 0 to 127 is encoded as usual and 128 is encoded as 0x8100 129 as 0x8101 an so on.

5 Queries

5.1 Formulating a Query

As mentioned in Section 1.1 [Protocol], page 1, freeWAIS-sf uses the original WAIS Protocol. This protocol was designed just for transporting a *free text query* which means a list of search term separated by spaces. We decided deliberately to use this old protocol so that all clients out there could use our new features.

So we had to encode the new and richer query semantics in the query string. This means that the query now has to obey a certain syntax (see Appendix B [Query Syntax], page 41) and consequently a user might get a syntax error when submitting a query. So our goal was to make the syntax as easy as possible and especially leave simple free text queries valid.

In the query the categories to be searched should be selectable for each term. To leave the original queries valid (and to support casual users) we provided a default category, which is used if no category is specified in the query. Now we give an outline of the query language:

The atomic search expressions of the language are terms, term wild-cards and phrases (e.g. '`information`', '`inform*`', '`"information retrieval"`').

Stemming is handled transparently for the client. Terms searched in a stemmed category are searched using their word stem automatically. For a wildcard (only tail truncation is implemented), all matching words from the dictionary are used as search terms. Phrase search looks up the words in the string. At least one of them must be an index term. Then the server scans the documents containing this word for string matches for the complete phrase. This means that string search can only work if the server has access to the documents. For type '`URL`' this is not the case.

Additionally the prefix operators '`soundex`' and '`phonix`' are allowed for converting the following query term into its Soundex/Phonix code. This is for example very useful when searching in phonebooks if the exact spelling of a name is not known. Arbitrary Boolean combination of these atomic expressions with the *binary* operators '`and`', '`or`' and '`not`' ('`not`' means '`and not`' in Boolean logic and is therefore a binary operation, too. See the examples below.) are allowed. Parentheses can be used for grouping. For compatibility with the original syntax, '`or`' may be omitted.

For each expression, a semantic category (field) can be defined using the '*category pred*' operator, where *pred* is '`=`' for text categories and one of '`=`', '`<`', '`>`' for numeric categories ('`==`' is also valid for backward compatibility to version 1.1).

Here are some examples:

'`information retrieval`'
: free text query

'`information or retrieval`'
: same as above

'`ti=information retrieval`'
: '`information`' must be in the title

'`ti=(information retrieval)`'
: one of them in title

'`ti=(information or retrieval)`'
: same as above

'`ti=(information and retrieval)`'
: both of them in title

'`ti=(information not retrieval)`'
: '`information`' in title and '`retrieval`' not in title

'`py==1990`'

'`py=1990`' numerically equal

'`py<1990`' numerically less

'`py>1990`' numerically greater

'`ed<19930101`'
: Date search. Format is yyyymmdd

'`au=(soundex salatan)`'
: soundex search, matches eg. '`Salton`'

'`ti=("information retrieval")`'
: phrase search

'`ti=(information system*)`'
: wild-card search

'`nuclear w/10 waste`'
: We have added a proximity search feature to freeWAIS-sf. With this feature, you could search for '`nuclear w/10 waste`' (like the Lexis/Nexis search syntax) to find all stories that have '`nuclear`' and '`waste`' within 10 words of each other.

'`nuclear pre/10 waste`'
: Proximity Searches

  If the order of the two words is important, then '`nuclear pre/10 waste`' will find all stories that have '`nuclear`' up to 10 words before '`waste`'. Proximity also works within fields; for instance, '`byline=(dan w/2 woods)`' will find every story with a byline that has '`dan`' within 2 words of '`woods`'. Note that you must use parentheses around the words you want to look for in the field.

'`atleast 20 clinton`'
: *At Least* Searches

  '`atleast 20 clinton`' finds every story that has at least 20 occurrences of '`clinton`'. '`atleast`' has to be all lower-case, and there cannot be any spaces between '`at`', '`least`', and the number – '`at least 20 clinton`' will not work!

5.2 Query Weighting

Let's for now disregard the boolean operators and assume that a query is simply a list of terms. Query weighting is done using the *Vector Space* model. Each term in the query is associated with a *query term weight*. Currently this weight is constantly 1. On the other side, the terms in each document get a *document term weight*. This weight is the product of a document specific weight and the *inverse document frequency*. The latter is defined as '`idf = log(`N`/`n`)`' where N is the number of documents in the database and n the number of documents the term occurs in.

