in the directory **BINDIR** and thus the user needs to include this path to the shell variable **PATH**. Then *fred* starts simply:

```
% fred
fred >
```

Now *fred* is ready to accept commands. When *fred* starts, it first reads the *$(ETCDIR)/fredrc* file and after that the file *$(HOME)/.fredrc*. The *$(ETCDIR)/fredrc* file defines the default areas where the searches are allowed. The user can override these values in *$(HOME)/.fredrc*.

Additional help is available with '?' sign. In addition to this the “manual” command lets you read the manual page.

It is also possible to make *fred* available via TELNET or e-mail. Consult [NYSERNet] for further instrcutions

WIDGET

Widget (Window Interface Developed at GEc hirsT) is a simple termcap based Directory User Agent. It is included only as an example. *Widget* supports:

- bind
- read
- list
- search
- modify

Widget splits the user's screen up into a number of small areas called “widgets”. Each widget is used to enter information.

The widget interface is invoked with the *widget* command. Help is given at any stage by typing '?'. The usage is then clear and easy.

SUNINT

This interface is also window based. It uses the feature of the SUNVIEW environment to give the user a set of windows and menus with which he/she can make queries to the Directory. Like *widget*, *sunint* only gives access to a subset of the X.500 protocol, and as such is a read only interface.

Appendix C: User Interfaces

The Directory user interfaces which are included to the ISODE package are described here shortly. For further information consult [SKill89b]. All the interface programs mentioned here are found in the subdirectory *others/quipu/uips* within the ISODE package.

DISH

Dish can be either invoked as a single process or as individual shell commands. This latter style allows *dish* to be used with other tools to provide access to the Directory. To run *dish* in interactive mode, invoke it, and then issue '?' sign to see what commands are available:

```
% dish
Dish-> ?
• • •
```

When dish starts, it first reads the *$(HOME)/.quipurc* file and executes commands according the definitions in the file. See [SKill89b] for tailoring this file. This mode of *dish* usage is especially useful for novice users.

When you start *dish* with the ´-pipe´ flag, you can give *dish* commands (see Section 3.6) directly at the shell level. Note that “dish -pipe &” has the same effect as the “bind” command.

Each *dish* command has a large number of flags. The shortest unique name is taken for the command name (e.g. “l” for “list”). The ´-help´ flag can be specified to get a short description of each command (e.g. “dsacontrol -help”).

FRED

The usage of the *dish* can be quite difficult to the ordinary user. Therefore, you community might prefer *fred w*hen binding to the Directory. *Fred* resides

Appendix B: Information about DSAs in the NORDUNET countries

COUNTRY	COUNTRY_DSA	MANAGER	FTP_HOST	FTPDIR
Denmark	—	—	—	—
Finland (FI)	Jaguar	Directory-Manager @funet.fi	funet.fi	networking/isode
Iceland (IS)	Elephant Seal	Directory-Manager @rhi.hi.is	isgate.is	pub/isode
Norway (NO)	Electric Eel	Directory-Manager @nac.no	nac.no	pub/isode
Sweden (SE)	Hummingbird	Directory-Manager @sunet.se	sunic.sunet.se	pub/isode

Appendix A: DSA Names

South American Wildlifes and Plants

Agouti	Alpaca	Anaconda
Giant Anteater	Armadillo	Brazilian Three-banded Armadillo
Giant Armadillo	Axolotl	Fruit Bat
Vampire Bat	Spectacled Bear	Boa Constrictor
Capybara	Andean Cat	Little Spotted Cat
Cayman	Long-tailed Chinchilla	Short-tailed Chinchilla
GreaterPichi Ciego	Lesser Pichi Ciego	Condor
Andean Condor	Argentinian Pampas Deer	Marsh Deer
Bush Dog	Small-eared Dog	Electric Eel
Guanaco	North Andean Huemul	South Andean Huemul
Hornero	Hummingbird	Iguana
Jaguar	Jaguarundi	Llama
Macaw	Amazonian Manatee	Caribbean Manatee
Margay	Buffy-headed Marmoset	Buffy-tufted-ear Marmoset
Goeldi's Marmoset	Tassel-eared Marmoset	White Marmoset
Brown Howler Monkey	Black Spider Monkey	Brown-headed Spider Monkey
Woolly Monkey	Yellow-tailed Woolly Monkey	Ocelot
Geoffroy's Spider Monkey	Long-haired Spider Monkey	Woolly Spider Monkey
Giant Brazilian Otter	La Plata Otter	Marine Otter
Southern River Otter	Pangolin	Chacoan Peccary
Brown Pelican	Galapagos Penguin	Rockhopper Penguin
Piranah	Thin-spined Porcupine	Nothern Pudu
Puma	Rhea	Southern Bearded Saki
White-nosed Saki	Elephant Seal	Galapagos Fur Seal
Juan Fernandez Fur Seal	Brazilian three-toed Sloth	Maned Sloth
Bare-face Tamarin	Cotton-top Tamarin	Emperor Tamarin
Golden Lion Tamarin	Golden-headed Lion Tamarin	Golden-rumped Lion Tamarin
White-footed Tamarin	Central American Tapir	Mountain Tapir
Masked Titi	Giant Tortoise	Toucan
Black-headed Uakari	Red Uakari	Red and White Uakari
Vicuna	Maned Wolf	

The References

[OSI-Services] Christopfer W. Moore, *OSI Directory Services*. The Wollongong Group, Inc, December 13, 1988.

[ISO88] *Information Processing Systems — Open Systems Interconnection — The Directory — Overview of Concepts, Models, and Service*. International Organization for Standardization and International Electrotechnical Committee, December, 1988. International Standard 9594-1.

[Skill89a] Stephen E. Kille, *The Design of QUIPU*. Research NoteRN/89/19, Department of Computer Science, University College London, March, 1989.

[Skill89b] Stephen E. Kille, Colin J. Robbins, and Alan Turland, *The ISO Development Environment: User's Manual, Volume 5: QUIPU*.November 17, 1989.

[NYSERNet] Marshall T. Rose, NYSERNet, Inc. *NYSERNet White Pages Pilot Project: Administrator's Guide*. September 1, 1989.

[JH] Juha Heinänen, *Introduction to the OSI Directory*. Software Systems Laboratory, Tampere University of Technology. NORDUNET Conference, October 25-27 1989.

[Ros89] Rose, M.T., *User's Manual* (version 5.0). The ISO Development Environment, March 1989.

members.

You should consult [SKill89b] for further instructions.

4.5.4 Publicly readable attributes

There are several attributes which should be readable by everyone (e.g. other DSAs). This is because of the DSA builds a connection based on these attributes. If the DUA reports “Unavailable”, the permissions might be defective. But remember first that other reasons for failures exists (e.g. network problems or the remote computer is down).

The following attributes should be publicly readable:

- masterDSA
- slaveDSA
- presentationAddress
- userCertificate
- treeStructure

The userPassword attribute should be comparable by everyone, but not readable. For the usage of these attributes refer [SKill89b].

4.6 Retrospect

A lot of information about Directory can not be found in this guide. The main purpose here was to give you qualifications for learning the basics of the Directory. The very final work is done however by learning and experimenting, and learning and experimenting again. This is not the most pleasant way but other ones do not exist. Have fun with the Directory!

(for example “quipu”). If this is not possible, it should run under the userid of the DSA manager. We refer this userid as “quipu” in the documentation.

4.5.2 File permissions

The EDB files are where QUIPU stores the contents of the Directory. They should have mode 0600 and the owner of the files should be “quipu”. The log files should be changed to the same mode as EDBs. However, ISODE’s logging facility usually makes log files writable by everybody. To be on the safe side, make a separate directory for QUIPU’s logs (*/usr/tmp/quipu-logs*, for example). Then make this directory only accessible by the “quipu” userid, and change the tailoring parameter logdir so that the logs are written there.

4.5.3 The Access Control List attribute

The “acl” (Access Control List) attribute controls entries authentication. Here is two emaples — one for non-leaf and another for leaf entry:

(1) The leaf entry (a person, for example) may have an acl like

```
acl= others # compare # attributes # acl $ userPassword
acl= group # <Manager's RDN > # write # attributes # acl
acl= self # write # attributes # userPassword
```

which means: The “acl” attribute is comparable by everyone, but only the Manager can change it. Also, the “userPassword” is comparable by everyone, but only the user can change it

(2) For the non-leaf entry (an organizational unit, for examaple) the acl could be

```
acl= others # read # child
acl= group # <Manager's RDN > # write # child
acl= group # <Manager's RDN > # write # entry
acl= others # read # entry
acl= group # <Manager's RDN > # write # default
acl= others # read # default
```

which implies: Everyone can search the members in the subtree of this entry, but only the Manager is allowed to add or delete

ifications to the database. Perhaps there will be some tools for updating the database in the next releases. So you must decide yourself how you are going to keep the database up the date.

One solution for this might be to use the internal phone book of your organization. You could pick up from the phone book at the regular intervals certain data (e.g. telePhoneNumber, homePhone and so on) and insert them to the database. The users of your organization could have the possibility to update other attributes (e.g. rfc822mailbox). Remember that even when an attribute is updated automatically from the phone book, the user can still add multiple values to the attribute.

This was only an example. The updating mechanism depends a lot on what kind of information your organization has available.

