FreeWAIS-sf*

Ulrich Pfeifer
Tung Huynh‡
University of Dortmund
Lehrstuhl Informatik VI
D-44221 Dortmund

March 30, 1995

Abstract

FreeWAIS-sf[1] is an extension of the freeWAIS software provided by the the Clearinghouse for Networked Information Discovery and Retrieval (CNIDR)[2]. The SF suffix in the software name stands for "structured[3] fields," an indexing and search feature which distinguishes this software from its predecessors.
It is based on the version 0.202[4] of this software but includes and enhances much of what freeWAIS-0.3[5] contains.
Major extensions of FreeWAIS-sf include:

- **Introduction of text, date, and numeric field structures within a document, which allows a document to be indexed using potentially overlapping fields.**
- **Support for complex Boolean searches (a query parser is integrated in the server).**
- **Stemming and phonetic coding may be switched on and off for each individual field.**
- **Definition of document format and layout of the headlines are now configurable by a new specification language based on regular expressions. No C-code must be written to index new document types.**
- **Installation procedure now just requires running a `sh` script and answering simple questions. The script is generated using the GNU autoconf[6] utility. No Makefile sets for individual systems are necessary. For development purposes, additional Imakefiles are provided since the Makefiles do not contain dependencies**
- **Support for country specific character sets (8-Bit)**
- **Lots of bug fixes.**

All changes are restricted to the indexer and server to allow existing clients to query FreeWAIS-sf databases. Document types contained in the original distribution remain intact. You can use FreeWAIS-sf as you would use original freeWAIS or take advantage of its enhanced features.

*You can get a Postscript Version† of this document.
‡email: {pfeifer,huynh1}@ls6.informatik.uni-dortmund.de
[1] ftp://ls6-www.informatik.uni-dortmund.de/pub/wais/freeWAIS-sf-1.1/freeWAIS-sf-1.1.tgz
[2] http://cnidr.org/welcome.html
[3] Histoically the "s" meant "soundex"
[4] ftp://nic.switch.ch/mirror/wais/servers/freeWAIS/freeWAIS-0.202.tar.Z
[5] ftp://ftp.bio.indiana.edu/util/wais/freeWAIS-0.3.tar.gz
[6] ftp://hpcsos.col.hp.com/mirrors/.hpib0/gnu/autoconf-1.10.tar.gz

1 Supported Systems

FreeWAIS-sf is known to compile cleanly on many different UNIX platforms, particularly using the GNU C compiler. The known platforms include:

OS	Version	Hardware	Compiler
A/UX	3.1	mc68040	gcc 2.5.7
AIX	2	0000001964	cc
AIX	2	0000008314	cc
AIX	2	0000024535	cc
AIX	2	000003085C	cc
AIX	2	0000052366	cc
AIX	2	0000056118	cc
AIX	2	0000061138	cc
AIX	2	0000080931	cc
AIX	2	0000091446	cc
AIX	2	0000195011	cc
AIX	2	0000201518	cc
AIX	2	0000261834	cc
AIX	2	0000298037	cc
AIX	2	0000334735	cc
AIX	2	0000420476	cc
AIX	2	0000603735	cc
AIX	2	0000610646	cc
AIX	2	0000826246	cc
AIX	2	0000840341	cc
AIX	2	0002048547	cc
AIX	2	0003809731	cc
BSD/386	1.0	i386	gcc
BSD/386	1.1	i386	gcc
FreeBSD	1.1.5(RELE	i386	gcc
FreeBSD	1.1.5.1(RE	i386	gcc
HP-UX	A.09.00	9000/822	gcc 2.5.6
HP-UX	A.09.01	9000/720	gcc 2.5.8
HP-UX	A.09.01	9000/755	gcc 2.5.8
HP-UX	A.09.04	9000/816	gcc 2.5.8
HP-UX	A.09.04	9000/887	gcc 2.3.3
HP-UX	A.09.05	9000/715	cc
IRIX	4.0.5F	IP20	gcc 2.5.2
IRIX	5.2	IP19	cc
IRIX	5.2	IP22	gcc 2.5.8
Linux	1.0.8	i486	gcc 2.5.8
Linux	1.0.9	i486	gcc 2.5.8
Linux	1.1.18	i486	gcc 2.5.8
Linux	1.1.18	i586	gcc 2.5.8
Linux	1.1.33	i486	gcc 2.5.7
Linux	1.1.45	i486	gcc 2.6.0
Linux	1.1.47	i486	gcc 2.5.8
Linux	1.1.49	i486	gcc 2.5.8
Linux	1.1.50	i486	gcc 2.5.8
Linux	1.1.51	i486	gcc 2.5.8
Linux	1.1.52	i486	gcc 2.6.0
Linux	1.1.55	i486	gcc 2.5.8

Linux	1.1.57	i486	gcc 2.5.8
Linux	1.1.59	i486	gcc 2.5.8
Linux	1.1.60	i486	gcc 2.5.8
Linux	1.1.61	i486	gcc 2.5.8
Linux	1.1.62	i486	gcc 2.5.7
Linux	1.1.64	i486	gcc 2.5.8
Linux	1.1.65	i486	gcc 2.5.8
Linux	1.1.70	i486	gcc 2.5.8
OSF1	V2.0	alpha	cc
OSF1	V2.0	alpha	gcc 2.5.4
OSF1	V2.1	alpha	cc
OSF1	V2.1	alpha	gcc 2.5.8
OSF1	V2.1	alpha	gcc 2.6.0
OSF1	V3.0	alpha	cc
SunOS	4.1.1	sun4c	cc
SunOS	4.1.1	sun4c	gcc 2.5.8
SunOS	4.1.1	sun4c	gcc 2.6.1
SunOS	4.1.1-JL	sun4c	gcc 2.6.2
SunOS	4.1.2	sun4	cc
SunOS	4.1.2	sun4	gcc 2.5.8
SunOS	4.1.2	sun4c	cc
SunOS	4.1.2	sun4c	gcc 2.2.2
SunOS	4.1.2	sun4c	gcc 2.4.2
SunOS	4.1.2	sun4c	gcc 2.4.5
SunOS	4.1.2	sun4c	gcc 2.5.8
SunOS	4.1.3	sun4	gcc 2.5.8
SunOS	4.1.3	sun4c	cc
SunOS	4.1.3	sun4c	gcc 2.3.3
SunOS	4.1.3	sun4c	gcc 2.5.5
SunOS	4.1.3	sun4c	gcc 2.5.8
SunOS	4.1.3	sun4m	acc
SunOS	4.1.3	sun4m	cc
SunOS	4.1.3	sun4m	gcc 2.3.2
SunOS	4.1.3	sun4m	gcc 2.3.3
SunOS	4.1.3	sun4m	gcc 2.5.6
SunOS	4.1.3	sun4m	gcc 2.5.7
SunOS	4.1.3	sun4m	gcc 2.5.8
SunOS	4.1.3	sun4m	gcc 2.6.1
SunOS	4.1.3-JL	sun4c	gcc 2.5.8
SunOS	4.1.3C	sun4m	gcc 2.5.8
SunOS	4.1.3_Axil	sun4m	gcc 2.6.0
SunOS	4.1.3_U1	sun4c	gcc 2.4.5
SunOS	4.1.3_U1	sun4c	gcc 2.5.8
SunOS	4.1.3_U1	sun4m	cc
SunOS	4.1.3_U1	sun4m	gcc
SunOS	4.1.3_U1	sun4m	gcc 2.4.2
SunOS	4.1.3_U1	sun4m	gcc 2.5.7
SunOS	4.1.3_U1	sun4m	gcc 2.5.8
SunOS	4.1.3_U1	sun4m	gcc 2.6.0
SunOS	5.2	sun4c	gcc 2.5.6
SunOS	5.2	sun4m	gcc 2.5.6
SunOS	5.3	sun4c	gcc 2.5.6
SunOS	5.3	sun4c	gcc 2.5.8
SunOS	5.3	sun4d	gcc 2.4.5

