NYSERNet White Pages Pilot Project: User's Handbook

Marshall T. Rose
NYSERNet, Inc.
mrose@nisc.nyser.net

May 29, 1990

Abstract

The need for a comprehensive white pages service increases in relation to the size of the user community. The early Internet was served well by a relatively simple facility. Today's rapidly expanding Internet has outstripped the capabilities of the existing system. In order to meet new requirements, NYSERNet, Inc. is sponsoring a pilot project to provide white pages service based on the OSI Directory.

This document describes the pilot project from a user's perspective and provides operational reference for the use of the white pages service.

[0] This work was partially supported by the U.S. Defense Advanced Research Projects Agency and the Rome Air Development Center of the U.S. Air Force Systems Command under contract number F30602–88–C–0016. The content of the information contained herein does not necessarily reflect the position or the policy of the U.S. Government, and no official endorsement should be inferred.

Contents

1 Introduction **1**

1.1 The White Pages Service . . . 3

1.2 The OSI Directory . . . 3

1.2.1 Naming . . . 4

1.3 Ramifications on the White Pages Service . . . 5

1.3.1 Unique Identification of Users . . . 5

1.3.2 Searching the White Pages . . . 6

1.3.3 Structure of Information . . . 7

1.4 Where to turn for help . . . 9

2 The White Pages User Interface **10**

2.1 Invoking Fred . . . 10

2.1.1 Fred resides locally . . . 10

2.1.2 Fred resides remotely . . . 12

2.2 Giving commands to Fred . . . 13

2.3 Let your fingers do the walking . . . 14

2.3.1 The Alias Command . . . 14

2.3.2 Back to Searching . . . 15

2.3.3 The Area Command . . . 16

2.3.4 Getting Help . . . 17

2.3.5 Reporting problems . . . 18

2.3.6 Quitting . . . 18

2.3.7 Setting Variables . . . 18

2.3.8 via WHOIS or mail . . . 19

2.4 More on Searching . . . 19

2.4.1 Input Field . . . 19

2.4.2 Record Type . . . 21

2.4.3 Area Designator 22
2.4.4 Output Control 23
2.5 Editing your Entry 24
2.5.1 Who are You? 24
2.5.2 Editing the Entry 24

3 The Directory User Interface 26

4 Other Programs of Interest 27
4.1 Faces 27
4.1.1 xwho 27
4.1.2 xface 28
4.2 Mail Composition 28

A Attributes for Persons in the Pilot Project 29

[0] This document (version #2.30) was LaTeXset May 29, 1990 with lplain v2.09-10/29/85.

Chapter 1

Introduction

This document is *The User's Handbook* for the NYSERNet White Pages Pilot. The goal of *The Handbook* is to provide a user with enough information to be able to make effective use of the white pages service. In practical terms, this means that *The Handbook* provides information on how to use the white pages from your host.

The OSI Directory is used to provide the white pages service. *The Handbook* is not intended as a tutorial nor a detailed description of the OSI Directory. However, in order to make effective use of the white pages, it is necessary to understand some rudimentary Directory concepts.

Your comments are welcome! The OSI Directory is a new, complex technology. Although *The Handbook* attempts to be straight-forward it probably doesn't succeed all the time. If you have comments on this document, send them to the Internet mailbox

```
wpp-camayocs@nisc.nyser.net
```

so that *The Handbook* can be improved.

If you are interested in sharing about experiences with other usres of the white pages, drop a note to

```
wpp-users-request@nisc.nyser.net
```

and ask to be added to the

```
wpp-users@nisc.nyser.net
```

discussion group.

Related Documentation

In theory, *The Handbook* is the only reference needed by all but the most sophisticated of users. However, if you're interested in finding out more about the NYSERNet White Pages Pilot Project and the underlying mechanisms and technology, there are a few documents you might want to look at.

The white paper *An Introduction to a NYSERNet White Pages Pilot Project* introduces the goals and phases of the pilot project. In addition, site administrators are provided with a document called *NYSERNet White Pages Pilot Project: Administrator's Guide*, or simply *The Guide*.

The research note *The Design of QUIPU* describes the design of the software used for the pilot project, whilst *Volume Five* of the ISODE User's Manual, henceforth termed *Volume Five*, describes the implementation of the software.

Your site administrator has access to all of these documents and can provide them to you as requested.

1.1 The White Pages Service

A natural function of computer networks is to form the *infrastructure* between the users they interconnect. For example, the electronic mail service offered by computer networks provides a means for users to collaborate towards some common goal. In the simplest cases, this collaboration may be solely for the dissemination of information. In other cases, two users may work on joint research project, using electronic mail as their primary means of communication.

Most network services are based on the implicit assumption that each user can supply *infrastructural information* to facilitate information transfers through the network. For example, electronic mail services expect that an originator can supply addressing information for all the intended recipients. It is not necessarily the task of electronic mail, per se, to provide this infrastructural information to the user.

This model works fine in small environments, particularly those where infrastructural information is not difficult to obtain and remember. However, the model does not scale well. Consider the case when the membership of a network consists of hundreds of thousands of users belonging to thousands of organizations. It is no longer reasonable for a single user to provide this information, except in very limited circumstances. Further, it is likely that some of the information changes frequently, due to personnel and other resource movement. The goal of a *white pages* service is to provide the necessary information, and to mask the complexity of the infrastructural information.