4.5 Common security features

Next, we pay attention to the security of the software and its usage. Consider the great number of entries held in the database: without due permissions anyone can manipulate the information of the DIT! Therefore, every user having an entry in the DIT has a DN and a password. The Directory can also be used by the users who do not have a DN in the DIT. Before allowing these users to bind the Directory, you should make sure the DIT has appropriate permissions (e.g. userPasswords are not visible to everyone).

4.5.1 The software

A DSA should not run under the userid of the super-user. The reason for this is trivial: any application should not be run as root unless it is absolutely necessary. New applications might have undetected bugs and then cause malfunctions (important files might be deleted if the processes are running as root). When running as an ordinary user, a process will be halted if it starts to misbehave. Ideally, the DSA should be assigned a special userid to run under

unit —"organizationalUnit" entry

These examples are in the directory *$(ETCDIR)/quipu/templates*. Every example entry contains editing instructions for building up a similar new entry.

4.3.1 Adding entries with *dish*

First, bind to the Directory as a manager and move to the node in the DIT where you want to do modifications:

```
% dish -c "own_dsa" -user "c=$(COUNTRY)@o=ORG@cn=Manager"
Enter password for "c=$(COUNTRY)@o=ORG@cn=Manager": secret
Dish -> moveto ...
```

You can now use "add" and "modify" commands the way you like. When you add entries for the first time, it is advisable to start with a template. For example:

```
Dish -> add ou=ORG_UNIT -template $(ETCDIR)/quipu/templates/
unit
```

The previous command creates new organizational unit below the current node. This way you can add any new objects to the DIT. If there are no template for your needs, you can create one or just start from scratch. When you are adding new entries, make sure that the MASTER copy of the node where you are doing additions allows you to create new entries. Also check always that the attribute "treeStructure" in the parent node allows you to use the objectClass you are creating.

4.4 Maintaining the database

When you have entered all the entries you wanted to the database, you must maintain the database, too. The simplest way is to use *dish* command "modify" or "modifyrdn". The command "modify" allows you the update the entry itself whilst with "modifyrdn" you can change the RDN of the entry.- Detailed information from *dish* commands you find in [SKill89b].

The ISODE software does not offer yet any tools for making larger mod-

4.2 The logfiles

The software makes with its default parameters a lot of logdata. For the administrator the most interesting file is *$(LOGDIR)/dsap.log* which contains common data about the load of the DSA. The second useful logfile is *$(LOGDIR)/stats.log* which includes statistical information about the DSA.

It is defined in the *dsaptailor* file how each logfile will be updated. You can change the levels and sorts of logs by editing the *dsaptailor* file or directly by using *dish*. For example, the following command changes the loglevel of the *dsap.log* file to “all”:

```
Dish -> dsacontrol -tailor "dsaplog level=all"
```

The logfiles have a common form:

<record> ::=<month>"/"<day><time><process><pid>"("<userid>")"<event>

where

- <time> is the current time in format hh:mm:ss
- <process> tells the name of the program for <event>
- <pid> is the process number
- <userid> is the id number of the user who started the process
- <event> is the actual logevent. Further details are explained in [SKill89b]

4.3 Adding an entry to the database

Small changes to the database are reasonable to make with *dish*. If you have a large collection of data to be updated, you might prefer editing the EDBs manually (for the present there are no tools to help you). When modifying EDBs, you must tell the DSA to reload its EDBs by either killing and restarting the DSA or giving the argument “refresh” the to *dish* command “dsacontrol”. To ease your work there are a couple of templates for building new entries:

alias —”alias” entry

person —”PilotPerson” entry

role —”organizationalRole” entry

```
# GB SLAVE at UNott
dsa_address "Vampire Bat"   ’025c’H/Internet=128.243.20.2+17003|Int-X25(80)=23426020017299+PID+03018100

# SE MASTER at UME
dsa_address "Hummingbird"       ’0101’H/Internet=130.239.1.15+2005|Int-X25(80)=240200100306

# IS MASTER
dsa_address "Elephant Seal"     ’0101’H/Internet=130.208.165.63+17003

# NO MASTER hold by UNINETT
dsa_address "Electric Eel"          ’0101’H/Internet=129.240.2.40+2005

# US MASTER at NYSERNet Inc.
dsa_address                         "Alpaca"                        ’0101’H/Internet=130.117.128.2+17007|Int-
X25(80)=31106070013600+PID+03018100

# US SLAVE at NYSERNet Inc.
dsa_address       "Fruit Bat"       ’0101’H/Internet=130.117.128.3+17007

# logging
dsaplog             level=notice file=dsap.log sflag=zero sflag=tty size=30

# local DIT position (where a DUA starts in the DIT)
local_DIT       "@"

# oid format
oidformat       short

# photograph process routines
photo               dumb        ttyphoto

# automatic quipurc creation
quipurc             off

# search/list sizelimit
sizelimit       20
```

Figure 7. cont.

```
################################################################################
#
# dsaptailor - ISODE DSAP Specific Tailoring File
#
# $Header: /f/osi/dsap/dua/RCS/dsaptailor,v 6.2 89/07/05 15:19:52 mrose Exp $
#
#
# $Log:       dsaptailor,v $
# Revision 6.2  89/07/05  15:19:52  mrose
# 5.2
#
# Revision 6.1  89/06/08  13:55:15  mrose
# touch-up
#
# Revision 6.0  89/06/07  22:07:26  mrose
# 5.1
#
# Revision 6.1  89/03/23  22:29:21  mrose
# out-the-door
#
# Revision 6.0  89/03/18  23:28:00  mrose
# Release 5.0
#
################################################################################


# this line must occur first
oidtable        oidtable

# which dsa to call initially
dsa_address       "rhea"              '0101'H/Internet=128.214.1.2+17003

# other DSAs of interest

# FI MASTER at Tampere
dsa_address     "Jaguar"          '0101'H/Internet=128.214.1.1+17003|Int-X25(80)=24432315101520+PID+03018100

# ROOT/GB MASTER at University College London
dsa_address     "Giant"       '0101'H/Internet=128.16.5.31+2005|Int-X25(80)=23421920030045

# AU MASTER at CSIRO
dsa_address     "Anaconda"  '0101'H/Int-X25(80)=5052334300013+PID+03018100

# DE MASTER at FOKUS
dsa_address     "Puma"      '0101'H/Int-X25(80)=26245300048207+PID+03018100
```

Figure 7. An example of the *dsaptailor* file

try). If the format is “long”, the whole definition of an attribute will be printed, (e.g. joint.ds.attributeType.country). The value can be printed in the numeric format with argument “numeric”, then the result could, for example, be 2.5.4.6.

E: local_dit: The argument is the DN which points to the place in the DIT where the user is placed when he is making a connection (e.g. default “moveto” in the DIT when starting the DUA).

F: dsaplog: Controlling log files. The syntax of the argument is:

key=value

The keys are:

- file: the name of the logfile
- size: the maximum size of the logfile
- level: the level of the logs produced

  The values for the key “level” are:

  – fatal: fatal errors
  – exceptions: serious errors (possibly temporary)
  – notice: the usual loglevel
  – trace: program tracing
  – pdus: Protocol Data Unit tracing
  – debug: full trace
  – all: includes all the previous values

- dlevel: the opposite to the previous point (the logs are not to be produced according the given values)
- dflag/sflag: setting special flags (dflag = on, sflag = off)

  The values for the key “dflag/sflag” are:

  - close: close the logfile after every entry made to the log
  – create: create the logfile if it does not exists
  – zero: truncate the logfile when it becomes too large
  – tty: copy the logentries to the terminal

In the Figure 7 there is an example of the *dsaptailor* file. By examining this file you can modify your own *dsaptailor* file. Hints for using the different keys are found on the manual page *quiputailor(8)*.

4. Maintenance

In this chapter we go through some points concerning the maintenance of the DSA. The following instructions are not going to be all covering but rather their goal is to work as a reference. Detailed description of the software is given in [SKillb89]. I recommend you to become familiar with this publication later at a later phase to get a more precise view about the functions of a DSA.

4.1 Editing the *$(ETCDIR)/dsaptailor* file

The *dsaptailor* file has effect on how the DUA works. This file contains the run time configuration for the DUA. You can change the configuration by editing the *$(ETCDIR)/dsaptailor* directly or by using the *dish* command "dsacontrol". Next, we look what kind of different configuration options there are:

A. oidtable: The path for *oidtable.** files. It is very important that this entry appears first in the *dsaptailor* file. The later entries might have arguments which will be searched from these files.

B. dsa_address: Defines the address of an DSA which will be used when the DUA is making a connection. If there are several addresses the first one will be used when not specified in the command.

C. sizelimit: Tells the maximum number of entries a successful "list" or "search" command gives.

D: oidformat: Defines how objectClasses will be printed. The default argument is "short" when the short key will be printed (e.g Coun-

If your *rc.local* file starts *tsapd(8c)*, place this entry after the one which starts *tsapd*. The next time your system reboots,check that also your DSA starts.

Congratulations! Your Level-1 DSA has now joined the Directory. If you have problems, it is worth thinking first critically what really has been done. If the problems still remains unresolved, make a clear report to the country manager.