SunOS	5.3	sun4d	gcc 2.5.6
SunOS	5.3	sun4d	gcc 2.5.7
SunOS	5.3	sun4d	gcc 2.5.8
SunOS	5.3	sun4m	acc
SunOS	5.3	sun4m	gcc
SunOS	5.3	sun4m	gcc 2.4.5
SunOS	5.3	sun4m	gcc 2.5.6
SunOS	5.3	sun4m	gcc 2.5.7
SunOS	5.3	sun4m	gcc 2.5.8
SunOS	5.3	sun4m	gcc 2.6.0
SunOS	5.3	sun4m	gcc 2.6.1
SunOS	5.4	i86pc	gcc 2.5.8
ULTRIX	4.2	RISC	gcc 2.5.6
ULTRIX	4.3	RISC	gcc 2.5.6
ULTRIX	4.3	RISC	gcc 2.5.8
ULTRIX	4.3	RISC	gcc 2.6.1
ULTRIX	4.4	RISC	gcc 2.6.3
dgux	5.4R2.10	AViiON	gcc
dgux	5.4R2.10	AViiON	gcc 2.4.1

This is not an exhaustive list of supported platforms but represents those systems reported to the authors. If you have ported **FreeWAIS-sf** to an additional platform, please provide the name, OS number, and compiler used to the authors for inclusion in updated release notes.

2 History

Development of **FreeWAIS-sf** was begun in Summer 1993 as bug fixes for version 0.202 of the CNIDR distribution. These fixes included boolean operators, partial match search and phonetic indexing. We mailed the fixes to CNIDR but received no acknowledgement. We decided to redesign the server to parse the queries since we felt that boolean operations cannot be performed correctly without ensuring that the query conforms to a syntax. At the same time we felt that adding C-code for indexing new document formats is too much to require of most data or system system administrators. We also saw a need to split up documents into a number of different fields with possibly different indexing methods.
Since feedback from CNIDR was still missing in February 1994, we released our first version called `freeWAIS-0.2-sf-alpha.tar.gz`. This first version used Imakefiles for installation and was successfully compiled on many systems. Due to fixes to installation code and numerous bug fixes 8 subsequent versions (through `freeWAIS-0.2-sf09--alpha.tar.gz`) were released.
Since this last alpha version contained most of the features we want to implement, we generated the first beta version. Many people, (namely running AIX and DG/UX systems) had no working `imake` on their machines, we added a configure script generated by autoconf for the installation procedure. This script generates templates containing system and installation information. Simple makefiles are thereby generated which allow compilation and installation of **FreeWAIS-sf** on a great variety of systems. See the list of supported systems in Section 1. This Makefiles do not contain dependencies of the generated files. To recompile after changes, run `make clean` then `make all` or use Makefiles generated by `imake`. The last beta version was `freeWAIS-0.2-sf-beta-05.tar.gz`[7]
At this time we decided not to wait for CNIDR, which seems mainly concerned with the Z39.50 Version 3[8] definition and differentiate this version from the CNIDR products by

[7] ftp://ls6-www.informatik.uni-dortmund.de/pub/wais/beta-05/freeWAIS-0.2-sf-beta-05.tar.gz
[8] http://www.research.att.com/ wald/z3950.html

dropping the -0.2- in the name.
During July, August and September we removed some bugs in memory handling. Purify is now completely happy with `waisindex`, `waissearch` and `waisserver`. For more information on some minor changes, look at sections 2.3 to 2.9.
A couple of beta testers spent their time for porting to other systems. From all the people helped us with comments, suggestions and patches (63 netters!), we would like to mention the following (which had a really hard time):

Eric Hagberg	<hagberg@med.cornell.edu>
Steve Hsieh <steveh@eecs.umich.edu>	
Douglas D. Nebert	<ddnebert@usgs.gov>
Jean-Philippe Martin-Flatin	<syj@ecmwf.co.uk>

Thank you all!
After that we be released

freeWAIS-sf-1.1

in September 1994.

Releases

2.1 1.1

Patches 1-9 for 1.0 are integrated. Also ctypes are faster now. Added field description to the `*.src` files.

patch001 Scandir

Fixes a problem with waisindex. Indexing a database the first time causes core dump on some systems because the return value of scandir was not checked.

patch002 X11R6

Makes the x client compile with X11R6.

patch003 waisserver

Adds a forgotten fflush which caused problems on some systems.

patch004 xwais

Fixes an "one-off" bug in `qcommands.c`.

patch005 server security

when the server accepted a connection from a client, the `host_name` and `host_address` variables were left as empty strings (and so never matched the entries in the`DATA_SEC` file)! These need to be reset for each new client connection, which is what the patch does.

patch006 long headlines

Fixes a bug regarding long headlines. This bug prevents long headlines from being returned using `waisq`, `waissearch`, etc.

patch007 line numbers for format file parsing

> Due to incompatibilities of flex, the fmt file parser always complains about syntax errors in line 0. This patch fixes this problem - you now will get the real line number, where the error occurred.

patch008 unreadable files

> Indexing with the '-r' option caused core dumps when encountering an unreadble file. Also encountering already indexed files was fatal. The patch solves this. A message is printed in both cases.

patch009 date in headline

> The date format for the headline did not work. This patch solves most of the known problems with this.