From the user's perspective, the NYSERNet White Pages Pilot Project focuses solely on providing infrastructural information dealing with human users.[1]

Naturally, this raises questions as to the underlying technology which provides the white pages service.

1.2 The OSI Directory

The OSI Directory is designed to provide for the management of information objects. The Directory's representation of an information object, typically called an *entry*, contains information about a person, a place, an organization, etc. Each entry consists of one or more attributes.

[1] The white pages service is perfectly capable of managing other kinds of information, e.g., keeping track of machine-related infrastructural information; however, this is beyond the scope of the pilot.

Each attribute consists of a type, indicating what kind of attribute it is, and one or more values (one of which is termed the *distinguished value*). Attribute values are structured using a data definition language called Abstract Syntax Notation One (ASN.1). This structuring is important. With structuring, different programs using the Directory will interpret information in the same way. In addition, the Directory will perform type-checking on the values in order to keep things consistent.

1.2.1 Naming

One of the attributes of an entry is particularly special: it is referred to as the *Relative Distinguished Name* (RDN) of the entry. The RDN is formed by taking the name of the attribute and its distinguished value. For example, if the attribute in question was called `countryName` and it had a distinguished value of `US`, then we might say that the RDN for the entry was `countryName=US`. Of course, this is strictly a "user-friendly" notation: the Directory uses a concise binary format for representing an RDN. Fortunately, the pilot project software allows simple textual strings to be used in their place and converts back and forth accordingly.

In the OSI Directory, information is primarily organized according to a hierarchical tree structure. The top of the tree is termed the *root*, and has no explicit name. To find the name of an object, termed its *Distinguished Name* (DN), one concatenates the RDNs found when traversing the tree by starting at the root and proceeding directly to the object's entry.

For purposes of discussion, we write a Distinguished Name as an ordered series of RDNs separated by an '`@`'-sign with the most significant RDN appearing at the left; e.g.,

```
countryName=US@organizationName=NYSERNet Inc.
```

refers to an entry with an RDN of `organizationName=NYSERNet Inc.` whose parent has an RDN of `countryName=US`. In turn, this parent entry is an immediate child of the root.

To avoid any potential ambiguity when using an interface to the Directory such as *fred*(1c) or *dish*(1c), one prefixes a '`@`'-sign to a string when referring to a fully qualified Distinguished Name; e.g.,

```
@countryName=US@organizationName=NYSERNet Inc.
```

always refers to the same entry regardless of context. Note that this is a convention only for interface programs such as these.

As a rule, unless searching, text before the `=`-sign is not case sensitive, neither is text after the `=`-sign.

The Directory itself is distributed, being composed of *Directory System Agents* (DSAs). A group of DSAs under a common administration is responsible for a portion of the tree, termed a *Directory Management Domain* (DMD). When a user wishes to access the Directory, a *Directory User Agent* (DUA) is invoked. This DUA contacts a DSA and issues requests. The DSA may (or may not) have the information locally available. If not, a decision has to be made: either the DSA can contact another DSA to get the information (this is called *chaining*); or, the DSA can tell the DUA to contact another DSA directly (this is called *referral*).

In short, the DSAs provide mechanisms for traversing the tree and manipulating the information contained therein.

In the context of the pilot project, each participating organization runs its own DMD for that organization. This usually consists of a single DSA containing information on that organization, with some of this information being replicated on additional DSAs.

1.3 Ramifications on the White Pages Service

In order to appreciate the "feel" of the white pages service, it is instructive to compare the white pages to an existing facility.

You might be familiar with an older facility called WHOIS. This uses a centralized database to keep track of information on various people, networks, hosts, and so on. This facility has proven useful for many years. Only recently, with the explosive growth of the Internet, has the WHOIS mechanism become unworkable.

1.3.1 Unique Identification of Users

Each entry in the WHOIS database is identified by a unique key, called a *handle*. This is (typically) a short string such as `MTR`. For a community many orders of magnitude larger than the current entries in the WHOIS database, a handle must contain some structure. This makes it possible to delegate naming authority to different organizations and thus de-centralize management of the white pages service.

In the white pages service, a Directory Distinguished Name is used to uniquely identify a person. Thus, while `MTR` might be enough to identify someone named "Marshall Rose" in the WHOIS database, the DN

```
c=US
    @o=NYSERNet Inc.
    @ou=Research and Development
    @ou=Western Development Office
    @cn=Marshall Rose
```

serves as the handle for the same person in the white pages service. (That's progress for you!)

Of course, you don't *really* have to type all that information in. The user interfaces provided with the pilot project allow you to manage very short strings to refer to these DNs. These interfaces also provide a means for incrementally building up a DN from scratch.

Actually, the handle in the example above is probably a somewhat longer than the average. In terms of the pilot project, a handle probably looks closer to:

```
c=US@o=Organization Name@ou=Unit Name@cn=FirstName LastName
```

While this is still a far cry from a simple three or four letter acronym, it is the price one pays for using a service designed to meet the needs of a global (or galactic) population.