When you have delivered the information about your DSA to the country manager, make sure the country level DSA recognize them (this is done by the country manager also, but do it anyway):

```
% dish -c "$(COUNTRY_DSA)"
Welcome to Dish (DIrectory SHell)
Dish -> moveto "@c=$(COUNTRY)@o=ORG"
Dish -> list
•••
Dish -> list "ou=ORG_UNIT"
•••
```

If the "list" commands prints out information about your organization, you can be sure the country DSA is able to talk to your DSA. The other Level-1 DSAs do not necessarily recognize the existence of your DSA yet, but as soon as the updates from the country level DSA spread they learn to know your DSA.

3.15 Controlling the DSA

When everything looks fine, leave the DSA running to the background. Use *dish* to abort the DSA:

```
% dish -user "@c=$(COUNTRY)@o=ORG@cn=Manager"
Welcome to Dish (DIrectory SHell)
Enter password for "@c=$(COUNTRY)@o=ORG@cn=Manager":
Dish -> dsacontrol -abort
*** Problem with DSA ***
Dish -> quit
```

Then edit the script *startup.sh* the way you like (at the same time make sure that the directory definitions are correct). Then run the script:

```
% $(ETCDIR)/quipu/own_dsa/startup.sh
```

Also, make sure that the loglevels are reasonable in *$(ETCDIR/quipu/own_dsa/quiputailor* and *$(ETCDIR)/dsaptailor* files. At the beginning it is usually meaningful to keep the level "all".

Finally it is time for the last bit of administration. Add an entry to the file */etc/rc.local:*

```
if [ -d $(ETCDIR)/quipu/own_dsa ]; then
   $(ETCDIR)/quipu/own_dsa/startup.sh & \
   (echo -n ' own_dsa') > /dev/console
fi
```

it. When you are satisfied with your modifications send it the country manager (see the addresses in Appendix A). Every time you make modifications to the *c=$(COUNTRY)/EDB* file, you must let the country manager know about that. He updates then the MASTER copy of the country level EDB file as you wish. When you are beginning to learn the usage of the attributes, you can of course change the *c=$(COUNTRY)/EDB* file and see if the DSA can load its EDBs after modifications. It is useful to pay special attention to the attributes "eDBinfo" (the update mechanism), "acl" (the protection mechanism), "objectClass" (which attributes are allowed in an entry) and "treeStructure" (the allowed objectClasses in the children of this entry).

3.14.1 Updating SLAVE copies

The update mechanism can be asynchronous or synchronous. QUIPU 6.0 uses asynchronous mechanism and having thus its own internal timing.

You can check manually if you can update your SLAVE copies. This assumes that the country manager has updated the MASTER copy of the country level EDB for your organization. Here it goes:

```
% dish -user "@c=$(COUNTRY)@o=ORG@cn=Manager"
Welcome to Dish (DIrectory SHell)
Enter password for "@c=$(COUNTRY)@o=ORG@cn=Manager": secret
Dish -> dsacontrol -slave
Done
Dish -> quit
```

Note that you must first log in as a manager to get the necessary access rights. You may look now at the root and country level portions of the DIT and search *EDB.bak* files. If there was information to update, backup files were created. Your DSA compares the date of SLAVE EDB (this is the second line in every EDB file) with the original MASTER EDB file. The SLAVE EDB file is updated only if the date differs. The update further requires that the option "update" in *quiputailor* is "on". Otherwise EDBs can be updated manually by hand.

The attribute “eDBinfo” tells the DSA how SLAVE copies are updated.The syntax of the attribute is:

eDBinfo= EDB concerning # where to get # where to send

The first field defines the level of the EDB we are interested in. If this field is empty the EDB is the root level EDB. Second field tells the DN of the DSA where the EDB is fetched. Third field defines the DN of the DSA where to send the EDB. For example, in the Figure 6 the “eDBinfo” attributes of the DSA Rhea are as follows:

Attribute	Function
First	Get ROOT EDB from cn=jaguar
Second	Get c=FI/EDB from cn=jaguar
Third	the DSA cn=Rhea holds the MASTER EDB of c=FI/o=Tampere University of Technology/EDB

You should have similar three lines in the entry of your DSA. Make sure you have them in the file *c=$(COUNTRY)/EDB*. If they are not correct , you are not able to get new copies of the root and country level EDBs. We will not concentrate here on the structure of the other attributes in the previous page. Note, that the values of the attributes depends on how you have filled the *$(ETCDIR)/quipu/own_dsa.dsa* file before running *dsaconfig*. Because your country level EDB (*c=$(COUNTRY)/EDB*) is a SLAVE copy (the MASTER copy is held by country level DSA), you should send a copy of your *c=$(COUNTRY)/EDB* file to the country level manager so that he can add your organization entry to the country level MASTER EBD. You may edit your *c=$(COUNTRY)/EDB* file to add or change information before sending

```
SLAVE
890901084500Z
cn=Rhea
quipuVersion= 6.0
eDBinfo= #cn=jaguar#
eDBinfo= c=FI#cn=jaguar#
eDBinfo= c=FI@o=Tampere University of Technology##
acl= others # read # entry
acl= group # c=FI@o=Tampere University of Technology@cn=Manager # write # entry
acl= others # read # default
acl= group # c=FI@o=Tampere University of Technology@cn=Manager # write # default
acl= self # write # attributes # userPassword$accessControlList
acl= others # compare # attributes # userPassword$accessControlList
lastModifiedBy= c=FI@cn=Manager
lastModifiedTime= 891121121247Z
manager= c=FI@o=Tampere University of Technology@cn=Manager
userPassword= {CRYPT}QKFB
supportedApplicationContext= X500DSP & X500DAP & quipuDSP
presentationAddress= '0101'H/TELEX+00728722+RFC-1006+03+128.214.1.2+17003|X121+24432315101521
description= The Rhea DSA located at Tampere Univ of Technology, Finland holding the TUT bit of the DIT.
ou= Software Systems Laboratory
o= Tampere University of Technology
l= Hervanta, Tampere
cn= Rhea
objectClass= dSA & applicationEntity & top & quipuDSA

o=Tampere University of Technology
masterDSA= c=FI@cn=Rhea
acl= others # read # entry
acl= group # c=FI@o=Tampere Universtity of Technology@cn=Manager # write # entry
acl= others # read # default
acl= group # c=FI@o=Tampere University of Technology@cn=Manager # write # default
acl= others # compare # attributes # userPassword$accessControlList
treeStructure= applicationEntity & applicationProcess & groupOfNames & organizationalUnit &
alias & domainRelatedObject & quipuNonLeafObject & quipuDSA
associatedDomain= tut.fi
lastModifiedBy= c=fi@cn=manager
lastModifiedTime= 891027130309Z
userPassword= {CRYPT}WVW
telephoneNumber= +358-31-162111
postOfficeBox= P.O. Box 527
postalCode= SF-33101
postalAddress= Tieteenkatu 21
o= TUT
o= TTKK
o= Tampereen teknillinen korkeakoulu
o= Tampere University of Technology
streetAddress= Tieteenkatu 21
stateOrProvinceName= {T.61}  Hämeen lääni
l= Hervanta, Tampere
objectClass= organization & top & domainRelatedObject & quipuNonLeafObject & quipuObject
```

Figure 6. An DIT entry.

in section “Editing the *dsaptailor* file”.

After all this we are ready to start the DUA:

```
% dish -c "own_dsa"
Welcome to Dish (DIrectory SHell)
Dish ->
```

If you are unable to contact, return to the section “DUA Failure” for further instructions. Next, try to identify yourself as a manager:

```
Dish -> bind -user "@c=$(COUNTRY)@o=ORG@cn=Manager"
Enter password for "@c=$(COUNTRY)@o=ORG@cn=Manager": secret
Dish ->
```

This indicates that you have bound to the directory with that DN (you can make it sure with the “squid”-command to *dish*). The password you type is the same you defined in *$(ETCDIR)/quipu/own_dsa.dsa*. Instead, if the DUA prints:

```
Dish -> bind -user "@c=$(COUNTRY)@o=ORG@cn=Manager"
Enter password for "@c=$(COUNTRY)@o=ORG@cn=Manager":
Security Error - check name and password
```

you have mistyped your password or the DSA does not recognize the entry. Look at the *EDB*s to check that the entry really exists. If you encounter other problems, contact the country manager.

3.14 Joining the global Directory

Your DSA needs to know how it can get the root and country level EDBs. The DSA needs this information in order to it know its location in the DIT. In addition your DSA has to transmit its own locally held organization level EDBs to the DSA one level up (**$(COUNTRY_DSA)**). This is done so that other DSAs know the existence of your DSA. On the next page there is an example of an entry concerning a DSA (Figure 6.). The example contains entry of the DSA and entries of the organization for which it is the MASTER DSA.

You can use these entries in Figure 6 as a basis when you fill the *EDB* files for your organization.

tening on the IP address and TCP port of the DSA, your DSA will print for example:

```
– ’0101’H/Internet=130.117.128.3+17003 –
TNetListen: [Congestion at TSAP] start_tcp_server
failed:
        Address already in use

FATAL ERROR: Couldn’t initialise the network moni-
tor!!
```

Perhaps you have already a copy of this DSA running or the TCP port is wrong (look at the *own_dsa/c=$(COUNTRY)/EDB* file). It is also possible that an earlier invocation of the DSA was unsuccessfully terminated and left the TCP port in a FIN_WAIT-X state. Use *netstat(8C)* to find out.