2.2 1.0

Jae W. Chang wrote in his article in comp.infosystems.wais[9]: What happens is that scandir is searching for files of the form `field_<db>.<field>`. If it exists, then they are removed since the user specified a new database to be created and new files have to be created by the same name.
This is a bug. The result from scandir should've been checked. If the result is 0 - meaning no files of the above form were found - the matches array is never allocated, BUT the code still dereferences matches as if it were allocated thus seg fault.
Just looking briefly at an Ultrix man page, freeWAIS-sf will bomb on this dec as well at the same spot, so it's not just isolated to a "linux" quirk.
Here's my diff:

```
diff -c -r1.22 field_index.c
*** 1.22        1994/09/07 13:29:22
--- field_index.c       1994/10/05 14:10:26
***************
*** 760,776 ****
    strcpy(path,dir);
    strncat(path,"/",MAX_FILENAME_LEN);

!   scandir(dir, &matches, rmselector, NULL);
!   for(i=0;matches[i];i++) {
!     path[strlen(dir)+1] = '\0';
!     strncat(path,matches[i]->d_name,MAX_FILENAME_LEN);
!     s_free(matches[i]);
!     waislog(WLOG_LOW, WLOG_INFO, "deleting \"%s\"", path);
!     if (unlink(path)) {
!       waislog(WLOG_HIGH, WLOG_ERROR, "unlink failed");
!     }
    }
-   s_free(matches);
    return(i);
  }
```

[9] news:UiYNLx600WB_RYIYUW@andrew.cmu.edu

```
--- 760,777 ----
    strcpy(path,dir);
    strncat(path,"/",MAX_FILENAME_LEN);

!   if ( scandir(dir, &matches, rmselector, NULL) > 0 ) {
!       for(i=0;matches[i];i++) {
!         path[strlen(dir)+1] = '\0';
!         strncat(path,matches[i]->d_name,MAX_FILENAME_LEN);
!         s_free(matches[i]);
!         waislog(WLOG_LOW, WLOG_INFO, "deleting \"%s\"", path);
!         if (unlink(path)) {
!             waislog(WLOG_HIGH, WLOG_ERROR, "unlink failed");
!         }
!       }
!       s_free(matches);
    }
    return(i);
  }
```

2.3 0.9.10

- Support for HPUX_SOURCES added.
- Compiler version is now reorted by udping

2.4 0.9.8

- Patch from Alberto Accomazzi, which causes stopwords to be taken **either** from the internal list **or** from the specified file. Option `-stop /dev/null` will run waisindex without any stopwords.
- Passes `cc` again.

2.5 0.9.7

- Removed the ANSI_LIKE define in Default.tmpl.in, which caused problems in compilation on some platforms. We will postpone the ANSI stuff.
- Added tests for overlapping copies with `bcopy()` and `memcpy()`. If neither bcopy nor memcpy can handle this, a slow but working function in `cutil.c` is used.
- Support for caching the synonyms in shared memory provided by Alberto Accomazzi `<alberto@cfa.harvard.edu>` was added. Here is what he wrote about it:

  Caching is turned on by running waisserver with the flag -cachesyn.

  For those of you who have fairly large synonym files (> 10Kb) and are running the software on a machine that supports shared memory (all the UNIX boxes that I have worked with do now), 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.

  You can find a brief explanation of how caching works in the header of `synonym.c`.

2.6 0.9.6

- Added the headline fix from Marko Niinimaki. Moved all defines to `Defaults.tmpl`. Removed `ir/irlex.h` from dist.
- Clean the "uninitialized Memory Read" bug in waissserver.

2.7 0.9.5

- Changed numbering for versions (to make jp happy)
- Changed configure code for -lsocket and -lnsl
- Fixed the TELL_USER code again to conform to ANSI
- Added install.lib target for the Makefiles. (Only with Imake)
- Some additions to documentation
- Files on waisindex command line may now have extension `".gz"`.

2.8 0.94

- Some little fixes to make purify happy.
- Fixed the keyword code.
- `config.h` is not in the distribution any more, which was a bug.
- The result of getenv("USER") will not break waisserver any more if `NULL` is returned.

2.9 0.93

In this version code was added to send me a UDP packet each time the INFO database gets re-indexed. This should not disturb the normal operation of the server, even if the sending fails. I included that to track use of the software. You can switch this off by defining DO_NOT_TELL_ABOUT_ME in `Defaults.tmpl`

2.10 Beta 05

- A bug in calculation the "Total word count" has been fixed
- Indexing and retrieval of files compressed by the GNU gzip is now supported. If you want to index a file `TEST.gz`, call waisindex with the extension stripped:

  ```
  waisindex -t text -d test TEST
  ```

  This worked formerly only with the standard `compress` command and the "`.Z`" extension.

3 Indexing

If you want to index a collection of files containing one or more documents using **FreeWAIS-sf** first look at the supported document type formats. You may look at the manual page of waisindex[10] or type `waisindex` without arguments for information about supported document types.
If your document object is one of the supported types, run the waisindex command with the t -t argument:
`waisindex -d index_file_root_name -t doc_type object object ...`
where:

-d denotes the rootname to be used for the collection of index files and will include suffixes created by the waisindex program

-t denotes the document types supported by the waisindex command

object is the file name of a target object to be indexed by the command.

Both the `-d` and object specifications support full pathnames and default to the current directory if no pathnames are provided.
If you have a document in an unsupported format or would like to split individual documents into fields, you must generate two document format files.
First you should decide which fields you will use, and what their name should be. Usually it is a good idea to provide further information about what the fields contain or mean. The field definition file <*database*>`.fde` contains this information. Here is an example:

```
py: publication year
au: author
ti: title
jt: journal title
ck: citation key
```

Waisindex will put the names of the generated fields in the server description (<*database*>`.src`) it will produce if a field definition file is encountered.
Now comes the hard part. You now have to generate a format file <*database*>`.fmt` for your new database. Look at the examples[11] on our ftp server if the following is too obscure. The abstract syntax for the specification files follows:

[10] http://ls6-www.informatik.uni-dortmund.de/htbin/man?waisindex()
[11] ftp://ls6-www.informatik.uni-dortmund.de/pub/wais

3.1 Document Specification Syntax

format	→	<record-end> regexp speclist	
speclist	→	spec ∣ spec speclist	
		<field>	REGEXP regexp
			field-list
spec	→		options
			index-specs
		<end>	regexp
options	→	□ ∣ option options	
		NUMERIC regexp INT	
options	→	HEADLINE regexp INT	
		DATE REGEXP REGEXP date date date regexp	
index-specs	→	□ ∣ index-spec index-specs	
index-spec	→	index-type dicts	
index-type	→	TEXT ∣ SOUNDEX ∣ PHONIX	
dicts	→	GLOBAL ∣ LOCAL ∣ BOTH	
date	→	DAY ∣ MONTH month-spec ∣ YEAR	
month-spec	→	□ ∣ STRING	
field-list	→	□ ∣ WORD field-list	

Now what do the index types LOCAL, GLOBAL and BOTH mean?
Note that **FreeWAIS-sf** generates dictionaries and inverted files for each field. If there were no global or default field for general text search one would always have to specify a field in your queries. To avoid this inconvenience waisindex generates a default field which is used for searching if there is no fields specified. This field is called global, since it usually contains the information of some of the other fields which the administrator assumes to be useful for unexperienced users.
The contents of the index field are defined by using the keywords LOCAL, GLOBAL and BOTH in the field definitions.