1.3.2 Searching the White Pages

When the WHOIS database is searched, *all* entries in the database are examined for a match. Since the current size of the WHOIS database is estimated at roughly 70,000 entries, this is an appropriate strategy.

Unfortunately, the potential size of the white pages is many orders of magnitude larger than that of the WHOIS database. As such, the information contained in the white pages is distributed. This makes management of the information a shared responsibility, and has the potential to address organization-specific privacy concerns.

Thus, when the white pages service is invoked, searches are performed relative to a particular *area*. This is similar to the White Pages of the telephone system — there are several white pages, one for each particular geographical area. As such, before you can find someone's entry in the white pages, you have to already know the area in which they are listed.

The default area is the portion of the Directory corresponding to your own organization. Of course, if you specify a user's handle (a fully-qualified Distinguished

Name), this bypasses the default area and goes directly to the portion of the Directory containing the desired entry.

Usually, when you are trying to find an entry, you have only partial information. For example, you might know parts of the name of the organization and the person you're looking for. In this case, it is natural to use an iterative process to find the information you desire. You begin by finding the organization(s) likely to contain the entry, you then initiate a search starting at that area.

Having said that, I'll let you in on a little secret: in addition to people, organizations and organizational units also have entries in the Directory. As such, searching an *area* is nothing more than starting a search at a particular node at the tree. Thus, you might look for the organization starting at the `@c=US` node. In order to make searching easy, the pilot project requires that all organizations be listed directly under this node. How the subtree is structured beyond that is an organization-specific matter, although the pilot project provides various guidelines.

Thus, to find someone, you look for the organization name in the `@c=US` area. This should give you back a single entry in the Directory, perhaps two or three at the most. You then look for that person in the area corresponding to that entry. To make this easier, the white pages user interface, *fred*, has a special command syntax which directs it to find out the names of the likely organizations and then search each one for the person you're looking for, automatically!

Of course, if you have the cycles and network bandwidth to burn, in theory there is nothing to stop you from simply going to the top of the tree and searching for the person. However, this is *very* resource-expensive, particularly in terms of time. Since time is probably the most valuable resource you have, it is worth it to issue two commands which complete quickly, rather than one command which may take hours.

There are two user interfaces provided with the pilot software. With the "simple" one, you follow this two-step process. With the "complicated" one, you can form *arbitrarily* complex queries to the Directory. Thus, if you want to type just one command and don't mind typing a bit more, you can still have an optimized search. Both of these interfaces will be introduced in due course.

1.3.3 Structure of Information

In addition to a handle, an entry in the WHOIS database consists of a *type*, which indicates what kind of user is recorded by the entry (e.g., a person); and, several *fields*, each containing a textual description.

For example, an entry for a person might look like:

```
Rose, Marshall T. (MTR)    mrose@nisc.nyser.net
    NYSERNet, Inc.
    Western Development Office
    420 Whisman Court
    Mountain View, CA  94043-2112
    (415) 961-3380
```

The first line contains both the handle and all fields available for searching. Here, the handle is `MTR`, and there are two fields available for searching: a name and a mailbox. The remainder of the entry is a textual annotation.

Because the Directory must accommodate many kinds of access from various users and programs. It is important that the information contained therein be highly structured. As noted earlier, this allows universal understanding of the information, and hence consistent interpretation. Fortunately, most of the information is represented by textual strings.

It is important to remember however that all information associated with an entry is contained in an attribute. This attribute has a type, describing both its syntax and semantics. For example, the `surName` attribute of a person has a textual string syntax and semantics corresponding to someone's last name.

How the information associated with an entry is displayed to you is *strictly* a function of the interface you use when talking to the Directory. The Directory will enforce all of the syntactic constraints associated with the attributes, but only the users of the Directory can assign meaning to the attribute semantics.

With this in mind, here's an entry associated with a person, as it might be displayed by a user interface:

```
Marshall Rose (3)           mrose@nisc.nyser.net
    aka: Marshall T. Rose

Senior Scientist
NYSERNet, Inc.
    Western Development Office
    420 Whisman Court
    Mountain View, CA  94043-2112

Telephone:    +1 415-961-3380
FAX:          +1 415-961-3282
```

```
Mailbox information:
    Internet: mrose@nisc.nyser.net
    UUCP:     nyser!mrose

Principal Implementor of the ISO Development Environment

Handle:       @c=US@o=NYSERNet Inc.@ou=Research ...
```

Of course, there are dozens of possible ways that this information could have been displayed. Or *not* displayed — for example, there are other attributes which the interface may not care (or be able) to display, such as access control information, passwords, and so on.

Appendix A on page 29 lists all of the attributes which may be present for a person participating in the pilot project.

1.4 Where to turn for help

The software supporting the pilot project is the QUIPU implementation of the OSI Directory, which is a part of the ISO Development Environment (ISODE).

QUIPU was originally developed as a part of the INCA project (under the auspices of the ESPRIT initiative of the EEC). The Inca of Peru did not have writing. Instead, they stored information on strings, carefully knotted in a specific manner with colored thread, and attached to a larger rope. Such a device was known as a *Quipu* (pronounced *kwip-ooo*). The encoding was obscure, and could only be read by selected trained people: the *Quipucamayocs*. The Quipu was a key component of Inca society, as it contained information about property and locations throughout the extensive Inca empire.