You might also see a message like this:

```
– ’0101’H/Internet=aaa.bbb.ccc.ddd+17003 –
TNetListen: [Congestion at TSAP] start_tcp_server
failed:
        Can’t assign requested address

FATAL ERROR: Couldn’t initialize the network moni-
tor!!
```

You have defined wrong the IP address of your DSA in your *wildlife.dsa* file.

3.13 Contacting your DSA

Now you have set up your own DSA. To make it available for contacts, edit the file *$(ETCDIR)/dsaptailor*. Look at the lines:

```
# the level-1 DSA
#dsa_address ....
```

Remove ’#’-sign at the beginning of the second line. This tells the DUA that your DSA is reachable, too. Next, look at the lines:

```
local_DIT "c=$(COUNTRY)"
#local_DIT "c=$(COUNTRY)@c=ORG"
```

Remove the first line and the ’#’-sign at the beginning of the second one. Now when you start the DUA, it moves directly to the part of your organization in the DIT. More information about the *dsaptailor* options can be found

again. Here are some common error situations:

A. Fatal error - Attribute Error

If you have misspelled the name of an attribute, the output might be:

```
line 13: unknown attribute type ...
File .../own_dsa/c=$(COUNTRY)/o=ORG/EDB not loaded
FATAL ERROR: DSA Halted
```

If you specify an illegal attribute for an object, the diagnostic will look like:

```
Schema error in entry ending line 17...
*** Attribute error ***
<<DN of entry in error>>
Attribute type<<illegal-name>> - Constrain violation
File ...own_dsa/c=$(COUNTRY)/o=ORG/EDB not loaded
FATAL ERROR: DSA Halted
```

If you leave out an attribute which should have been in an entry:

```
Schema error in entry ending line 16...
*** Attribute error ***
<<DN of entry in error>>
Attribute type <<mandatory-name>> - No such attribute
in the entry
File ...own_dsa/c=$(COUNTRY)/o=ORG/EDB not loaded
FATAL ERROR: DSA Halted
```

B. Fatal error — Structure Error

These errors are generated when an objectClass is not allowed in an entry. The allowed objectClasses are defined using the attribute “treeStructure”. The output might then be:

```
Schema error in entry ending line 16...
*** Attribute error ***
<<DN of entry in error>>
Attribute type objectClass - Constrain violation
File ...own_dsa/c=$(COUNTRY)/o=ORG/EDB not loaded
FATAL ERROR: DSA Halted
```

You must now look at the file *own_dsa/c=$(COUNTRY)/o=ORG/EDB* to search the attribute “treeStructure”. Then you compare its values with the attribute “objectClass” of that entry which gave the error message.

C. Fatal error — Unable to start network monitor

If the local EDBs are consistent, but another process is already lis-

3.12 Starting the DSA

You start the DSA in the directory *$(ETCDIR)/quipu/own_dsa*:

```
# $(SBINDIR)/ros.quipu -t ./quiputailor
```

The parameter '-t' tells the DSA to read the configuration from the file *quiputailor* in the current directory. Look at the manual page (*quipu(8c)*) if you need further details of parameters to *ros.quipu*. You should pay careful attention to the contents of the *quiputailor* file, since setting up the DSA depends essentially on this file. If you have done a successful configuration, the DSA will print out something like

```
• • •
DSA c=$(COUNTRY)@cn=own_dsa has started on
Internet=aaa.bbb.ccc.ddd+17003
• • •
```

where you find the IP address and the TCP port number the DSA is listening on. When you have seen the DSA to start correctly, abort it with control-c. Then restart it to the background:

```
# $(SBINDIR)/ros.quipu -t ./quiputailor > /dev/null 2>&1
```

3.12.1 DSA Failure

If the DSA fails to start, check first that ISODE and the DUA (*dish*) are working properly (see Sections 3.7 and 3.8). There are generally three causes of failure:

- there is an attribute related schema problem
- there is a structure related schema problem in the DIT
- something else is running at the address of the DSA

Since the files have been generated automatically, the first two errors are unlikely. However, there might be spelling errors or catastrophic editing in your files. Consult the logfile *dsap.log* in the directory *$(ETCDIR)/quipu/own_dsa* to get more information about the failure. You should also change the "dsaplog" level to the value "all" in the *$(ETCDIR)/quipu/own_dsa/quiputailor* file (e.g. "**dsaplog level=all**") and then try to start the DSA

```
# logging directory
logdir              /var/log/

# ‘‘higher-performance’’ nameservice
nameserver        on

# default tree directory
treedir               /export/common/quipu-db/

# slave updates
update              on

# level below which users can search
searchlevel       2

# limits...
adminsize         100
admintime         60
cachetime         21600
conntime          120
nsaptime          30

# allow addition of "last modified by" attributes...
lastmodified          on

# chain requests ?
dspchaining         on

# read-only mode ?
readonly        off

# The length of time before between attemping to update SLAVEs
slavetime         21600
```

Figure 5. cont.

```
###############################################################################
#
# quiputailor - ISODE QUIPU Directory Service Specific Tailoring File
#
# $Header: /usr/files/osi/quipu/RCS/quiputailor,v 6.1 89/06/08 13:56:19 mrose Exp $
#
#
# $Log:       quiputailor,v $
# Revision 6.1  89/06/08  13:56:19  mrose
# touch-up
#
# Revision 6.0  89/06/07  21:38:45  mrose
# 5.1
#
# Revision 6.0  89/03/18  23:41:49  mrose
# Release 5.0
#
###############################################################################


# this line must occur first
oidtable        /usr/local/etc/oidtable

mydsaname       "c=FI@cn=Rhea"

# DSAs to contact if we need a referral to a parent

parent       "cn=jaguar"              '0101'H/Internet=128.214.1.1+17003

# ROOT MASTER at University College London
parent      "cn=Giant Tortoise"    '0101'H/Internet=128.16.5.31+2005|Int-X25(80)=23421920030045

# US SLAVE at NYSERNet Inc.
parent       "cn=Fruit Bat"           '0101'H/Internet=130.117.128.3+17007

# IS MASTER at
dsa_address "Elephant Seal"      '0101'H/Internet=130.208.165.63+17003

# NO MASTER hold by UNINETT
dsa_address "Electric Eel"          '0101'H/Internet=129.240.2.40+2005

# SE MASTER at UME
dsa_address "Hummingbird"        '0101'H/Internet=130.239.1.15+2005|Int-X25(80)=240200100306

# AU MASTER AT CSIRO
parent       "cn=Anaconda"           '0101'H/Int-X25(80)=5052334300013+PID+03018100

# full screen logging
dsaplog           level=debug sflag=tty file=dsap.log size=100 sflag=zero
stats             level=all dflag=tty sflag=zero file=stats.log size=50
```

Figure 5. An example of the *quiputailor* file.

attempting to connect to a DSA that could not be contacted earlier.

conntime: Time to hold an unused connection open.

readonly:User modifications to the DIT are refused. SLAVE updates are still allowed.

admintime: The maximum time which a DU can spend in a query.

On the next two pages there is an example of the *quiputailor* file (Figure 5).

treedir: The directory where the database is stored. This path points to the root of the DIT. (*$(ETCDIR)/quipu/own_dsa*)

nameserver: The value "off" prevents Nameserver access to the directory. By default this is "on". (off)

update: The value "on" tells the DSA to update SLAVE EDBs when it starts up. When you start the DSA for the first time, this should be "off". (off)

dspchaining: The value "on" tells the DSA to chain requests to other DSAs if necessary. (on)

parent:This entry consists of a name/address pair. The DSA referenced needs to hold a MASTER or SLAVE copy of an EDB higher up in the DIT than the highest locally held EDB. If more than one parent entry is found, the DSA will choose which DSA to contact. Currently, the first entry (`$(COUNTRY_DSA)`) is always used; if this fails the second is used etc.

stats: The value has the same format as the dsaplog entry in *$(ETCDIR)/dsaptailor* (See "Editing the *dsaptailor* file").

The following options are not so essential when starting the DSA for the first time, but they might be significant later. More options are described in [SKill89b].

lastmodified: If "off", the DSA will not add the attribute "lastmodifiedby" when an entry is altered.

searchlevel: Defines the level from which users will be able to search the DIT subtree — default is 2 (i.e. organizations). If the DU tries to search from higher up, an "unwilling to perform" error will result.

retrytime: The length of time in seconds before deciding it is worth

of the entry does not exceed the limit (usually 15 characters). If the RND is longer than the limit, *EDB.map* file is created. You can use map files on every level in the DIT.

After you run *dsaconfig*, the initial database looks like:

EDB
c=$(COUNTRY)/EDB
c=$(COUNTRY)/o=ORG/EDB
c=$(COUNTRY)/o=ORG/ou=ORG_UNIT/EDB

Note the possible spaces in the names of the subdirectories

3.11 Testing your DSA

The first thing to do is to start the DSA interactively so that you can see that it can successfully load its local EDB files. Before you start the DSA, make sure the options in the *quiputailor* file are correct.

This file contains information for the DSA and it was created automatically by *dsaconfig*. You give the *quiputailor* file as a parameter to *ros.quipu*. The behavior of the DSA depends a lot on the configuration of this file!