LOCAL Words in this field are not inserted in the global database and are only retrievable by field query. Numeric and date fields are particularly well suited to the use of this option.

GLOBAL Words in this field are only inserted in the global database. This is analogous to the default free-text search of other versions of freeWAIS but allows all or part of the document to be indexed for general search. Do not specify a field name in this case since the field will be empty!

BOTH Words in this field are inserted in both the current field database and the global database.

Regular expressions are used to find, match, and parse strings encountered in a document. These regular expressions are used within the .fmt file to delimit field entries. For those not familiar with regular expressions, some conventions are provided in the following section:

3.1.1 REGULAR EXPRESSION SYNTAX

Operator	Meaning
`x`	the character "x"
`"x"`	an "x", even if x is an operator
`\x`	an "x", even if x is an operator
`[xy]`	the character x or y
`[x-z]`	the characters x, y or z
`[^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

Note that the scanner requires an aditional level of escaping because the `'/'` indicates the end of the regular expression. So `'/'` **must** be escaped by a backslash: `'\/'`. If you need a backslash in you regexp, it must me escaped to: `'\\'`. Since formfeed and other control characters are often needed `'\`*x*`'` for *x* from `'A'` to `'Z'` is mapped to `'^`*x*`'` (ctrl *x*,). This means `'A'` is subtracted from the original character.
For example `\A=^A(ctrnl A)`, `\B = ^B`, `...\J = \n` (newline). This is somewhat ad-hoc, but was easy to implement and allows users of limited editors to enter control characters in the format file.
Here is a first small example of a structured document collection to be indexed.

3.1.2 Small example

Suppost you have files containing many documents, from which you will only index thier titles contained between `<TI:>` tags:

```
<TI:> Information Retrieval <:TI>
[...]
<TI:> Database Systems <:TI>
```

You format file (.fmt) should look like this:

```
<field> /<TI:>/
ti TEXT LOCAL
<end> /<:TI>/
```

Now that you have you format file <*database*>`.fmt`, call waisindex with option `'-t fields'`. Because the `.fmt` file already begins with the index file root name it is used by the **FreeWAIS-sf** waisindex program. The -t fields option *must* have a `.fmt` file present.
Now its time to give a more complicated example:

3.1.3 DOCUMENT SPEC EXAMPLE

For an example file like this

```
CK: Mostert/etal:89
AU: Mostert, D.N.J.; Eloff, J.H.P.; von Solms, S.H.
TI: A Methodology for Measuring User Satisfaction.
JT: Information processing & management.
```

```
ED: JAN-01-1994
VO: 25
PY: 1989
NO: 5
PP: 545
^L
CK: Qiu:90
AU: Qiu, Liwen
TI: An Empirical Examination of the Existing Models for
    Bradford's Law.
JT: Information processing & management.
ED: JAN-01-1994
VO: 26
PY: 1990
NO: 5
PP: 655
^L
```

the following format file could be used:

<record-end> /^L/	records are separated by form feeds (Cntrl-L not '^L' ! \L would be equivalent)
<layout>	
<headline> /^TI: / /^[A-Z][A-Z]:/ 50 /TI: /	line which starts with 'TI: ' and ends with /^[A-Z][A-Z]:/ first 50 chars after 'TI: ' are copied to the chars 1 to 50 of the headline.
<headline> /^AU: / /^[A-Z][A-Z]:/ 50 /AU: /	line which starts with 'AU: ' and ends with /^[A-Z][A-Z]:/ first 50 chars after 'AU: ' are copied to chars 51 to 100 of the headline.
<date> /^ED: / /%s-%d-%d/ month string day year /^ED: [^]/	line starts with /^ED: / /%s-%d-%d/ is sscanf_argument Month is a string (nummber by default if you don't type 'string') after month is day, then year. /^ED: [^]/ is the begin of index position. <end> end of layout.
<field> /^PY: / py <numeric> /^PY: [^]/ 4 TEXT LOCAL <end> /^[A-Z][A-Z]:/	It is a numeric field of length 4, begin at first number of PY, e.g if the number is 1990 then the regexp /^PY: [^]/ means ^ here is the begin of the number (begin of line by default) indexed with type TEXT in the local dictionary only and ends with the next tag. Note that matching for the end tag is restricted to posintions after the skip regexp /^PY: [^]/. This enshures, that the PY: is not recognized as end tag, causing the field to be empty.
<field> /^AU: / au SOUNDEX LOCAL TEXT LOCAL <end> /^[A-Z][A-Z]:/	field 'au' is indexed with types TEXT and SOUNDEX in the local dictionary.
<field> /^CK: / ck TEXT BOTH <end> /^[A-Z][A-Z]:/	field 'ck' is indexed with type text in the local and the global dict.
<field> /^TI: / ti stemming TEXT BOTH <end> /^[A-Z][A-Z]:/	field 'ti' is indexed with type text in the local and the global dict. 'stemming' indicate that the stemmer is to call for this field (no stemming by default).
<field> /^AU: / au TEXT BOTH <end> /^[A-Z][A-Z]:/	field 'au' is indexed with type text in the local and the global dict.
<field> /^JT: / /^JT: [^]/ ti jt TEXT BOTH <end> /^[A-Z][A-Z]:/	fields 'jt' and 'jt' are indexed with type text in the local and the global dict. The begin is at the first character after this regexp /^JT: [^]/ (optional, begin of line by default), e.g JT: Information processing & management. ^ here is the beginning to index.
<field> /^AU: / TEXT GLOBAL <end> /^[A-Z][A-Z]:/	line which begins with the regexp /^AU: / should be indexed only in global dictionary.

3.2 Note

- If a separator is a empty line the regexp for this is `\J`.
- The length of a headline is 100 characters. If you want to change the length of headline update `MAX_HEADER_LEN` and `MAX_HEADLINE_LEN` in `Defaults.tmpl`.
- Of course, you can use other options too, e.g, `waisindex -d index_filename -t fields -r filename`

- If you want to create only one field, but the old fields should not be deleted you can use the option `-nfields`. In the document specification you must add new fields which you want to index.

  Example

  ```
  ...
  <field> /^AU: /
  names TEXT LOCAL | BOTH
  <end> /^[A-Z][A-Z]:/
  ```

  Only field `'names'` would be created.

- If you want to specify the headline corresponding to the format defined, e.g. (irlist, mail_or_rmail, etc.), and don't want to use the standard field format for headlines you must call this:
  `waisindex -d test -t fields -t mail_or_rmail TEST`.

  The `-t mail_or_rmail` option must be after the `-t fields` option!

When you have generated your format file, run waisindex with the `'-t fields'` flag and see if your specification works. If the parser encounters a syntax error, there is very limited support for debugging the offending part of your specification.
Best way to circumvent this is to start with a very simple definition and try waisindex every now and then.