The other part of the document weight is computed as follows: Let *tf* be the number of occurances of the term in document and *maxtf* the maximum frequency of any term in the document. A preliminary weight is computed according to '`w = (0.5 * tf)/(1 + maxtf)`'. Then these weights are normalize by dividing them by the sum of the squares of all preliminary weights for terms in this document. So the document specific weights make up a vector of length 1. The final document term weight is yielded by multiplying this weight to the *idf*.

For simple queries (no booleans) the weight of a document is computed by multiplying the query term weight to the query term weight for each term in the query and summing up the results. This is often referred to as the vector product (hence the name of the model) or scalar product.

Now let's get to the booleans. The '`or`' operator is just dropped. So '`information or retrieval`' yields exactly the same weight than '`information retrieval`'. To interpret the vector product in another way, you can say the '`or`' operator just sums up the weights of its arguments. For the '`and`' operator the weights of both arguments is computed and the final weight is just the minimum of these weights. Similar the *binary* '`not`' operator returns the minimum of the weight of the left argument and 1 - the weight of the right argument.

Remarks

Numeric searches are currently handled the same way. This is clearly a bug. Why should a document from 1988 yield a higher weight with respect to a query '`<=1990`' than one from 1987 just because there are more documents from 1987 in the database than from 1988?

I have no idea how strings searches are weighted.

Partial searches hopefully are weighted as if all matching terms where given.

The proximity operators might work like the booleans?

6 Linking freeWAIS-sf with other Systems

6.1 WWW Clients

Linking was tested with:

- NCSA Mosaic 2.4
- Tuebingen Univ Mosaic 2.4.2

6.2 WWW Server

By Jean-Philippe Martin-Flatin <*syj@ecmwf.int*>

Linking was tested with:

- CERN httpd 3.0

Direct WAIS access for CERN httpd 3.0 is easy to provide with freeWAIS-sf-2.0. The only thing you need to do is rename the WAIS libraries in CERN httpd Makefile's.

If you're a lucky guy and your system supports imake, you need to:

- Retrieve Rainer Klute's Imake extension to CERN httpd 3.0. See Section 2.3 [Sources], page 4.
- Replace '`WWW.cf`' in the top-level directory with the code in the Appendix (see Appendix D [freeWAIS-sf and CERN httpd], page 45)
- Update the location of your freeWAIS-sf code in '`WWW.cf`' (variable `WAISDIR`)

If your system doesn't support imake, you need to update manually the WAIS libraries in the Makefile pertaining to your architecture and the top-level Makefile. And think about how easy life would be if only you had imake.

6.3 Gopher 2.1.1

By Steve Hsieh, originaly written for freeWAIS-sf-1.0

In '`gopher2_1_1/gopherd/Makefile`':

replaced original `SFWAISOBJ` with:

linux

```
SFWAISOBJ       = ../regexp/libregexp.a ../ir/libinv.a ../ir/libclient.a \
                  ../ir/libwais.a ../ir/liblocal.a ../ir/libsig.a \
                  ../ui/source.o ../lib/libftw.a
```

solaris & SunOS

```
SFWAISOBJ       = ../ir/libinv.a ../ir/libclient.a ../ir/libwais.a \
                  ../ir/liblocal.a ../ir/libsig.a ../ui/source.o \
                  ../regexp/libregexp.a ../lib/libftw.a
```

In '`gopher2_1_1/gopherd/waisgopher.c`': change

```
MIN(
```

to

```
MINIMUM(
```

Run `Configure` in freeWAIS-sf. See Section 3.2 [Configure], page 6.

For the following configure questions, '`required`' means that I had to use that value to get gopher to work with freewais-sf. '`doesn't matter`' means that the decision is up to you ....

```
Do you want to use your systems regexp.h (no)?  no <-- required
Will you have HEADLINE files greater than 16 MB (no)?   no <-- doesn't matter
Use your systems ctype (no)?    yes <-- required
Do you want to compile with -DLOCAL_SEARCH (yes)?       yes <-- required
Do you want to use the modified URL handling (no)?      no <-- doesn't matter
Where should the installation go (/usr/local/wais)? (specify your own path)
Do you want to use shm cache (no)?      no <-- doesn't matter
Disable the UDP packet sending (no)?    no <-- doesn't matter
```

Make freewais-sf ... create symbolic links to in gopher2_1_1 to the appropriate freewais-sf directories ...