3.11.1 Options in the *quiputailor* file

When you start the DSA, the default *quiputailor* file is *$(ETCDIR)/quiputailor*. You can override this with parameter "-t" to *ros.quipu*. The *quiputailor* file indicates such things as:

- name of the DSA (the name is indicated with the option "mydsaname" and its value is the DN of the DSA)
- location of the database
- location and level of the logs the DSA will produce
- location of any other DSAs for initial bootstrap

Next, we look at the options. For the most important options I have added examples in parenthesis.

mydsaname: The DN of the DSA. (cn=own_dsa)

owner and group of the files appropriately. The database files are structured according to the UNIX directory hierarchy.

3.10 Database structure

At every level in the DIT for which your DSA holds data, there is an *EDB* file — also at the root level. If an entry defined in an *EDB* has information in the DIT at the level lower than this one, this information is found in the subdirectory whose name is the Relative Distinquished Name (RDN)[5] of the entry. For example, if we have an entry "o=Tampere University of Technology", the children of this entry will be found in a file *o=Tampere University of Technology/EDB*. In some UNIX variants (e.g. System V) the names of the subdirectories are restricted in length. In that case you can abbreviate the long names as follows.

Suppose we have an entry "o=Tampere University of Technology" in the *c=FI/EDB* file and it seems to us that the name is too complicated for the name of the subdirectory. We can then create a new file *c=FI/EDB.map* with the line:

o=Tampere University of Technology # TTKK

This line tells the DSA to search the children of "o=Tampere University of Tecnology" from the *c=FI/TTKK/EDB* file. If the *EDB.map* file does not exist, the DSA uses the RDN of an entry as the subdirectory name. The common format for *EDB.map* is thus:

<RDN> '#' <abbreviated name>

When the DSA needs to create a subdirectory (e.g. adding a branch to the DIT), it creates the subdirectory based upon the RDN of the entry if the length

5. The RDN defines an entry unambiguous at one level whereas the DN (Distinquished Name) defines the whole path of an entry. For example, if the DN of the organization Tampere University of Technology is "c=FI@o=Tampere University of Technology", then its RDN is "o=Tampere University of Technology".

can choose the name for your DSA freely when you do not use names the already existing ones. It is recommended that you choose the name for your DSA from the Appendix A. There is a table of South-American wildlife and plants (this is an international habit for the DSA names!). Since some of the names are long, it is acceptable to use only part of the wildlife names for the common name of your DSA.

3.9.2 Editing the configuration file of the DSA

The configuration of the DSA is done with *dsaconfig(8)*. It takes the name of the DSA as an argument. When you have chosen the name, go to the directory *quipu* in **$ETCDIR**.

```
# cd $(ETCDIR)/quipu
```

The file *wildlife.dsa* contains a template for the configuration. Copy it and change its permissions

```
# cp wildlife.dsa own_dsa.dsa
# chmod 0600 own_dsa.dsa
```

where “own_dsa” is the name of your DSA.

Now edit this new file to define the initial configuration for your DSA. There are some examples in the file helping you to fill in the needed information. Note that all the information you fill in concerns your organization and the organizational unit in which you are working. Afterwards you can respectively change the filled information by editing the files *c=$(COUNTRY)/o=ORG/EDB* and *c=$(COUNTRY)/o=ORG/ou=ORG_UNIT/EDB*.

3.9.3 Running the DSA configuration program

Dsaconfig updates the files *$(ETCDIR)/fredrc* and *$(ETCDIR)/dsaptailor* based upon the knowledge in *own_dsa.dsa*. When you have finished filling in the information for your organization, run *dsaconfig*:

```
# $(SBINDIR)/dsaconfig own_dsa
```

The program creates database directories, all needed files and sets the

the TCP port.

4. If you can connect, close the connection and make a report to your country manager. Something strange is now going on: *telnet* can reach the address but the DSA can't!

5. If *telnet* reports

```
Connection refused
```

this means that the DSA is not listening at that TCP port. Either the DSA is temporarily unavailable or it has been moved to an another address. Wait some time and try again. If you still are unable to connect, send a message to the country manager. Go then on at step 2.

If you have gone through all the alternatives in *dsaptailor*, contact the country manager for further instructions.

B. Service error — missing network address separator

You have possibly mistyped the name of the DSA. Try again. If you see other error messages, contact the country manager.

3.9 Configuring a DSA

Now when we have got a connection to another DSA we can concentrate to set up your own DSA. We use symbol “ORG” to describe the name of your organization and “ORG_UNIT” to describe the name of an organizational unit.

3.9.1 Choosing a name for the DSA

In order to find out what names are already reserved, give the following command:

```
% dish -c $(COUNTRY_DSA)
Welcome to Dish (DIrectory SHell)
Dish -> search @c=$(COUNTRY) -filter objectClass=dsa
•••
```

Here we searched at the country level DSA names that already exists. You

various databases used by ISODE. Normally you do not have to deal with these files unless they are corrupted for some reason.

isotailor: ISODE is tailored to the system with this file. You will configure this file before starting a DSA.

3.8 Testing the DUA

When ISODE is running properly, try to connect to your country level DSA:

```
% dish -c $(COUNTRY_DSA)
Welcome to Dish (DIrectory SHell)
Dish ->
```

This indicates that you are able to make the connection. You can get help by typing a question mark. You might now try some simple commands. If you are unable to connect the country DSA, try other DSAs listed in *$(ETCDIR)/dsaptailor*.

3.8.1 DUA Failures

The DUA might fail from some reason. Make sure first that ISODE is operating correctly (see Section "Starting and testing basic ISODE"). If so, here are the most common errors encountered:

A. Service error — Unable to contact DSA

It is likely that there is some problem with the network, or you have mistyped the name of the DSA. Look at the file *$(ETCDIR)/dsaptailor* and the line there which begins:

```
dsa_address "$(COUNTRY_DSA)"
```

You find an Internet address and a TCP port number on this line.

1. Use *ping(8c)* to see if you can reach the IP address

2. If not, try other DSAs in *dsaptailor*. Repeat this process until *dish* successfully contacts a DSA.

3. If *ping* can not reach the IP address, use *telnet* to try to connect to

delete
dsacontrol
list
modify
modifyrdn
moveto
search
showentry
showname
squid
unbind

Don't forget to run *unbind* after you are done with your actions!

fred: Users interface to the Directory

editentry: Edit a Directory entry. This shell script is invoked by *dish* when you ask *dish* to edit an entry.

In the directory **SBINDIR** — usually */usr/etc/* or */usr/local/etc/* — resides the DSA itself:

ros.quipu: QUIPU DSA. This program is started only once for each DSA. The program will be started in */etc/rc.local*.

dsaconfig: A configuration program for Level-1 DSA.

In the directory **ETCDIR** — usually */usr/etc/* or */usr/local/etc* — there are a couple of programs and files of interest:

oidtable.at, oidtable.gen, oidtable.oc: These files define the attribute types and the way to identify objects and object classes. Normally you do not have to deal with these files unless they are corrupted for some reason. If there are national oidtables, they are found in the **FTPDIR** directory (see Appendix B) in each Nordic country. Contact the **FTP_HOST** to see if you can find oidtables and pick them up to the **ETCDIR** directory.

dsaptailor: The DUA is tailored to the system with this file. You will configure this file before starting the DSA.

isoaliases, isobjects, isoentities, isomacros, isoservices: These are

```
# $(SBINDIR)/tsapd -t 1>/dev/null 2>&1 &
```

or when you use *csh*:

```
% $(SBINDIR)/tsapd -t >& /dev/null
```

The easiest way to test ISODE is to run the script *isode-test* from the directory *$(ETCDIR)/quipu/scripts*.

```
% isode-test
```

On the screen you can follow how the test goes further. When the script terminates, it will indicate the number of errors encountered. If an error occurs, try to determine what the problem is. Examine also the logfiles (*$(LOGDIR)/*.log*). If you are unable to find the reason for the error message, send a message to your country level manager. The message should contain the output of *isode-test*, configuration files you have changed and the description of the hardware you are using.

3.7 Interesting files

The number of files needed is quite small.

In the directory **BINDIR** — usually */usr/local/bin*, there are a few programs of interest:

dish: DIrectory SHell. This is an interface to the OSI Directory with which you can access the services of the Directory. The program is not intended directly to ordinary users but the administrators employ this to perform routine maintenance and diagnostics.

bind: Shell interface to *dish*. Dish-interface can be exported to the UNIX shell with this program. You can issue commands to *dish* from shell, rather than running *dish* directly. First you “bind” the Directory with *bind* and then you can use the following commands (actually they are same commands when using *dish* directly):

add
compare

```
# ./make zap
```

in the *isode-6.0* directory as the super-user.Before the DUA and the DSA can be started, you have to configure the *$ETCDIR/isotailor* file. The installation we have done so far has not installed the *isotailor* file. We do it now by hand:

```
# cp $(ETCDIR)/quipu/scripts/isotailor.empty $(ETCDIR)/iso-
tailor
```

See the manual page (*isotailor(5)*) for a detailed explanation of the *isotailor* file. At time we need only the options “localname” and “logpath”. Add the following lines to the *isotailor* file:

logpath: **LOGDIR**
localname: $(NAME)

where you replace $(NAME) with your hostname. For example:

logpath: /var/log/
localname: tut.fi

3.6 Starting and testing basic ISODE

From now on we use the names **$(SBINDIR)**, **$(BINDIR)**, **$(LOGDIR)** and **$(ETCDIR)** as the names of the directories you have chosen in the configuration. Replace them with the absolute path names.