4 Queries

4.1 How can you make a search query ?

4.1.1 QUERY SYNTAX

<table>
<tr><td>query</td><td>→</td><td>expression</td><td></td></tr>
<tr><td>expression</td><td>→</td><td>term</td><td></td></tr>
<tr><td></td><td></td><td>expression OR term</td><td></td></tr>
<tr><td></td><td></td><td>expression term</td><td>OR may be ommited</td></tr>
<tr><td>term</td><td>→</td><td>factor</td><td></td></tr>
<tr><td></td><td></td><td>term AND factor</td><td></td></tr>
<tr><td></td><td></td><td>term NOT factor</td><td>NOT really means AND NOT</td></tr>
<tr><td>factor</td><td>→</td><td>word</td><td></td></tr>
<tr><td></td><td></td><td>(expression)</td><td></td></tr>
<tr><td></td><td></td><td>field = (s_expression)</td><td></td></tr>
<tr><td></td><td></td><td>field = word</td><td></td></tr>
<tr><td></td><td></td><td>field = phonix|soundex word</td><td>phonix or soundex search</td></tr>
<tr><td></td><td></td><td>field = = word</td><td>for numeric fields</td></tr>
<tr><td></td><td></td><td>field < word</td><td></td></tr>
<tr><td></td><td></td><td>field > word</td><td></td></tr>
<tr><td colspan="4">same as above, but no field spec is allowed, since one is given already</td></tr>
<tr><td>s_expression</td><td>→</td><td>s_term</td><td></td></tr>
<tr><td></td><td></td><td>s_expression OR s_term</td><td></td></tr>
<tr><td></td><td></td><td>s_expression s_term</td><td></td></tr>
<tr><td>s_term</td><td>→</td><td>s_factor</td><td></td></tr>
<tr><td></td><td></td><td>s_term AND s_factor</td><td></td></tr>
<tr><td></td><td></td><td>s_term NOT s_factor</td><td></td></tr>
<tr><td>s_factor</td><td>→</td><td>WORD</td><td></td></tr>
<tr><td></td><td></td><td>(s_expression)</td><td></td></tr>
</table>

4.1.2 QUERY EXAMPLES

`information retrieval`	free text queries
`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)`	one of them in title
`ti=(information AND retrieval)`	both of them in title
`ti=(information NOT retrieval)`	"information" in title and "retrieval" not in title
`py==1990`	numeric equal
`py<1990`	
`py>1990`	
`au=(soundex salatan)`	soundex search matches eg. "Salton"
`ti=('information retrieval')`	literal search
`ti=(information system*)`	partial search

The use of capital letters for the Boolean operators is not required but is provided in this example for clarity. All search matching is case-insensitive.

5 Weighting

Here is an excerpt of the corresponding smart routine:
The documents would be presented by term vectors of the form

$$D = (t_0, w_{d0}; t_1, w_d 1; ..., t_t, w_{dt})$$

where each t_k identifies a content term assigned to some sample document and w_{dk} represents the weight of term t_k in Document D (or query Q). Thus, a typical query Q might be formulated as

$$Q = (q_0, w_{q0}; q_1, w_q 1; ...; q_t, w_{qt})$$

where q_k once again reprents a term assigned to query Q. The weights could be allowed to vary continuosly between 0 and 14, the higher weight assignments near 1 being used for the most important terms, whereas lower weights near 0 would characterize the less important terms. Given the vector representation, a query-document similarity value may be obtained by comparing the corresponding vectors, using for example the conventional vector product formula

$$similarity(Q, D) = sum(w_{qk} * w_{dk}), k = 1tot.$$

Three factors important for term_weighting:

1. term frequency in individual document (recall)
2. inverse document frequency (precision)
3. document length (vector length)

Term frequency component used: $new\_wgt = 0.5 + 0.5 * tf/max\_tf$ augmented normalized term frequency (tf factor normalized by maximum tf in the vector, and further normalized to lie between 0.5 and 1.0).
Collection frequency component used: 1.0 no change in weight; use original term frequency component.
Normalization component used: $\sqrt{\sum new\_wgt^2} = vector\_length$.
Thus, document term weight is: $w_{dk} = new\_wgt/vector\_length$
By query term weighting, it is assumed that tf is equal to 1. So that $w_{qk} = 1$.

5.1 Document term weighting by standard Boolean formulations

Given queries "$AorB$", "$AandB$", and "$AnotB$" (A and-not B), a document X with weights $d_A(X)$ and $d_B(X)$ for terms A and B. The retrieval values are:

- $d_A(X) + d_B(X)$ for query $(AorB)$
- $min(d_A(X, d_B(X)$ for query $(AandB)$
- $min(A, 1 - d_B(X))$ for query $(AnotB(Aand - notB))$

Note: If you use these new formula the inverted files (.inv) will have a new structure.

5.1.1 Term weighting in wais

$$w_{dk} = ((log(tf) + 10) * idf)/number\_of\_terms\_in\_a\_document$$

- tf = term frequency. Initially is tf = 5.
- idf = 1/term_frequency_in_the_collection

5.1.2 Disadvantages

- For example a database consists of 10 documents. A term which occurs 10 times in a document has the idf = 1/10. The same term which occurs in 10 documents has also idf = 1/10. One can say in both cases the term has the same relevance. This may not be correct.
- The normalization factor is not the weight of each term in the document but number of terms in a document.

6 Installation

Just run the `configure` script in the Distribution. If you have a **working** `imake` on you system, enter `xmkmf -a` now.
Then type:

`make`
: to build the system and run the tests

`make install`
: to install binaries and scripts

`make install.man`
: to install the manual pages *only with imake. The default Makefiles install them with* `make install`

`make install.lib`
: to install the libraries *only with imake*.

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

`make veryclean`
: removes files generated by flex, bison, dvips, latex

7 FreeWAIS-sf and WWW

Linking was tested with:

- NCSA Mosaic 2.4
- Tuebingen Univ Mosaic 2.4.2
- CERN httpd 3.0

7.1 freeWAIS-sf and CERN httpd

Direct WAIS access for CERN httpd 3.0 is easy to provide with freeWAIS-sf 1.1. 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[12]
- Replace WWW.cf in the top-level directory with this code in Appendix A
- 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.
Enjoy
Jean-Philippe (syj@ecmwf.co.uk)
There is also a CGI Gateway especially suited for **FreeWAIS-sf** which enables usage of Mosaic forms for searching. See the SFgate Documentation and Demos[13] on our http server.

8 FreeWAIS-sf and gopher 2.1.1 (by Steve Hsieh)

Files used:

```
gopher2_1_1.tar.Z
freeWAIS-sf-1.0.tgz
```

Changes made to the distribution:

Apply patches 1-6,8-9 (not 7) to the original freewais-sf-1.0 source tree. You may or may not want to apply all of them. Patches available in the pub/wais directory on ftp://ls6-www.informatik.uni-dortumund.de
FreeWAIS-sf-1.1 does not need any patches.