In '`gopher2_1_1`':

```
ln -s ../freeWAIS-sf-1.0/ir
ln -s ../freeWAIS-sf-1.0/ui
ln -s ../freeWAIS-sf-1.0/lib
ln -s ../freeWAIS-sf-1.0/regexp
```

Edit '`gopher2_1_1/Makefile.conf`' '`gopher2_1_1/conf.h`' as necessary for your system and site.

Make sure to uncomment `-DFREEWAIS_SF` in '`Makefile.conf`' !

make gopher ...

In the case of an '`index type not recognized`' error, test an index on an database that has been reindexed using the newly compiled `waisindex` in '`freeWAIS-sf-2.0/ir`' (as opposed to gopherindex) just to make sure that there really still is a problem...

Use '`waisindex -export ...`' It is important that the host and port information is compiled into the '*database*`.src`' file, **even if the searches are local!** I have my gopher, wais , and all the databases all on the same machine. It is also important that the `waisserver` is running.

7 Discussion of freeWAIS-sf

7.1 Newsgroup comp.infosystems.wais

Discussion of freeWAIS-sf mostly happens in '`comp.infosystems.wais`'. Please make sure that you have the term '`freeWAIS-sf`' somewhere in the subject line and the version you are referring to somewhere in the body. Please do not send mails to me. I will answer postings in c.i.w if they are of general interest - and I have the time. Do not reply to my postings by mail. Keep the readers of c.i.w informed by posting there.

7.2 freeWAIS-sf Mailing-list

There is a mailing-list about freeWAIS-sf, which is intended for beta testers. Messages to '`wais@wsct.wsc.com`' will be forwarded to the subscribers. To join the mailing-list send a mail to '`majordomo@wsct.wsc.com`':

```
To: majordomo@wsct.wsc.com

subscribe wais
```

To leave the mailing-list send the same mail with '`subscribe`' replaced by '`unsubscribe`'.

7.3 Bug Reports

Send bug reports not referring the last version to the newsgroup (see Section 7.1 [Newsgroup comp.infosystems.wais], page 35).

If you are sure that you use the latest version, try to submit patches rather than bug reports. See Section 7.4 [Submitting Patches], page 36.

If you cannot solve the problem, send the bug report to the mailing-list. See Section 7.2 [freeWAIS-sf Mailing-list], page 35.

Include if possible:

- Description of your hardware and operating system
- Description how to reproduce the bug – preferably with the included test database.
- Debugger back-trace if the system crashes or of description of the misbehaving of the wais program if it just works not as it should.
- The configuration options you chosed. See Section 3.2 [Configure], page 6.

Remember that clients are not the focus of freeWAIS-sf. So definitely we are only interested in patches for problems with the clients.

7.4 Submitting Patches

If you submit patches, make sure that they apply to the latest version. If you have patches for old version, please verify that the problem persists with the latest version and try you path on it. Otherwise send the patch to the newsgroup (see Section 7.1 [Newsgroup comp.infosystems.wais], page 35).

Patches for the newest version should be send to the mailing-list. See Section 7.2 [freeWAIS-sf Mailing-list], page 35.

Try to send in context diffs like the ones generated by '`diff -c`'. Try to restrain from reformatting and indenting the source before generating the patch to restrict the differences to the real changes. Please also include diffs to the '`Changelog`' in each directory of the distribution. Add a description of your patch stating the problem, the solution, and the names of the functions you changed.

Appendix A Sample Format Description

Sample document:

```
CK: Frei/Schauble:91
AU: Frei, H.P.; Schauble, P.
TI: Determining the Effectiveness of Retrieval Algorithms.
JT: Information processing & management.
PP: 153
^L
CK: Schauble:89
AU: Schauble, Peter
TI: Improving the effectiveness of retrieval systems by information
    structures.
JT: Information processing & management.
VO: 25
PY: 1989
NO: 4
PP: 363
```