In order to make sure that ISODE system is always running, there are some things you need to do:

1. Verify that the file */etc/rc.local* has lines:

```
if [ -f $(SBINDIR)/tsapd ]; then
    $(SBINDIR)/tsapd -t & (echo -n ' tsapd')  > /dev/console
fi
```

2. Verify that the file */etc/services* has an entry :

```
tsap        102/tcp
```

3. Verify that the file */usr/lib/crontab* has an entry:

```
0 4 * * * su daemon < $(SBINDIR)/isologs
```

or some other entry which cleans up the logfiles

4. Start the *tsapd(8c)* daemon by hand. When you use *sh* start it with

something has gone wrong. Try to find out what the problem is. If you can not find the problem, send a message to the country level manager. The message should contain the output of the *make* command and the description of the hardware you are using. If you catch up **messages like**

```
warning: statement not reached
ranlib: warning: ../libisode.a(aetdbm.o): no symbol table
```

or

```
*** Error code 1 (ignored)
```

these **are normal**. The error messages might be caused by incomplete configuration files or the system you are using is non-standard (i.e. the compiler might be different).

Now generate QUIPU:

```
% (cd others/quipu; ./make pilot)
```

3.5 Installing and tailoring ISODE

Once you are satisfied with the generation, install both ISODE and QUIPU. To be able to run installation, you must first become the super-user. Make sure the directories **BINDIR**, **SBINDIR**, **ETCDIR** and **LOGDIR** (defined in *config/CONFIG.make*) exist. If not, create them using *mkdir*. The protection of **LOGDIR** should be 0777.

In order to install programs, libraries and databases:

```
# ./make inst-all inst-quipu
# (cd others/quipu; ./make inst-pilot)
```

Note the usage of the './' again! Make sure that the files in **ETCDIR** have read permission for the world. Later starting the DUA as an ordinary user may fail because of incomplete permissions of these files. Other applications — such as FTAM — are not installed at this moment. You can do it on your own later.

If you are not going to do any program development, you can remove the various libraries and program development tools by typing

```
sunos4      SunOS release 4
sys52-exos  SVR2 UNIX with EXOS
sys52-rt    RT/PC with AIX
sys52-sun   SVR2 UNIX emulation on SunOS release 3
sys52-win   SVR2 UNIX with WIN/TCP
sys53       generic SVR3
ultrix      Ultrix 3.1
```

The configuration is very simple. For example, the configuration for SunOS 4.0 with TCP/IP only is done as follows:

```
% cp config/sunos4.h h/config.h
% cp config/sunos4.make config/CONFIG.make
% cp config/*.local support/
```

Look for your configuration files in the list above and execute the commands for your system as in the previous example.Then look at the files

h/config.h
config/CONFIG.make

to see if the directories **BINDIR**, **SBINDIR**, **ETCDIR** and **LOGDIR** are suitable for your purposes. Normally **SBINDIR** and **ETCDIR** should point to the same directory. The directories and their meaning are:

BINDIR	/usr/local/bin/	where to install user programs
SBINDIR	/usr/etc/	where to install administrator programs
ETCDIR	/usr/etc/	where to install administrator files
LOGDIR	/usr/tmp/	where to install log files
INCDIR	/usr/include/isode/	where to install include files
LIBDIR	/usr/lib/	where to install object libraries
LINTDIR	/usr/lib/lint/	where to install lint libraries
SYSTEM		directs how to create loader libraries
MANOPTS		see util/inst-man.sh for details

Once you have configured the files properly, generate both ISODE and QUIPU using the command:

```
% ./make everything >& make.out &
```

The generation takes a few hours, so we started it as a background job. Notice the use of ’./’ in the command! If you see a line (and **only** line like these) with

```
*** Error code 1
```
or
```
Exit
```

3.3 Getting sources

You can get a copy of the source release by using anonymous *ftp*:

```
% cd /usr/src/local
% ftp $(FTP_HOST)
Connected to $(FTP_HOST).
220 $(FTP_HOST) FTP server (SunOS 4.0) ready.
Name ($(FTP_HOST):user): anonymous
331 Guest login ok, send ident as password.
Password:guest
230 Guest login ok, access restrictions apply
ftp> cd $(FTPDIR)
200 PORT command successful.
ftp> binary
200 Type set to I.
ftp> get isode-6.0.tar.Z
200 PORT command successful.
ftp> quit
221 Goodbye.
% zcat isode-6.0.tar.Z | tar xbf -
% rm isode-6.0.tar.Z
```

For the usage of the variables **$(FTP_HOST)** and **$(FTPDIR)** refer appendix B.[4] Next, change to the directory *isode-6.0* release of the software:

```
% cd isode-6.0/
```

The file *READ-ME* in this directory contains instructions on how ISODE will be configured, generated and installed.

3.4 Configuring and generating ISODE

The configuration is very simple. For each supported system there is a pair of skeleton files (".make" and ".h") in the *config* subdirectory. Currently the following systems are supported:

FILE	CONFIGURATION
apollo	Apollo
aux	A/UX release 1.1
bsd42	generic 4.2BSD UNIX
bsd43	generic 4.3BSD UNIX
bsd43-rt	RT/PC with 4.3BSD
bsd44	4.4BSD UNIX with OSI
hpux	HP-UX
mips	MIPS RISC/OS
osx	Olivetti LSX 30xx
ros	Ridge Operating System
sunlink3	SunOS release 3 with SunLink OSI release 5.2
sunlink4	SunOS release 4 with SunLink OSI release 6.0
sunos3	SunOS release 3

4. From now on consult always the appendix B when there are variables like these in the text

3. Installation and configuration

In this chapter we describe how you can install, configure and test the Directory services. Finally you have successfully joined the Directory!

3.1 The Software

The Software we are using is QUIPU implementation of the OSI Directory. QUIPU is a part of the ISO Development Environment (shortly ISODE). The purpose of QUIPU is to build up an experimental environment for standardized Directory services because we do not have yet any wide experience about the OSI Directory. The present implementation of the software contains a DSA and interfaces to the Directory: DIrectory SHell — *dish*, FRont End to Dish — *fred*, Window Interface Developed at GEc hirsT — *widget*, SUNVIEW interface — *sunint* (see Appendix C). A complete explanation of QUIPU is found in [SKill89a]. You can get more information about the software by sending mail to your country level manager[3].

3.2 Selecting a computer

The machine you are going to use for the Directory service should support the Berkeley variant of UNIX®, (although the software is known to run on other variants of UNIX — such as AT&T and HP — too.)

Your computer running the DSA should have available at least 1/2 Kilobytes of RAM per stored entry. Make sure that this is the case since a paging DSA performs extremely poorly.

---

3. c=**$(COUNTRY)** @cn=Directory Manager where **$(COUNTRY)** is NO, SE, FI or IS respectively. See appendix B.

Level-1 DSAs. At present, this guide doesn't support the use of Level-2 DSAs. Nevertheless, it is important to structure an organization so that it contains organizational units too. You can set up a Level-2 DSA when your organization is so large that running only one DSA loads your computer system too much. A part of the Level-1 DSA's functions can be passed to the Level-2 DSA and so reduce the memory requirements on the system running the Level-1 DSA.

from Sweden (c=SE), Norway (c=NO) and Iceland (c=IS) and propagates a copy of its country level EDB (c=FI) to these other Level-0 DSAs (*Hummingbird*, *Elephant Seal* and *Electric Eel*). Because of *Jaguar* is a MASTER DSA for Finland, it propagates also a copy of its own highest MASTER EDB (e.g. c=FI EDB) to the root level DSA (*cn=Giant Tortoise*). Correspondingly, all Finnish Level-1 DSAs obtain a copy of c=FI EDB and root EDB from *Jaguar*. Figure 4 tries to clarify the updating scheme.

2.4 Level-1 DSA

Each participating organization should have their own DSA with MASTER EDB for that organization. These DSAs reside directly under Level-0 DSAs in each country. Level-1 DSAs receive a copy of the root and country level EDB (c=NO, c=SE, c=FI or c=IS) from a Level-0 DSA (*Electric Eel*, *Hummingbird*, *Jaguar* or *Elephant Seal*) and propagate their own organization level EDBs to the Level-0 DSA. Correspondingly, Level-1 DSA propagates a copy of its own organization level EDB and a copy of both the root and country level EDB to all Level-2 DSAs below this DSA. We take an example:

"c=FI@o=Tampere University of Technology" is an organization having its own DSA (*cn=Rhea*). *Rhea* has the MASTER EDB for the organization Tampere University of Technology. In addition, *Rhea* receives a copy of both root and c=FI EDBs from the country level DSA *Jaguar* and propagates a copy of its own highest MASTER EDB (o=Tampere University of Technology EDB) to *Jaguar*. Because of there are no more Level-2 DSAs in this organization, there is no information to transfer downwards in the DIT either.

2.5 Level-2 DSA

Each participating organization can run zero or more Level-2 DSAs as the size of their organizational units requires. These DSAs reside directly under

Suppose we want to set up a DSA for an organization “o=ISO Company” which would lie below “c=SE” in the DIT. Then we must transmit to the new DSA SLAVE copies of both the root and country level EDBs. Now the new DSA knows its place in the DIT. It can be reached by other DSAs and it may contact others DSAs. When a DU wants to move from one node to another, it is done via the root of the DIT. That is why every DSA have to know how to reach the root. For efficiency reasons every DSA should have SLAVE copy (or copies) of all EDBs above itself.