SPECIAL NOTES:

Linux: you must at least apply patch001

Solaris: do not apply patch005

[12] ftp://ftp.germany.eu.net/pub/infosystems/www/cern/WWW-Imake.tar.gz
[13] http://ls6-www.informatik.uni-dortmund.de/SFgate/SFgate

How to 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-1.0
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-1.0 . . .
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-1.0/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 *.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.

9 FreeWAIS-sf and gopher 2.016 (by Steve Hsieh)

Here are the changes that I made to get freewais1.0 and gopher2.016 running happily together the way we wanted:

- The important freewais-sf configure options (questions below with no value can take on a value of your choice):

```
use your systems regexph.h? (no)       <-- necessary for gopher to work
headlines > 16MB? (yes)
use systems ctype? (yes)               <-- necessary for gopher to work
compile with -DLOCAL_SEARCH? (yes)     <-- necessary for gopher to do local searches
modify URL?
install where?
use shm cache?
disable UDP packet sending?
```

- make freewais-sf
- cd to *freeWAIS-sf-dir*`/ir` and type, `ar cq sfextra.a query_y.o field_y.o query_l.o`. *Some systems need a call of ranlib:* `ranlib sfextra.a`.
- cd to *freeWAIS-sf-dir*`/bin` and type,

```
ln -s ../regexp/libregexp.a regexp.a
ln -s ../lib/libftw.a libftw.a
```

- Create `ui`, `ir`, and `bin` symbolic links in gopher source directory to corresponding `ui`, `ir`, and `bin` dirs in *freewais-sf-dir* as instructed in gopherd installation docs regarding wais.
- Apply this patch[14] to the gopher-dir/gopherd directory. Instructions on how to do this are included with the file.
- make gopher

That's all there is to it...

Contents of patch file

Below, I have summarized (in words) the changes that the patch above makes to files in *gopher-dir* /gopherd. **Please use the patch to make the actual changes**, as there may be typos or other kinds of errors below...

- in *gopher-dir*`/gopherd/waisgopher.c` Between the two lines

```
     readSearchResponseAPDU(&query_response, response_message +
HEADER_LENGTH);

     display_search_response(query_response, server_name, service, database,
SourceName, sockfd, view, isgplus);
```

  added:

```
     LOGGopher(sockfd, "search %s for %s", database, keywords);
```

  Also replaced:

[14] ftp://ls6-www.informatik.uni-dortmund.de/pub/wais/gopher2.016.patch.gz

```
MIN
```

with

```
MINIMUM
```

- in *gopher-dir*/`gopherd/Makefile`: Commented out existing `WAISOBJ` and replaced with:

```
WAISOBJ = ../ir/libinv.a ../ir/libclient.a \
          ../ir/libwais.a $(WAISGATEOBJ) \
          ../bin/regexp.a ../ir/libinv.a \
          ../ir/sfextra.a ../bin/regexp.a ../bin/libftw.a
```

- For `Path=7/indexdir/index` to work in `.links` files, we could do this by changing the `openDatabase` call `Waisindex.c` from

```
db = openDatabase(new\_db\_name, false, true);

db = openDatabase(new\_db\_name, false, true, false);
```

But booleans don't work when we do this...so instead, we modify the following files:

- In *gopher-dir*/`gopherd/gopherd.c`:

Replace

```
case '7':
     /*** It's an index capability ***/
     result = GDCCanSearch(Config, CurrentPeerName, CurrentPeerIP,
                           NUMgopherds);

     if (result == SITE_NOACCESS) {
          Abortoutput(sockfd, GDCgetBummerMsg(Config));
          LOGGopher(sockfd, "Denied access for %s", Selstr+1);
          break;
     } else if (result == SITE_TOOBUSY) {
          Abortoutput(sockfd, "Sorry, too busy now...");
          break;
     }

     Do_IndexTrans(sockfd, Selstr+1, cmd, TRUE);

     break;
```

with:

```
case '7':
{
     int Index_type=0;

     /*** It's an index capability ***/
     result = GDCCanSearch(Config, CurrentPeerName, CurrentPeerIP,
                           NUMgopherds);

     if (result == SITE_NOACCESS) {
          Abortoutput(sockfd, GDCgetBummerMsg(Config));
          LOGGopher(sockfd, "Denied access for %s", Selstr+1);
          break;
     } else if (result == SITE_TOOBUSY) {
          Abortoutput(sockfd, "Sorry, too busy now...");
          break;
     }
```

```
/* see if index is type 1, which is a wais index */
Index_type = Find_index_type(Selstr+1);

if (Index_type == 1)
  {
     char waisfname[512];  /*** Ick this is gross ***/

     strcpy(waisfname, Selstr+1);
     if (strlen(waisfname) <= 4 ||
         strncmp(&waisfname[strlen(waisfname)-4],".src",4) )
          strcat(waisfname, ".src");
     SearchRemoteWAIS(sockfd, waisfname, cmd, view);
   }
else
  {
    Do_IndexTrans(sockfd, Selstr+1, cmd, TRUE);
  }

break;
}
```

- Then for mindex searches to work, edit *gopher-dir*/`gopherd/mindexd.c` and comment out:

```
if (strcmp(slaves[i].host, "localhost") == 0 ||
    strcasecmp(slaves[i].host, Zehostname)  == 0) {
     CMDsetSelstr(cmd, GSgetPath(gs));
     CMDsetSearch(cmd, queryline);
     CMDsetGplus(cmd, FALSE);

     Do_IndexTrans(sockfd, slaves[i].pathname+1, cmd, FALSE);
} else {
```

  I also commented out the associated closing } with that paragraph a little over 30 lines down.

- In function `do_mindexd(sockfd, config_filename, search, isgplus, view)`: Between the two lines

```
HandleQuery(sockfd, search);
close(sockfd);
```

  added:

```
     LOGGopher(sockfd, "mindex search using %s.mindex for %s",
config_filename,
search);
```

  There is also a bug in *gopher-dir*/`gopherd/waisgopher.c` As it stands, if you do a search using `Path=waissrc:...`, and search is empty, server does not return a '.', leaving old clients hanging

  To fix, found the line (near or on line 557)

```
writestring(sockfd, ".\r\n");
```

  and moved it to after the second to last curly brace of that procedure call. In other words, at the end of this procedure:

```
            Mydisplay_text_record_completely( info->Text[k++], false, sockfd);
               }
            }
      }
}
```

became

```
            Mydisplay_text_record_completely( info->Text[k++], false, sockfd);
               }
            }
      }
      writestring(sockfd, ".\r\n");
}
```

- Some additional enhancements to *gopher-dir*/`gopherd/mindexd.c` were made as well to make it more robust and handle connections that time out. They are not documented here; see the patch for details.