The pilot sponsors and the administrators of the participating organizations form the *camayocs* for the pilot project. There is a special mailing list

```
wpp-camayocs@nisc.nyser.net
```

which each *Camayoc* belongs to.

If you need help with the white pages service, you should first contact your site administrator. If you are unable to do so, then you should contact another *Camayoc* or drop a note to the `wpp-camayocs` list.

Chapter 2

The White Pages User Interface

The *fred*(1c) program is the "simple" interface for the white pages service. This is actually a small program which talks to the "complicated" interface, a program called *dish*. The *fred* program should be able to handle most of the queries you wish to make. However, it is purposefully limited in its power. As experienced users will find, *fred* is a great pair of crutches when you need help walking, but if you want to sprint, then you need a good pair of sneakers, called *dish*, instead.

2.1 Invoking Fred

When your local *Camayoc* told you about the white pages pilot, you should have been given three bits of information:

- your handle (a Directory Distinguished Name);
- a password for the handle (so you can keep your own entry up to date); and,
- the location of the *fred* program.

2.1.1 Fred resides locally

In most cases, the *fred* program will reside on a machine which you have a login on. This will be a UNIX[1] machine.

[1] UNIX is a trademark of AT&T Bell Laboratories.

Normally, the UNIX directory containing *fred* is already in your search path, e.g., */usr/local/bin/*. Type the command:

```
% which fred
```

If the *which* command says something like:

```
/usr/local/bin/fred
```

then *fred* is already in your search path: simply type

```
% fred
```

and skip to Section 2.2 on page 13.

Otherwise, if *which* says

```
fred not found in path
```

then take a look at the *.cshrc* or *.profile* file in your home directory to see if this is so. You should see a line such as:

```
set path=(. /bin /usr/ucb /usr/bin /usr/local/bin)
```

in your *.cshrc* file, or some lines similar to

```
PATH=.:/bin:/usr/ucb:/usr/bin:/usr/local/bin
export PATH
```

in your *.profile* file. Verify that the name of the UNIX directory containing fred is listed. If not, edit the file to make it so. Then, tell your shell to re-read the appropriate file, e.g., either

```
% source ~/.cshrc
```

from the *C* shell, or

```
% . $HOME/.profile
```

from the Bourne shell.

As they say in the trade, "you are now ready to rock and roll". Simply type

```
% fred
```

to name that tune. Now skip to Section 2.2 on page 13.

2.1.2 Fred resides remotely

If for some reason you do not have a login on the UNIX machine where *fred* lives, there are three other ways to use the *fred* program. These are described in order of descending preference.

via Guest account

Your *Camayoc* might provide an anonymous login to a machine where *fred* resides. In this case, you just use TELNET to the machine, using the information provided by the *Camayoc* who will have already initialized the environment. For example:

```
% telnet wp.nyser.net
login: fred

Welcome to the NYSERNet White Pages Pilot Project

  accessing service, please wait...

fred>
```

and away you go. Now skip to Section 2.2 on page 13.

If you are accessing the White Pages from an X Windows display, and if your TELNET connection is to a host running X client software, then you might see something like this instead:

```
% telnet wp.nyser.net
login: fred

Welcome to the NYSERNet White Pages Pilot Project

If you want X windows access, please enter your DISPLAY name,
otherwise, if you do not wish to use X windows, enter "none"

DISPLAY (default=myhost.nyser.net:0.0)=
```

If you intend to use X windows, then you should make sure that the machine you have a TELNET connection to is allowed to access your X display. This is usually accomplished by adding a line such as

```
xhost +wp.nyser.net
```

to your X Windows start-up file.

via WHOIS

The white pages service might also be available via the WHOIS network port. In this case, you use the *whois* program found on your machine, usually with a special option, e.g.,

```
    % whois -h wp.nyser.net "query"
```

Your *Camayoc* will provide the details. There may even be a command called *whitepages* which shortens type-in somewhat, e.g.,

```
    % whitepages "query"
```

Note that a key disadvantage of this approach is that it provides read-only access to the white pages service. You can never update your own entry using this technique.

via mail

Finally, the white pages service might also be available via electronic mail. In this case, you use the *mail* program found on your machine, and send a message to a special address, e.g.,

```
    whitepages@wp.nyser.net
```

Your query is placed in either the `Subject:` field or the body of the message. Your *Camayoc* will provide the details.

In addition to providing read-only access to the white pages service, this approach has the added disadvantage of not being interactive.

2.2 Giving commands to Fred

After invoking *fred*, you are prompted with "`fred> `" indicating that *fred* is ready. (Actually, this is only true if *fred* is invoked from a UNIX machine. If invoked via WHOIS or mail, then no prompts are given.)

If *fred* is invoked interactively, it will look for a file in your home directory called *.fredrc*. It will execute the commands contained in this file just as if you had typed them directly to *fred*. Following this, you are given the "`fred> `" prompt.