2.2 Organizational Model

DSAs have been classified into three categories: there are zero level (Level-0), first level (Level-1) and second level (Level-2) DSAs. This classification is made for the purposes of the NYSERNet Pilot Project, but it clarifies us too how the DIT has been mapped between DSAs.

2.3 Level-0 DSA

These DSAs are highly-available authoritative servers, which can give referrals to requests from Level-1 DSAs. At present Nordic countries Norway, Sweden, Finland and Iceland have each one DSA of this type: in Norway a DSA called *cn=Electric Ee*l, in Sweden *cn=Hummingbird*, in Finland *cn=Jaguar* and in Iceland *cn=Elephant Seal*. We could now examine how EDBs are updated between Level-0 DSAs.[2] As an example we take the Level-0 DSA in Finland *(cn=Jaguar)*.

Jaguar has a SLAVE copy of the root EDB which is the highest level EDB of the DIT. *Jaguar* obtains this EDB from a DSA *cn=Giant Tortoise*. In addition, *Jaguar* is a MASTER DSA for c=FI which implies that *Jaguar* contains information about the organizations in Finland. *Jaguar* receives EDBs

2. This might change later, but the present situation suits as an example here.

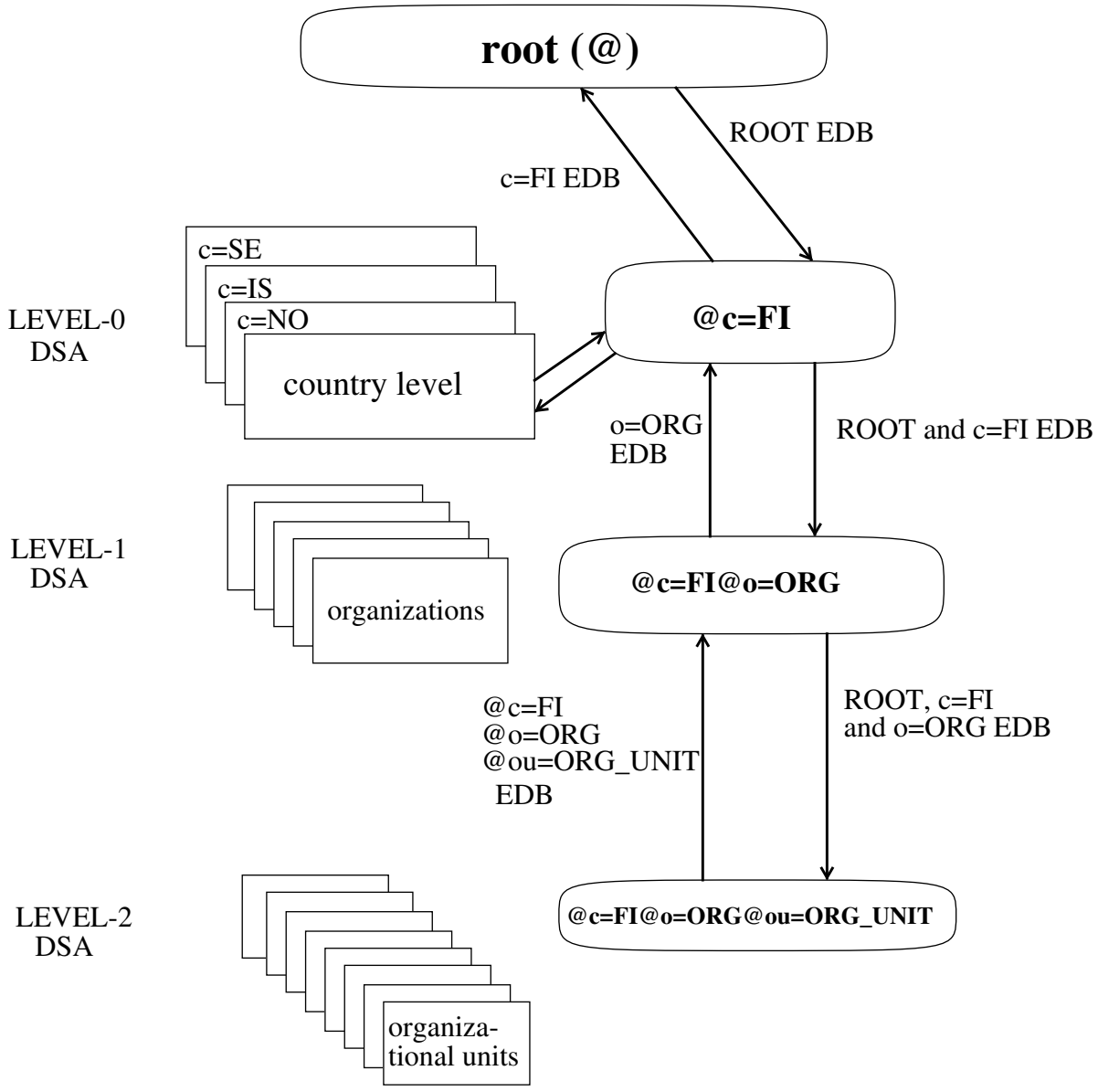

Figure 4. DSA levels.

Each level in the DIT is described with attributes (see Figure 3). Every attribute consists from the name and the value(s) of the attribute:

```
<attribute name> = <attribute value(s)>
```

For example, Sweden is described with the attribute “c=SE” (the name of the attribute is here 'countryName' — shortly 'c'). An organization is defined with the 'o' attribute — for example “o=Universitet i Oslo”. Alternatively, organizational units are defined with the 'ou' attribute and persons with the 'cn' (commonName) attribute.