10 Special needs

10.1 Increasing the index block size (by Steve Hsieh)

In order to increase the index block size, so that words that appear many many times are indexed:

- In `freewais-sf-dir/ir/irfiles.h` change

```
#else
#define INDEX_BLOCK_SIZE_SIZE 2
#endif
```

to

```
#else
#define INDEX_BLOCK_SIZE_SIZE 3
#endif
```

- In `freewais-sf-dir/ir/server.h` change

```
     #define BUFSZ 100000     /* size of our comm buffer */
```

to

```
     #define BUFSZ 1000000    /* size of our comm buffer */
```

- In *freewais-sf-dir*/`ui/waissearch.c` Change

```
     #define MAX_MESSAGE_LEN 100000
```

to

```
     #define MAX_MESSAGE_LEN BUFSZ
```

The last step will be obsolet in version > 1.0.

11 TODO

Here is a somewhat adhoc list of things to fix and features to add:

X11R6 Some might have noticed that the X clients do not (yet) compile with the new X11 release. Porting should not be too difficult for someone with some X11 Knowledge

ANSI Code currently does not pass a strikt ANSI Compiler. We intend to switch to the Prototyping scheme known from the WWW library:

```
#ifdef __STDC__
#define ARGS1(t,a) \
                (t a)
#else  /* not ANSI */
#define ARGS1(t,a) (a) \
                t a;
#endif /* __STDC__ (ANSI) */
```

waisindex BUGS

filenames Waisindex has still problems with filenames. E.g. files with apostophes or asterics in them are not handled properly. Filenames with wildcards may enter the filename table despite the fact, that they do not exists.

`-a` The `-a` flag is not handled properly. Adding a file, which contains only a subset of the declared fields causes the other fields to be ignored by the server until a "complete" document is added.

Compressed Indexes There are several know methods for compressing inverted files which could save us disc space and significatly improve search speed.

Spatial Indexes (Notes from Doug Nebert)

We would like to add a field type into the SF software which would allow for the parsing of and indexing of geographic coordinates that describe the outline of a data set or document. Software has been written outside of SF to do the parsing (using flex), and the indexing and overlay routines have been included into the freeWAIS-0.3 code. Now we need to integrate the code so that we can perform full field searching of text, dates, numbers, and geography in one indexing system.

Forms (Notes from Doug Nebert)

It seems to me that if the SF crowd can consistently use the .fde file incorporated into the available .src file that a functionality like "explain" can be developed to allow the client to determine what attributes are being used and formulate a query window to match it. probably easier would be to have a "form" resource file which could be retrieved from the server (e.g. query.html) by a "smart" http client...

Relevance Feedback Notice, that the thing build in freeWAIS* is not "Relevance Feedback". It is rathersome kind of query expansion. Real Relevance Feedback is proved to produce much more effective ranking.

update (Notes from Marc Edgar)

How about having a script that could automatically update a the database. That is, a record would be kept of which files were in the database. This record (.rec or some such) could be used instead of having to remember or find the command that created it. That is, it would support something like,

```
waisindex -update filename.rec
```

to rebuild the database.

format files (Notes from Marc Edgar)

Programs like this become immortal when they do not take any special knowlege to use. Format files for common data types would make **FreeWAIS-sf** more accessible. Maybe you've already done this but format files like, FAQ.fmt, email.fmt, usenet.fmt would be very helpful, (and probably not that hard to write.) Maybe creating an incoming directory on your ftp server would be useful, so that users could post their .fmt files and save you from having to do the work.

Z39.50 V2 (Notes from Doug Nebert)

It seems that the functionality you have provided matches very well the basic abilities of Z39.50 V2 and V3 in terms of fields and search. If there were a way to identify registered attributes then the construction of a gateway from ZDIST to an **FreeWAIS-sf** store of data would be possible, allowing people to keep their data in one format and serve the V1 and non V1 communities.

My thoughts regarding a linkage between **FreeWAIS-sf** and a full Z39.50 V2-3 release such as ZDIST were to provide a link into the new capabilities and other "compliant" clients out there. But I think much of the API work could be done with the help of CNIDR personnel – their "linkage" back into freeWAIS-0.3 disabled some of the functionality whereas **FreeWAIS-sf** is more on the same level of sophistication as V2 and should be easier to connect to. If such a connection can be made it would allow you all to maintain and enhance the existing code and have some partners out here work on maintaining the API connection, taking the load off you except in consultation . . .

Fields Note from Alberto Accomazzi (Darin McKeever proposed similar features).

First of all, when indexing the documents, the user should be able to specify the following for each field to be indexed:

- minimum word length
- set of characters composing the terms – i.e. the delimiter set
- synonym file
- stopword file

This could be done by allowing the entries in the format file to look like:

```
<field> /^Authors: /
au TEXT BOTH minchars 2 word /[^ ;\n=()]+/
    stop abstracts_field_au.stop syn abstracts_field_syn.syn
<end>
```

Other things such as headline length should be specifiable in the `.fmt` file as well.

Documentation Counting the mails I receive every day leads to the conclusion that there is a lack of documentation.

man The online manuals are out of date.

document specs Many people have difficulties in building document specifications. Either there should be a nicer input format or someone should provide a compiler (+checking and testing?) for some prettier specification format.

other systems There should be more info on: How do i use **FreeWAIS-sf** with Gopher, Mosaic, httpd, perl, . . .

. . .