2.3 Let your fingers do the walking

Although *fred* has several commands, the most interesting command is `whois`, which performs a white pages query.

Let's begin with some simple examples and introduce the other commands along the way. If you already know the handle of the person you're interested in finding out about, just give the handle:

```
fred> whois @c=US@cn=Manager
Manager (1)

Handle:         @c=US@cn=Manager
```

2.3.1 The Alias Command

Since handles are long strings, *fred* will automatically maintain a list of aliases of the entries you have seen in the current session. The alias is always a number. When an entry is displayed, it appears on the first line in parenthesis after the name of the object. In the example above, the alias is `1`.

To find out what aliases are currently defined, use the `alias` command:

```
fred> alias
1    @c=US@cn=Manager
```

Thus, the previous `whois` command could have been shortened to simply:

```
fred> whois !1
Manager (1)

Handle:         @c=US@cn=Manager
```

Each time you invoke *fred*, its list of aliases is empty. If there are few handles which you use often, you might wish to define them in your *.fredrc* file, e.g.,

```
alias "@c=US@o=NYSERNet Inc.@cn=Manager"
```

Of course, the ordering of aliases is important. *fred* will start numbering from 1 starting with the first `alias` command.

2.3.2 Back to Searching

Suppose however, that you don't know the handle for the person. In this case, you need to specify some search parameters. Logically, the first step is to ascertain the organization which the person is likely to be associated with, e.g., "NYSERNet, Inc.". This is done as:

```
fred> whois organization nyser
NYSERNet Inc. (2)               +1 315-443-4120
  111 College Place
  Syracuse, NY 13244--1260
  US

FAX:       +1 315-443-1973

Not-for-profit organization providing network services and software

Locality:      Syracuse, New York

Handle:        @c=US@o=NYSERNet Inc. (2)
```

Second, to search for a particular person, you might use:

```
fred> whois rose -area 2
Marshall Rose (3)            mrose@nisc.nyser.net
    aka: Marshall T. Rose
    aka: mtr

Senior Scientist
NYSERNet, Inc.
    Western Development Office
    420 Whisman Court
    Mountain View, CA  94043-2112

Telephone:    +1 415-961-3380
FAX:          +1 415-961-3282

Mailbox information:
    Internet: mrose@nisc.nyser.net
    UUCP:     nyser!mrose
```

```
     Principal Implementor of the ISO Development Environment

     Handle:          @c=US@o=NYSERNet Inc.@ou=Research ...
```

Note the use of the alias 2. The command could also have been:

```
     fred> whois rose -area "@c=US@o=NYSERNet Inc."
          ...
```

Double-quotes are used so that the DN appears as a single token to *fred*.

Of course, this two-step process, whilst logical, is tedious. Thus, you can combine them like this:

```
     fred> whois rose -org nyser
          ...
```

which says to look for any organizations with “nyser” in its name. Then, for each of these, look for something called “rose”.

2.3.3 The Area Command

Suppose you want information on several persons belonging to an organization. You can use the `area` command, by itself, to tell *fred* where to search for subsequent commands. For example,

```
     fred> area "@c=US@o=NYSERNet Inc."
```

or simply

```
     fred> area 2
```

both tell *fred* the default area used by the `whois` command. Of course, you can still use the ‘`-area`’ area with the `whois` command to override the default area. Thus,

```
     fred> whois alan -area "@c=US@o=Columbia University"
```

will do what you expect.

If you use the `area` command without any arguments, *fred* will tell you what its default area is:

```
fred> area
@c=US@o=Yoyodyne
```

This indicates the default area for all commands, *including* any subsequent `area` commands. Thus, issuing:

```
fred> area @c=US@o=Yoyodyne
@c=US@o=Yoyodyne

fred> area ou=Research
@c=US@o=Yoyodyne@ou=Research
```

is equivalent to

```
fred> area @c=US@o=Yoyodyne@ou=Research
@c=US@o=Yoyodyne@ou=Research
```

because a leading '`@`'-sign was not used before `ou=Research`.

As you might expect, there is a special string "`..`" which may be used to move up one level:

```
fred> area ..
@c=US@o=Yoyodyne
```

Combinations are possible as well, such as:

```
fred> area ..@"NYSERNet Inc."
@c=US@o=NYSERNet Inc.
```

which moves up a level and then down to `o=NYSERNet Inc.`

2.3.4 Getting Help

For a brief summary of *fred* commands, type:

```
fred> help ?
```

This will list the commands that *fred* knows about along with a one-line summary of their function.

For help on a particular command, type the name of the command followed by '`-help`', e.g.,

```
fred> alias -help
```

If you need more help, try

```
fred> manual
```

which is the same as

```
% man fred
```

from the shell. If that's not enough, contact your local *Camayoc*.

2.3.5 Reporting problems

To report something to your local white pages manager, simply use the `report` command. You will be prompted for some text, which *fred* will send to the appropriate mailbox.