Sample format description:

```
#                               -*- Mode: Sh -*-
# test.fmt --
# Author          : Ulrich Pfeifer
# Created On      : Fri Oct  6 13:27:35 1995
# Last Modified By: Ulrich Pfeifer
# Last Modified On: Fri Nov  3 10:14:12 1995
# Language        : FMW
# Update Count    : 23
# Status          : Unknown, Use with caution!
#
# (C) Copyright 1995, Universitt Dortmund, all rights reserved.
#
# Remember: indexing and matching is done line by line. Each regular
# expression can match one line maximum.

# This regular expression should match the line separating documents.
# The matching line is NOT indexed!  Use /\n\n/ if each file contains
# only one record.

record-sep: /\f/    # formfeed (alias for \L)

# Now we specify how the headline is constructed.  The date is part of
# the headline and not searchable.
layout:

# First me map the regions starting with /^PY: / up to (and excluding)
# the next /^[A-Z][A-Z]:/ skipping everything from the beginning up after
# the end of /^PY: */ to the first 4 characters of the headline.
```

```
  headline: /^PY: / /^[A-Z][A-Z]:/ 4 /^PY: */
  headline: /^AU: / /^[A-Z][A-Z]:/ 20 /AU: */
  headline: /^TI: / /^[A-Z][A-Z]:/ 40 /TI: */

# The date is found on lines matching /^ED: /. The line is kipped up
# to the end of /^ED: *./. The line is handled to scanf with format
# string "%d.%d.%d". Day, month, and year are expected in this
# order. Month may be a string containing the 3 letter abreviation for
# the name of the month. In this case the keyword "string" must follow
# "month"

  date: /^ED: / /%d.%d.%d/ day month year /^ED: *./

# Now we define the first region. It starts at /^CK:/. everything up
# after the end of /^CK: */ is skipped. Region ends at
# /^[A-Z][A-Z]:/. The match for end is search BEHIND the match for the
# begin.
region: /^CK:/ /^CK: */

# All words are transfered in the 'ck' index and the global
# index. Description of 'ck' is overwritten by "Citation Code".

ck "Citation Code" TEXT BOTH
end: /^[A-Z][A-Z]:/

# All words in this region will be numeric. They are merge in the 'py'
# index which will be tagged as nummeric from here on (thats not how
# it should be - I know). Make sure that you do not pollute 'ck' in
# the following with non-numeric data.  The "numeric" keyword has it's
# own skip expression /^PY: [^ ]/ which overwrites the region skip
# expression. This way you can index the same region nonnumeric (This
# not ...).

region: /^PY: / /^PY: */
py "Publication Year" numeric: /^PY: */ 4 TEXT LOCAL
end: /^[A-Z][A-Z]:/

# This region will be indexed plain for the global index and the 'au'
# index. The 'au' index will also contain the soundex codes for the
# words.

region: /^AU: / /^AU: */
au "Author" SOUNDEX LOCAL TEXT BOTH
end: /^[A-Z][A-Z]:/

# All words in theis reagion will be stemmend before insertion in the
# global and the 'ti' index'

region: /^TI: / /^TI: */
ti "Title" stemming TEXT BOTH
```

```
end: /^[A-Z][A-Z]:/

# This region is indexed for multiple local and the global index.
# Stemming applies for the 'ti' index and the global index.

region: /^JT: / /^JT: */
jt "Journal Title" ti stemming TEXT BOTH
end: /^[A-Z][A-Z]:/

# Here we make the data searchable. Format is merely the same as in
# the layout section. There is no local skip expression. That wasnt a
# good idea for numerics anyway. The date can be search with the
# yymmdd format. See test DATE.

region: /^ED: / /^ED: */
ed "Date of insertion" <date> /%d.%d.%d/ day month year TEXT LOCAL
end: /$/

region: /^JT: / /^JT: */
TEXT GLOBAL
end: /^[A-Z][A-Z]:/
```

Appendix B Query Syntax

```
expression      : term
                | expression or term
                ;
or              : %prec 'or'
                | 'or'
                ;
and             : 'and'
                | 'not'
                ;
term            : factor
                | term and factor
                ;
factor          : unit
                | unit 'pre/' unit
                | unit 'w/' unit
                | 'atleast/' unit
                ;
unit            : word-or-literal
                | phonsound word-or-literal
                | '(' expression ')'
                | word-or-literal  '=' '(' s_expression ')'
                | word-or-literal  '='  word-or-literal
                | word-or-literal  '='  phonsound word-or-literal
                | word-or-literal relop word-or-literal
                ;
relop           : '=' '='
                | '<'
                | '>'
                ;
phonsound       : PHONIX
                | SOUNDEX
                ;
s_expression    : s_term
                | s_expression or s_term
                ;
s_term          : s_factor
                | s_term and s_factor
                ;
s_factor        : s_unit
                | s_unit 'pre/' s_unit
                | s_unit 'w/' s_unit
                | 'atleast/' s_unit
                ;
s_unit          : word-or-literal
                | phonsound word-or-literal
                | '(' s_expression ')'
                ;
```