A new WWW.cf

```
/* 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
```

B Maintaining Databases

Here is what I use to maintain my databases. Most of the tricky stuff is handled by this Imakefile:

```
# Imakefile -- Imakefile to update wais databases
# Author          : Ulrich Pfeifer
# Created On      : Thu Feb 13 16:01:48 1992
# Last Modified By: Ulrich Pfeifer
# Last Modified On: Mon Aug 22 13:45:31 1994
# Update Count    : 238
# Status          : Unknown, Use with caution!

# HISTORY
# 21-Feb-1992           Ulrich Pfeifer
#    Last Modified: Tue Feb 18 11:01:45 1992 #8 (Ulrich Pfeifer)
#    Changed IRDIGEST
# 18-Feb-1992           Ulrich Pfeifer
#    Last Modified: Fri Feb 14 12:06:41 1992 #7 (Ulrich Pfeifer)
#    Added bibdb format
# 14-Feb-1992           Ulrich Pfeifer
#    Last Modified: Thu Feb 13 18:14:24 1992 #2 (Ulrich Pfeifer)
#    Added Emacs-info

#undef DEBUG
#undef CHECK_ONLY

# Commands
               LS = /bin/ls
              CMP = /bin/cmp
        WAISINDEX = /usr/local/ls6/wais/bin/waisindex -nopairs -nocat -export
             PERL = /usr/local/bin/perl
           TMPDIR = /tmp

        SERVERDIR = /usr/local/ls6/wais/wais-sources

         MAILSERV = /home/crew/mailserv
        NOSFERATU = ${MAILSERV}/Mail/NOSFERATU
```

```
              SCOG = ${MAILSERV}/Mail/SCOG


         LWAISHOME = /usr/wais
       WAISDOCSDIR = ${LWAISHOME}/wais-docs
          WAISDOCS = ${WAISDOCSDIR}/wais-docs

           PFEIFER = /home/crew/pfeifer
          SUNFLASH = $(WAISDOCSDIR)/SunFlash $(PFEIFER)/Mail-public/SunFlash
           SUGINFO = $(WAISDOCSDIR)/sug-info $(PFEIFER)/Mail-public/sug-info

       IRDIGESTSRC = ${WAISDOCSDIR}/IRLIST.abstracts
        BIBDB_HTML = ${WAISDOCSDIR}/bibdb.html
            HCIBIB = ${WAISDOCSDIR}/hcibib
              DEMO = ${WAISDOCSDIR}/demo.html
          JOURNALS = ${WAISDOCSDIR}/journals
         LIBRARIES = ${WAISDOCSDIR}/libraries.america.gz
               FAQ = /usr/local/ls6/doc/faql
           WWWHOME = /usr/WWW/pages
           FTPHOME = /local-home/ftp/pub/doc
NOSFERATU_GLOSSAY = /usr/local/ls6/src+data/src/nosferatu/glossary


#define WaisCleanTarget(database)                                     @@\
veryclean::                                                           @@\
        ${RM} Concat(database,.cat) Concat(database,.dct) \
        Concat(database,.dlm) Concat(database,.doc) \
        Concat(database,.fn) Concat(database,.hl) \
        Concat(database,.inv) Concat(database,.src) \
        Concat(database,_field_*)                                     @@\
                                                                      @@\
Concat(database,.fmt):                                                @@\
        touch Concat(database,.fmt)
/*
 * WaisIndexProc, the working horse
 */
#ifdef CHECK_ONLY
#define WaisIndexProc(database, type, sources, options) \
        echo Database database needs reindexing ;
#else /* CHECK_ONLY */
#define WaisIndexProc(database, type, sources, options) \
        cp Concat(database,.fmt) $(TMPDIR)/Concat(database,.fmt) ; \    @@\
        $(WAISINDEX) options -t type -d $(TMPDIR)/database sources ; \  @@\
        if test -f $(TMPDIR)/Concat(database,.src) ; then \             @@\
            echo Indexing of database was successfull ; \              @@\
            $(RM) $(TMPDIR)/Concat(database,.fmt) ; \                  @@\
            mv $(TMPDIR)/database/**/* . ; \                           @@\
            if test -f manifest  ; then \                              @@\
               mv manifest  Concat(database,-MANIFEST); \              @@\
            fi ; \                                                     @@\
        else \                                                         @@\
            echo Indexing of database failed ; \                       @@\
            $(RM) $(TMPDIR)/database/**/* manifest ; \                 @@\
        fi ;
#endif /* CHECK_ONLY */

/*
 * WaisOptionTarget - the normal case
 */
#define WaisOptionTarget(database, type, sources, options)            @@\
all:: Concat(database,.doc)                                           @@\
                                                                      @@\
Concat(database,.doc): Concat(database,.fmt) sources                  @@\
        WaisIndexProc(database, type, sources, options)               @@\
WaisCleanTarget(database)

#define WaisTarget(database, type, sources)                           @@\
```

```
WaisOptionTarget(database, type, sources,)

/*
 * WaisDir2TargetOpt index a directory, if /bin/ls signals change
 */
#define WaisDir2Target(database, type, lsargs, sources, options)        @@\
all:: database Concat(database,.doc)                                    @@\
                                                                        @@\
database Concat(database,-MANIFEST):                                    @@\
        @echo Testing sou/**/rces of database                           @@\
        @${LS} -l lsargs | grep -v MANIFEST > manifest; \
        if ${CMP} Concat(database,-MANIFEST) manifest; \
        then \
           echo "No differences encountered"; \
           ${RM} manifest; \
        else \
           WaisIndexProc(database, type, sources, options) \
        fi;                                                             @@\
                                                                        @@\
Concat(database,.doc): Concat(database,.fmt)                            @@\
        WaisIndexProc(database, type, sources, options)                 @@\
                                                                        @@\
clean::                                                                 @@\
        ${RM}  manifest                                                 @@\
                                                                        @@\
clean::                                                                 @@\
        ${RM} Concat(database,-MANIFEST)                                @@\
                                                                        @@\
WaisCleanTarget(database)

#define WaisDirTarget(database, type, sources, options) \
        WaisDir2Target(database, type, sources, sources, options)
/*
 * The databases
 */
#ifdef DEBUG
WaisDirTarget(test,fields,TEST,-r)
WaisTarget(test1,fields,TEST)
WaisOptionTarget(test2,fields,TEST, -T HTML)
WaisDir2Target(test3,fields,TEST,TEST, -T HTML)
#else /* DEBUG */
#define WSRCPAT *.src
WaisDirTarget(directory-of-servers,server,Concat(${SERVERDIR}/,WSRCPAT),)
WaisDirTarget(journals,fields,$(JOURNALS),-r)
WaisDirTarget(nosferatu-glossary,text -T HTML,${NOSFERATU_GLOSSAY},-r)
WaisDirTarget(wais-docs,text,${WAISDOCS},-r)
WaisDirTarget(www-pages,fields,`find $(WWWHOME) -type f -name "*.html"
\
-print`, -t URL $(WWWHOME) \@@http://ls6-www.informatik.uni-dortmund.de)
WaisDirTarget(ftp-pages,fields,`find $(FTPHOME) -type f -name "*.html"
\
-print`, -t URL $(FTPHOME) \@@ftp://ls6-www.informatik.uni-dortmund.de/pub/doc)
WaisDir2Target(HCIBIB,fields, $(HCIBIB), `$(PERL) -e 'for \
(<$(HCIBIB)/*\.html.gz>) { s/.gz/\n/; print }'`, -T HTML -stop HCIBIB.stop)
WaisOptionTarget(bibdb-html,fields,${BIBDB_HTML}, -T HTML)
WaisOptionTarget(demo,fields,${DEMO}, -T HTML)
WaisOptionTarget(ls6-help,formfeed,${FAQ},-T HTML)
WaisTarget(INFO,server,${SERVERDIR}/bibdb-html.src ${SERVERDIR}/journals.src)
WaisTarget(irdigest,fields,${IRDIGESTSRC})
WaisTarget(libraries,dash,$(LIBRARIES))
WaisTarget(nosferatu,mail_or_rmail,${NOSFERATU})
WaisTarget(scog,mail_or_rmail,${SCOG})
WaisTarget(suginfo,mail_or_rmail,${SUGINFO})
WaisTarget(sunflash,mail_or_rmail,${SUNFLASH})
#endif /* DEBUG */
```