2.3.6 Quitting

To terminate *fred*, simply use:

```
fred> quit
```

2.3.7 Setting Variables

fred contains a few variables that may be manipulated to modify its behavior:

debug: debug *fred*

manager: mail address of local white pages manager

namesearch: type of named use for matching (either `fullname` or `surname`)

pager: program to use for terminal pagination

query: confirm two-step operations

soundex: use soundex for matching when no wildcards are present

timelimit: maximum number of seconds to spend searching

verbose: verbose interaction

watch: watch dialogue with *dish*

To view or change settings, use the `set` command:

```
fred> set query
```

will display the current value of the `query` variable, whilst

```
fred> set query on
```

will change the variable accordingly.

2.3.8 via WHOIS or mail

If you are accessing the white pages via WHOIS or mail, then only the `whois`, `area`, `help`, and `manual` commands are available. If your command to *fred* does not start with one of these keywords, then it is assumed that your input is a list of arguments to the `whois` command.

Note that it is not possible to use aliases with the `whois` command since your *fred* session is not interactive.

2.4 More on Searching

The full syntax to the `whois` command is:

```
whois input-field [record-type] [area-designator] [output-control]
```

Only the *input-field* component need be present. The components may appear in any order.

2.4.1 Input Field

This component tells the white pages who or what to look for.

name NAME or **.NAME**

surname NAME

fullname NAME gives the name of the target.

Searching for names follows these rules:

1. If the "*"-sign appears at the beginning and/or end of the name, then wildcard-style matching is used: the "*"-sign matches zero or more characters at the beginning or end of a name.
2. Otherwise, if soundex has been enabled (set the variable `soundex` to `on`), then imprecise matching occurs according to a Soundex algorithm.
3. Otherwise, if searching is to occur for a person's surname (either the keyword `surname` is used or the value of the variable `namesearch` is `surname`), then a case-insensitive match is used.[2].
4. Finally, as a last resort, *fred* will force a rather liberal wildcard-style match (e.g., "`NAME`" is treated as "`*NAME*`").

For compatibility with the WHOIS service, an input field of "`NAME.`" is equivalent to "`NAME*`" (i.e., a partial match for names having the given prefix). Similarly, an input field of "`*NAME`" is equivalent to "`NAME expand`". Thus, to have wildcard matching at the beginning of the name, use two "*"-signs, e.g., "`**inc`" matches names ending in "`inc`". (A terrible hack, but that's the price one pays to be consistent with the WHOIS service.)

handle HANDLE or **!HANDLE** gives the Distinguished Name (DN) of the target. Instead of a DN, an alias for the DN may be used.

mailbox STRING gives the mailbox of the target. (The mailbox search is more properly a Yellow Pages function. It is included solely for the backwards compatibility with the WHOIS service. As such, use of mailbox matching is not recommended.)

If a keyword is not given, then *fred* attempts to intuit which kind of input field is being provided. In most cases, *fred* will treat the input field as a name, unless it contains the '`@`'-sign, which makes it either a handle or a mailbox.

[2] For compatibility with the older WHOIS service, if `surname` searching is used, then a case-insensitive match is also performed on the local-part of the entry's electronic-mail address

2.4.2 Record Type

The *record type* is a single keyword, one of:

person: a person

organization: an organization

unit: a division under an organization

role: a role within an organization

locality: a geographic locality

dsa: a white pages server

If one of these keywords is not present, then *fred* will not know what kind of entry you are looking for and this can result in inefficient searches. If you know what kind of entry you're looking for, be kind and tell *fred*.

If you are searching for a person with a particular title, then rather than using `person`, use the '`-title`' switch. For example,

```
whois rose -title scientist
```

looks for someone named `rose` who is a scientist, whilst

```
whois -title operator
```

looks for anyone who is an `operator`. Naturally, these searches are carried out relative to the appropriate area.

More on the Area Command

If the `area` command is given two arguments then this defines the default area for a particular record type. For example,

```
fred> area unit @c=US@o=Yoyodyne
```

indicates that searches for organizational units should, by default, occur under the indicated area.

It is now time to confess to a little white lie told earlier. If the `area` command is invoked without arguments, it doesn't just print one line of information, it prints several:

```
fred> area
                     default area @c=US@o=Yoyodyne
area for record-type organization @c=US
area for record-type         unit @c=US@o=Yoyodyne
area for record-type     locality @c=US@l=NY
area for record-type       person @c=US@o=Yoyodyne
area for record-type          dsa @c=US
area for record-type         role @c=US@o=Yoyodyne
```

The first line tells what the default area is. The remaining lines tell what the default area is for particular kinds of searches.

These values are normally set by your local *Camayoc* in a system-wide file read by *fred* during initialization. The name of this file is probably */usr/etc/fredrc*, check with your system administrator. Since this file is read before your own *.fredrc* file, you can automatically override these settings if you choose.

So, what use does *fred* make of the record type you supply? At the moment, *fred* employs some rather simple heuristics. If a default area is declared for a `organization`, `unit`, `locality`, or `dsa`, then searches for objects of this type will look *only* one-level deep. This is an artifact of the rules used by the sponsors of the pilot project when they defined the overall structure of the Directory tree. In practical terms, this severely limits the search to the areas where it is possible for these objects to exist. This means that searching is faster since parts of the tree, believed to never contain entries of this type, are never searched.