Appendix C Format Specification Syntax

```
docspec         : parseformat | format
                ;
parseformat     : record layout
                ;
format          : record layout speclist
                ;
record          : ‘record-sep:’ REGEXP
                ;
layout          :
                | ‘layout:’ layoutspecs ‘end:’
                ;
layoutspecs     : layoutspec
                | layoutspec layoutspecs
                ;
layoutspec      : ‘date:’ REGEXP REGEXP date date date regexp
                | ‘headline:’ REGEXP REGEXP INT regexp
                ;
speclist        : spec
                | spec speclist
                ;
spec            : ‘region:’ REGEXP regexp
                  fieldlist
                  options
                  indexspecs
                  ‘end:’ REGEXP
                ;
options         :
                | option options
                ;
option          : ‘numeric:’ regexp INT
                | ‘date:’ date date date
                | ‘stemming’
                ;
regexp          :
                | REGEXP
                ;
indexspecs      :
                | indexspec indexspecs
                ;
indexspec       : indextype dicts
                ;
indextype       : ‘TEXT’
                | ‘SOUNDEX’
                | ‘PHONIX’
                ;
dicts           : ‘GLOBAL’
                | ‘LOCAL’
                | ‘BOTH’
                ;
```

```
date            : `day'
                | `month' monthspec
                | `year'
                ;
monthspec       :
                | `string'
                ;
fieldlist       :
                | WORD descr fieldlist
                ;
descr           :
                | `"' any string `"'
                ;
```

Appendix D freeWAIS-sf and CERN httpd

```
/* Version numbers of library, daemon, and client: */
#define LibwwwVersion 2.17
#define DaemonVersion 3.0


/* Where to install binaries? */
BINDIR = /usr/local/infosystems/WWW/bin


/* Do you want to compile the WAIS gateway? If yes, where is freeWAIS
   located?
*/
#define FreeWAIS_sf YES
#define FreeWAIS_02 NO
#define FreeWAIS_03 NO
#define FreeWais (FreeWAIS_02 || FreeWAIS_03)

#if FreeWais
/* WAISDIR = /usr/local/infosystems/freeWAIS-0.2 */
WAISDIR = /usr/local/infosystems/freeWAIS-0.3
WAISLIBDIR = $(WAISDIR)/bin
WAISINCDIR = $(WAISDIR)/ir
#endif

#if FreeWAIS_sf
WAISDIR = /usr/local/infosystems/freeWAIS-sf-1.0
WAISLIBDIR = $(WAISDIR)/ir
WAISINCDIR = $(WAISDIR)/ir
#endif

/*
 * If you don't want to use GCC for compilation, but the compiler defaulted in
 * Imake's configuration files, set UseGcc to NO. If you want to change the
 * native's compiler default options, set CCOPTIONS and/or CDEBUGFLAGS as
 * needed, preferrably depending on the architectur as show below.
 */

#define UseGcc YES

#if UseGcc   /* use GNU C compiler: */

         CC = gcc -fpcc-struct-return
  CCOPTIONS =

#if defined (SunArchitecture) && !defined (i386Architecture) && !defined
(SparcArchitecture)
  CCOPTIONS = -m68881
#endif /* Sun-3 */
```

```
#else   /* use native C compiler: */

#if defined (HPArchitecture) && defined (hp9000s800)
CCOPTIONS = -Aa -D_HPUX_SOURCE -Wl,+b $(USRLIBDIR) -Wl,+s
#endif /* HPArchitecture */

#if defined (XyzArchitecture) /* e. g. IBMArchitecture, SGIArchitecture etc. */
         CC = compiler-to-use
  CCOPTIONS = compiler-options
#endif /* XyzArchitecture */

#endif /* UseGcc */

/* No need to modify anything below here */

 LIBWWWDIR = $(TOP)/Library/Implementation
    LIBWWW = -L$(LIBWWWDIR) -lwww
 DEPLIBWWW = $(LIBWWWDIR)/libwww.a

#if FreeWais
   LIBWAIS = $(WAISLIBDIR)/client.a $(WAISLIBDIR)/wais.a
DEPLIBWAIS = $(LIBWAIS)
#endif

#if FreeWaisSf
   LIBWAIS = $(WAISLIBDIR)/libclient.a $(WAISLIBDIR)/libwais.a
DEPLIBWAIS = $(LIBWAIS)
#endif

/* Imake rule to create a generic name as a link */
#ifndef InstallAsLink
#define InstallAsLink(file,link,dir) @\
install:: file                       @\
        set -x; cd dir; \            @\
        $(RM) link; \                @\
        $(LN) file link
#endif
```