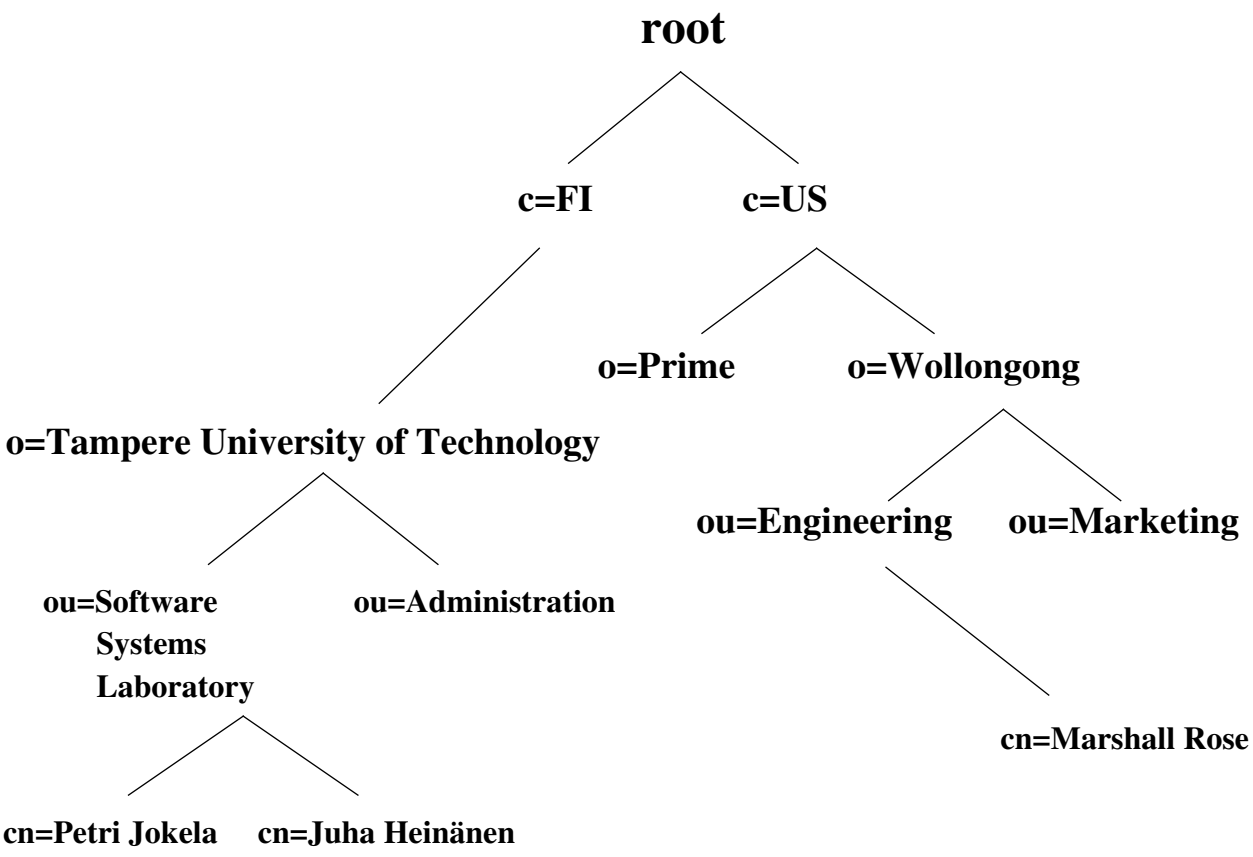

Figure 3. The DIT

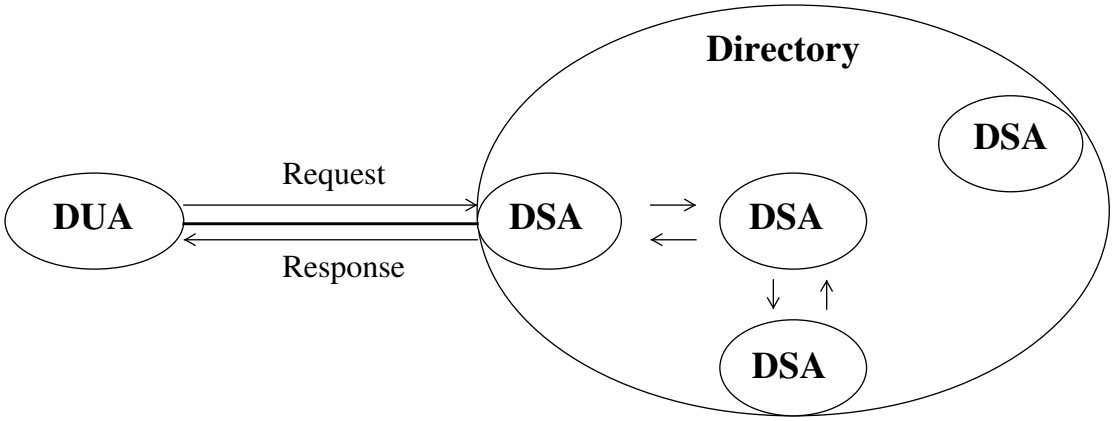

Figure 1. Operation chaining.

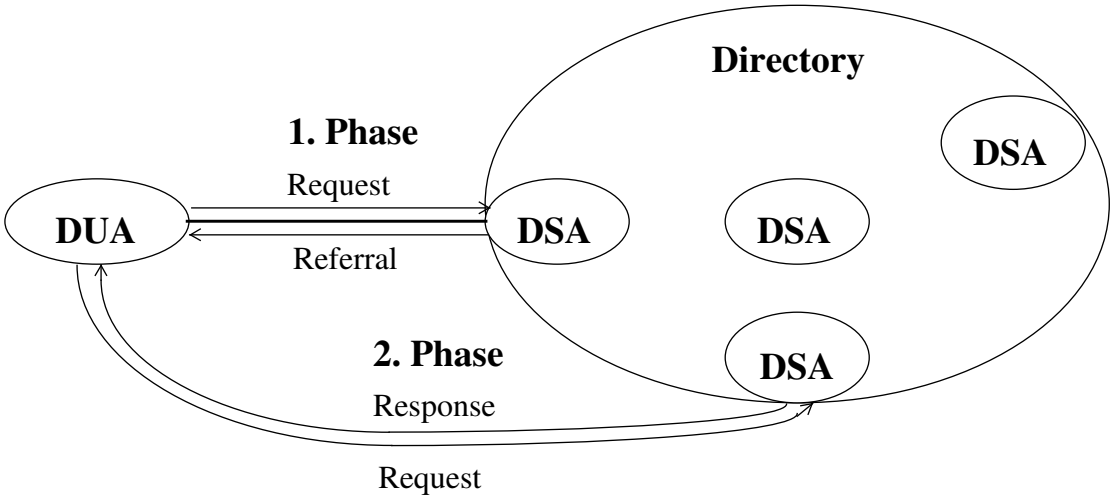

Figure 2. Operation referral.

2. Introduction to the Directory services

A Directory User (DU) contacts the Directory through a Directory User Agent (DUA). In our software the DUA is a program called *dish* (DIrectory SHell). The DU "converses" with the DUA to make searches, updates, etc. The DUA on its part directs the DU's requests to a Directory System Agent (DSA), which actually provides the services. If the DSA does not have the requested information resident, it has two choices. (1) it may either contact another DSA which is "closer" to the information and propagate the request (this is called chaining, see Figure 1) or alternatively (2) the DSA may return information about this "closer" DSA to the DUA, and let the DUA reissue its request (this is called referral, see Figure 2). Next we can ask: how the DSAs transmit information to each other?

2.1 Functional Model

A DSA can have MASTER or SLAVE copies of its Entry Data Blocks (EDBs) which are nodes in the Directory Information Tree (DIT). The DIT has organized hierachically (see Figure 3). At each level in the DIT —except the leaves — there is an EDB. The EDB contains **all** the information of the next lower level in the DIT (see Figure 4). MASTER EDB copies are original and every time they need to be updated, it is done by the DSA which has the MASTER copy of that EDB. DSAs with the SLAVE EDB copies receive this new information from the DSA owning the MASTER EDB copy. SLAVE copies can be updated when the DSA starts or by the manager when he/she wants so or automatically at certain interval.

NYSERNet White Pages Pilot Project's manual [NYSERNet].

1.1 Scope of this document

The information in this document is valid in Nordic countries Sweden, Norway, Iceland and Finland and the instructions are tailored for Nordic organizations to ease their work in setting up the Directory Services.

In chapter two we will try to clarify the Directory concepts. We get to know how the Directory System Agents (DSAs) discuss with each other and how they have been categorized according to the information they hold.

In chapter three we will describe how you can set up your own DSA. The installation and configuration will be done step by step in order to ease your work. If you are already familiar with the structure and concepts of the Directory, you can move directly to the chapter three.

After you have successfully set up the Directory Services, you can start to learn its usage. For this purpose Appendix C contains a short description of the user interface tools of the ISODE package11

.

1. Overview to the Directory Services

The development of the Directory started from the telecommunications users' need to find out how a person can be reached via telephone, telefax, electronic mail, etc. Today the Directory services include many other applications. The Directory can hold information not only about individuals but also about OSI applications such as FTAM (File Transfer, Access and Management), MHS-routing (Message Handling System) and yellow pages services. The White Pages service of ISODE [Ros89] package — called QUIPU[1] — is a freely distributed implementation of the OSI Directory. ISODE contains the Directory as well as OSI implementation of the layers 5 and 6 plus other OSI applications. ISODE forms a good basis to move on to OSI.

The Directory is a collection of open systems that cooperatively hold information about real world objects. An object can be anything named in some world, but usually the world is restricted to that of telecommunications and information processing areas. For example, typical directory objects include countries, organizations, persons, computer systems or applications like FTAM. A detailed introduction to the Directory can be found in [JH].

The standard of the Directory is defined in [ISO88]. The design of the QUIPU software is described in the report "The Design of QUIPU" [SKill89a] and the implementation and usage of QUIPU in the fifth volume of the report "ISODE Users Manual" [SKill89b]. This document is mainly based on

1. QUIPU is a name of a project which got its name from the world of Incas. They stored information on strings and knotted them in a specific manner with coloured threads. Then they attached strings to a larger rope which were known as *Quipus*.

3.15 Controlling the DSA 33
Chapter 4: Maintenance 35
4.1 Editing the *$(ETCDIR)/dsaptailor* file 35
4.2 The logfiles 39
4.3 Adding an entry to the database 39
4.3.1 Adding entries with *dish* 40
4.4 Maintaining the database 40
4.5 Common security features 41
4.5.1 The software 41
4.5.2 File permissions 42
4.5.3 The Access Control List attribute 42
4.5.4 Publicly readable attributes 43
4.6 Retrospect 43

The References 44
Appendix A: DSA Names 45
Appendix B: Information about DSAs in the NORDUNET countries 46
Appendix C: User Interfaces 47

Table of Contents

Chapter 1: Overview to the Directory Services 1
1.1 Scope of this document 2
Chapter 2: Introduction to the Directory services..... 3
2.1 Functional Model..... 3
2.2 Organizational Model..... 7
2.3 Level-0 DSA..... 7
2.4 Level-1 DSA..... 8
2.5 Level-2 DSA..... 8
Chapter 3: Installation and configuration..... 10
3.1 The Software 10
3.2 Selecting a computer 10
3.3 Getting sources 11
3.4 Configuring and generating ISODE 11
3.5 Installing and tailoring ISODE..... 13
3.6 Starting and testing basic ISODE..... 14
3.7 Interesting files 15
3.8 Testing the DUA..... 17
3.8.1 DUA Failures..... 17
3.9 Configuring a DSA..... 18
3.9.1 Choosing a name for the DSA..... 18
3.9.2 Editing the configuration file of the DSA..... 19
3.9.3 Running the DSA configuration program..... 19
3.10 Database structure 20
3.11 Testing your DSA..... 21
3.11.1 Options in the *quiputailor* file 21
3.12 Starting the DSA..... 26
3.12.1 DSA Failure..... 26
3.13 Contacting your DSA 28
3.14 Joining the global Directory 29
3.14.1 Updating SLAVE copies 32

The Directory Guide: Installation and configuration

Petri Jokela
pjj@funet.fi

Tampere University of Technology
P.O.B 527
33101 Tampere
Finland

You are holding now a document, which helps you to join to the OSI Directory! In this document we concentrate on the installation and configuration of the QUIPU directory software which is part of the ISODE package. This is not a tutorial on the OSI Directory (shortly the Directory) and its contents. However, in order to join and maintain the Directory, we must go through a few basic concepts concerning it. The goal of this document is to give you enough information, so that you can join the Directory and maintain it. This guide is made for NORDUNET purposes to ease Nordic organizations to join the Directory but it can be useful for other vendors, too.

March 7, 1990