Note however that in order to take advantage of this heuristic, not only must a default area be declared for the record type you are searching for, but you must also tell the `whois` command what the record type is. While this might seem obvious:

```
fred> whois organization nysernet
```

will, given the default areas defined above, invoke the search heuristic and result in a "lightning fast" search. However,

```
fred> whois nysernet
```

will result in a slow search since *fred* has no way of knowing that `nysernet` isn't someone's name unless *you* tell it.

2.4.3 Area Designator

The *area designator* takes one of two forms. The most common form is one of the switches:

'`-org`' (short for '`-organization`'),
'`-unit`', or,
'`-locality`'

followed by a name. For example,

```
fred> whois rose -org nyser
```

which was introduced earlier, says to look for any organization with "nyser" in its name. Then, for each of these, look for something called "rose".

In the second form, the area designator consists of the switch '`-area`' followed by a location in the white pages, either a Distinguished Name or an alias for a DN. If an area designator is not given, then the area for the search is chosen as follows:

1. If a *record type* was given for the search (always a good idea), and if a default area was declared for that record type, then the appropriate area is used.

   This also allows *fred* to use the special searching heuristic based on the way the Directory tree is structured under the pilot project.

2. Otherwise, the default area is used.

2.4.4 Output Control

Any combination of the following keywords may appear in the *output control* component:

expand: give a detailed listing and show children of matched entries.

full: give a detailed listing, even on ambiguous matches (the default is to give a detailed listing only if a single match is found.)

summary: give a one-line listing, even on unique matches (the default is to give a one-line listing only if a multiple matches are found.)

subdisplay: give a one-line listing and show children of matched entries (the default is to not show any children.)

2.5 Editing your Entry

Finally, you should keep your own entry as accurate and current as possible. This is your responsibility in order to add value to the white pages service.

The `edit` command is used to examine and modify your entry in the white pages.

2.5.1 Who are You?

Of course, the first step is to tell *fred* who you are so that it can find your entry. Recall that your local *Camayoc* indicated your handle and password earlier. You use the `thisis` command to tell the whitepages who you are, e.g.,

```
fred> thisis "@c=US@o=NYSERNet Inc.@cn=Manager"
Enter password for "@c=US@o=NYSERNet Inc.@cn=Manager": secret
```

You can place this command in your *.fredrc* file to automatically tell *fred* about your entry in the white pages. Note however that if you do so, then *fred* will also prompt you for a password whenever it starts. To get around this you might choose to place the password in your *.fredrc* file. e.g.,

```
thisis "@c=US@o=NYSERNet Inc.@cn=Manager" secret
```

However, you must now protect your *.fredrc* accordingly. Entering:

```
% chmod 0600 $HOME/.fredrc
```

will do the trick.

2.5.2 Editing the Entry

Once the *fred* knows about your entry in the white pages, to edit your entry, use the `edit` command:

```
fred> edit
```

This will invoke the editor defined by your `$EDITOR` shell variable on a template file. (If you do not have the shell variable `$EDITOR` set, then you will be prompted for the name of an editor to use.) Edit the template file accordingly and then exit the editor. You will be asked if you want to update your entry with the new template. If so, the white pages will be informed of the changes.

It is a good idea to take a look at your entry after editing it. To simplify type-in, there is a special option to the `whois` command which directs it to display the entry for you:

```
fred> whois !me
```

which isn't great grammar, but gets the job done.

Appendix A on page 29 lists the syntax and semantics of the attributes that you may have in your entry.

Chapter 3

The Directory User Interface

The *DIrectory SHell*, *dish*(1c), is the raw interface to the OSI Directory. Unless you intend to be a really sophisticated user of the white pages service, you should never need to use the *dish* program directly.

Chapter 4 of *Volume Five* describes *dish* in considerable detail. As such no further information is provided in *The Handbook*.

Chapter 4

Other Programs of Interest

In addition to the *fred* program, there are some other programs which may be available, depending on whether your system administrator has installed them.

4.1 Faces

When *fred* and *dish* display the entry for someone, they check to see if there is a photograph associated with the user. This is stored in facsimile format in the `photo` attribute for the entry. If a photograph is present, then a file called */usr/etc/dsaptailor* is consulted for directives indicating how the picture should be displayed (check with your system administrator for the exact pathname of this file).

If you are running the X windows system, then your administrator may have installed a program called *Xphoto*. If your terminal type is `xterm`, then this program will be automatically invoked whenever *fred* or *dish* wish to display someone's photo.

If you want your picture to be kept in the OSI directory, ask your system administrator if this is possible.

4.1.1 xwho

If your system is running the *rwhod*(8c) daemon, and you are running the X windows system, then you might want to run the *xwho* program to see who is logged in on your local network. Consult the *xwho*(1c) manual page for more details.

4.1.2 xface

If you use the MH system to read your mail, then you might want to run the *xface* program in the background. Whenever MH display a message for you, it will ask *xface* to display the picture of the person who sent the message.

4.2 Mail Composition

If you use the MH system to send your mail, then you can use the White Pages automatically to lookup the electronic mail addresses of the people you are sending messages to.