Index

A

Architecture 1
autoconf 6

C

catalog 23
Character Set 8
Client Server Architecture 1
Compiling 10
config.status 6
Configuration files 6
configure 6
Configure Questions 6
CPP macros 6
Customization 6

D

Default port 13
Directory layout 5
Document separation 17
Document term weight 28

E

Emacs 11

F

Fields, definition 18
fmt files 16
Format Description 16
Format descriptions, generating 17

H

Headlines 17
HTML 9

I

idf 28
Indexing types 18
inetd 13
Installation 11

L

LCHARS 8
LOCAL_SEARCH 9

M

Manual 11
MAX_HEADLINE_LEN 22
maxtf 28
Mirror Site, become one 3
MYCTYPE 8
MYREGEXP 7

O

Option at configure time 6

P

Posting 24
Protocol, supported 1
PROXIMITY 8
Proximity Search 7

Q

Query term weight 28

S

Services 13
Size, headlines 7
Stand-alone clients 8
Stopwords 15
String Search 7

T

Tests 10
tf 28

U

UCHARS 8
Unpacking 5
URL 9
URLDOCID 9

V

Vector space model 28

W

WAIS 1
waisindex 1, 15
Waisindex options 15
Weighting 28
WWW 9

X

X11, application resources 11

Z

Z39.50 1

Table of Contents

1 Overview 1
- 1.1 The WAIS Protocol 1
- 1.2 Client Server Architecture 1

2 Getting freeWAIS-sf and related stuff 3
- 2.1 Mirror Sites 3
- 2.2 Documentation 3
- 2.3 Sources 4

3 Building and Installation 5
- 3.1 Unpacking the Distribution 5
- 3.2 Running `Configure` 6
  - 3.2.1 Files generated 6
  - 3.2.2 Questions 6
    - 3.2.2.1 Regular Expressions 7
    - 3.2.2.2 Huge Headlines 7
    - 3.2.2.3 Proximity vs. String Search 7
    - 3.2.2.4 Character Set 8
    - 3.2.2.5 `LOCAL_SEARCH` 8
    - 3.2.2.6 Modified URL handling 9
    - 3.2.2.7 Installation Prefix 9
    - 3.2.2.8 Synonym Files 9
    - 3.2.2.9 Registration 10
- 3.3 Compiling the system and running the tests 10
  - 3.3.1 Running the tests 10
- 3.4 Installation 11
  - 3.4.1 Installation Directories and Files 11
  - 3.4.2 Installing the Server 13

4 Building Databases 15
- 4.1 `waisindex` Options 15
- 4.2 Bulding a Format Description 16
  - 4.2.1 Regular Expressions 16
  - 4.2.2 Structure of the Format Descriptions 17
    - 4.2.2.1 Separation of Documents 17
    - 4.2.2.2 Layout Section 17
    - 4.2.2.3 Definition of Field and Index Types 18
- 4.3 The Index Files 19
  - 4.3.1 The Database Description 20
  - 4.3.2 The Filename List 22
  - 4.3.3 The Headline List 22
  - 4.3.4 The Document Table 22
  - 4.3.5 The Catalog Files 23
  - 4.3.6 The Dictionary Files 23
  - 4.3.7 The Inverted Files 24

5 Queries 27
- 5.1 Formulating a Query 27
- 5.2 Query Weighting 28

Remarks 29

6 Linking freeWAIS-sf with other Systems 31

6.1 WWW Clients 31
6.2 WWW Server 31
6.3 Gopher 2.1.1 31

7 Discussion of freeWAIS-sf 35

7.1 Newsgroup comp.infosystems.wais 35
7.2 freeWAIS-sf Mailing-list 35
7.3 Bug Reports 35
7.4 Submitting Patches 36

Appendix A Sample Format Description 37

Appendix B Query Syntax 41

Appendix C Format Specification Syntax 43

Appendix D freeWAIS-sf and CERN httpd 45

Index 47