Rather than specifying an address, you can specify a name by bracketing a White Pages query between "`<<`" and "`>>`" using the `whois` syntax of the *fred* command, e.g.,

```
To: << rose -org psi >>
```

At the `What now?` prompt, you can say `whom` to have the names expanded into addresses. Alternately, the `send` option can be used as well. For each query appearing between "`<<`" and "`>>`", *fred* will be asked to perform a White Pages resolution. All matches are printed and the you are asked to select one. If one is not selected, you remains with *fred*, to make more queries, until eventually one is selected (or you exit *fred* to abort the expansion process).

Note that expansion can occur only if `whom` or `send` is invoked interactively. If the `push` option is used instead, then the expansion will fail because *fred* will be unable to ask you to select/confirm the right entry to use for the substitution.

Appendix A

Attributes for Persons in the Pilot Project

Here is the list of the attributes which may be present for a person represented in the NYSERNet White Pages Pilot Project.

There are two mandatory attributes:

commonName: which gives a (potentially ambiguous) name for the person. The value of this attribute is a string usually containing the person's first and last names; e.g.,

```
Marshall Rose
```

This attribute is usually multi-valued, containing variations on the first, middle, and last names; e.g.,

```
Colin Robbins
Colin John Robbins
Colin J. Robbins
```

For purposes of the pilot project, the distinguished value of the attribute should contain only the person's first and last names.

surName: which gives the person's last name. The value of this attribute is a string; e.g.,

```
Rose
```

There are several attributes that may be present, which are divided into five groups: physical address, telecommunication information, computer environment, miscellaneous information, and home information.

The first group describes the physical address of the object:

postalAddress: which describes how physical mail is addressed to the object. The syntax of this attribute's value is special: it consists of 1 to 6 fields, seperated by the "`$`"-sign, each field being from 1 to 30 characters long; e.g.

```
NYSERNet Inc. $ 165 Jordan Road $ Troy, NY 12180
```

registeredAddress: which defines how registered physical mail is addressed to the object. The syntax is identical to that of the `postalAddress` attribute.

roomNumber: which is a string describing where the object resides at the location, e.g.,

```
Building T-30
```

streetAddress: which is a string describing where the object physically resides (i.e., the street name, place, avenue, and building number); e.g., object

```
165 Jordan Road
```

This need have no relationship to the object's postal address.

postOfficeBox: which is a string describing the box at which the object will receive physical postal delivery; e.g.,

```
1010
```

physicalDeliveryOfficeName: which is a string describing the geographical location of the physical delivery office which services the postal address of this object; e.g.,

```
Troy
```

stateOrProvinceName: which is a string describing the state in which the `locality` is found; e.g.,

```
New York
```

postalCode: which is a string containing the ZIP code; e.g.,

```
12180
```

or

```
94043-2112
```

localityName: which is a string describing the geographical area containing the `streetAddress`; e.g.,

```
Troy, New York
```

The second optional group describes telecommunications addressing information for the object.

telephoneNumber: which is a string describing the phone number of the object using the international notation; e.g.,

```
+1 518-283-8860
```

facsimileTelephoneNumber: which is a string describing the fax number of the object using the international notation; e.g.,

```
+1 518-283-8904
```

telexNumber: which is defines the TELEX address of the object in a three-part string:

```
number $ country $ answerback
```

e.g.,

```
650 103 7390 $ US $ MCI UW
```

The third group describes information relating to the person's computer environment:

rfc822Mailbox: which is the user's computer mail address, e.g.,

```
mrose@nisc.nyser.net
```

otherMailbox: which is the user's computer mail address in various domains. The syntax of this attribute's value is special:

```
<domain> $ <mailbox>
```

e.g.,

```
internet $ mrose@nisc.nyser.net
```

or

```
uucp $ nyser!mrose
```

userid: which is the user's login name; e.g.,

```
mrose
```

userClass: which describe's the user's classification; e.g.,

```
staff
```

The next optional group contains a few miscellaneous attributes:

description: which is a simple textual description; e.g.,

```
Principal Implementor of the ISO Development Environment
```

info: which is additional information about the object; e.g.,

```
It's nearly as good as BIND
```

businessCategory: which describes the person's business, e.g.,

```
networking
```

title: which is the person's job title, e.g.,

```
Senior Scientist
```

userPassword: which is a string containing the object's password in the Directory. This is used, for example, when the user wants to update the entry. The password is kept in the clear; e.g.,

```
secret
```

mobileTelephoneNumber: which is a string describing the user's mobile number (e.g., for a cellular phone).

pagerTelephoneNumber: which is a string describing the user's pager number.

favouriteDrink: which is a string describing the user's favorite drink.

secretary: which is the Distinguished Name of the user's secretary.

seeAlso: which is a Distinguished Name pointing to another entry related to the user (perhaps in a different role).

photo: which is a facsimile bitmap of the user's face.

The final optional group contains a few attributes about the person at home:

homePostalAddress: which describes how physical mail is addressed to the person's home. The syntax of this attribute's value is special: it consists of 1 to 6 fields, seperated by the "$"-sign, each field being from 1 to 30 characters long; e.g.

```
NYSERNet Inc. $ 165 Jordan Road $ Troy, NY 12180
```

homePhone: which is a string describing the phone number of the object using the international notation; e.g.,

```
+1 518-283-8860
```