NAME

isode–gen – generating the ISO Development Environment

READ THIS

This documentation describes how to configure, generate, and install the ISO Development Environment.

Acquisition, use, and distribution of this module and related materials are subject to the restrictions of a license agreement. Consult the Preface in the *User's Manual* for the full terms of this agreement.

You will probably want to read over this entire document first, before typing any commands; e.g., there are optional components described later on that require additional settings in the configuration file.

Comments concerning this release should be directed to the mailbox ‘‘Bug–ISODE@XTEL.CO.UK’’. Do **not** send bug reports to the ISODE discussion group. If you want to subscribe to the ISODE discussion group, drop a note to ‘‘ISODE-Request@NIC.DDN.MIL’’.

SYNOPSIS

% cd isode–7.0
% cp config/*system*.h h/config.h
% cp config/*system*.make config/CONFIG.make
% cp config/∗.local support/
% ./make everything
# ./make inst–everything

DESCRIPTION

This is a description of how one can bring up the ISODE. It is assumed that you have super–user privileges in order to (re–)install the software. Super–user privileges are not required to configure or generate this software.

The distribution tape contains the hierarchy for the **isode–7.0** directory. Bring the sources on–line by changing to a directory for local sources and running tar, e.g.,

% cd /usr/src/local/
% tar x
% cd isode–7.0

CONFIGURATION

First, go to the **config/** directory.

% cd config

Select the Makefile and include-file skeletons which most closely match your system. The current choices are:

file	*configuration*
apollo	Apollo
aux	A/UX release 2.0.1
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/X.25 release 5.2
sunlink4	SunOS release 4 with SunLink OSI/X.25 release 6.0
sunnet7	SunOS release 4 with SunNet OSI release 7.0
sunnet7x	SunOS release 4 with SunNet X.25 release 7.0
sunos3	SunOS release 3
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
sys54	generic SVR4
ultrix	Ultrix 3.1

The makefile skeleton has the extension **.make**, whereas the include–file skeleton has the extension **.h**.

MAKEFILE

Copy the makefile skeleton of your choice to **pickle.make**, where ‘‘pickle’’ is the name of your system. Now edit this file to set the following *make* variables:

variable	*default*	*specifies*
OPTIONS		options to *cc* and *lint* (e.g., -I../h)
LSOCKET		libraries to link in (e.g., -lcci)
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
MANDIR	/usr/man/	where to install man pages
MANOPTS		see compat/inst-man.sh for details

NOTE THAT ALL THESE DIRECTORIES MUST BE ABSOLUTE PATH NAMES (i.e., start and end with a ‘/’).

Finally,

```
ln pickle.make CONFIG.make
```

(yes, that's ‘‘CONFIG’’ in uppercase and ‘‘make’’ in lowercase). Both of these files are in the **isode–7.0/config/** directory. This latter file is the one which the software uses to configure itself during generation.

INCLUDE–FILE

Copy the include–file skeleton of your choice to **pickle.h**, where ‘‘pickle’’ is the name of your system. Now add any additional definitions you like (usually none). Consult the file **config/OPTIONS** for a list.

Now:

```
ln pickle.h ../h/config.h
```

This latter file is the one which the software uses to configure itself during generation.

ALIASES DATABASE

Typically, sites run with the default aliases database used by the OSI directory. In this case, simply copy the default local configuration file to the **support/** directory:

```
% cp aliases.local ../support/
```

If you have local modifications you wish to make, either copy in your own file or edit the file **support/aliases.local** as appropriate.

SERVICES DATABASE

Typically, sites run with the default services database. In this case, simply copy the default local configuration file to the **support/** directory:

```
% cp services.local ../support/
```

If you have local modifications you wish to make, either copy in your own file or edit the file **support/services.local** as appropriate.

ENTITIES DATABASE

Typically, sites run with the default application entity database used by the stub–directory service. However, once things are running, sites should use the OSI Directory to keep track of application entities. So, to begin, simply copy the default local configuration file to the **support/** directory:

```
% cp entities.local ../support/
```

If you have local modifications you wish to make, either copy in your own file or edit the file **support/entities.local** as appropriate.

In particular, if you are using SunNet OSI, it will be necessary to put an entry in your **support/entities.local** file of the form:

```
myhost  default    1.17.4.1.0    #1/NS+mynsap
```

where ‘‘myhost’’ is the name of the local machine, and ‘‘mynsap’’ is the NSAP of the local machine. For SunNet OSI 7.0 the NSAP is most easily determined by running

```
% /usr/sunlink/osi/etc/osirstat -n | grep ^DA
```

provided that the SunNet OSI osi.routed program is running. For earlier SunLink OSI releases you can run

```
% cd others/osilookup
% ./make
% xosilookup localhost CLIENT
```

providing that the SunLink OSI file **/etc/sunlink/osi/hosts** has an entry defining the service for ‘‘localhost’’ called ‘‘CLIENT’’. (Note that in releases earlier than SunLink OSI 6.0, the file is called **/usr/etc/osi.hosts**) Note that this entry is mandatory if you are running SunLink OSI release 5.2 or greater.

One further note for users of a release earlier then 7.0 of SunLink OSI: if you intend to run the standard SunLink OSI listener (osi.netd), then you must change the TSEL used by *tsapd* when it listens. This is done in two steps: First, in **support/entities.local**, change your entry to read as:

```
myhost  default    1.17.4.1.0    #2/NS+mynsap
```

Second, in **support/services.local**, add a line that reads as:

```
tsap/session    #2    tsapd-bootstrap
```

which overrides the default TSEL in the **support/services.db** file.

MACROS DATABASE

Typically, sites run with the default macros database. In this case, simply copy the default local configuration file to the **support/** directory:

```
% cp macros.local ../support/
```

If you have local modifications you wish to make, either copy in your own file or edit the file **support/macros.local** as appropriate.

OBJECTS DATABASE

Typically, sites run with the default objects database. In this case, simply copy the default local configuration file to the **support/** directory:

```
% cp objects.local ../support/
```

If you have local modifications you wish to make, either copy in your own file or edit the file **support/objects.local** as appropriate.

GENERATION

Go to the **isode–7.0** directory

```
% cd ..
```

Now reset the dates of the configuration files for the system. This is done only once per source-tree:

```
% ./make once-only
```

then generate the basic system.

```
% ./make
```

If you are using SunOS, do not use the *make* program supplied with the SunPro package. It is not, contrary to any claims, compatible with the standard *make* facility. Further, note that if you are running a version of SunOS 4.0 prior to release 4.0.3, then you may need to use the *make* program found in **/usr/old/**, if the standard *make* your are using is the SunPro *make*. In this case, you will need to put the old, standard *make* in **/usr/bin/**, and you can keep the SunPro *make* in **/bin/**.

If you are using SVR3, then you will probably have to type this command before starting the compilation:

```
% ulimit 32768
```

Similarly, you may need to increase the stacksize limitation on other systems. For example, some users of the RT, report needing to use

```
% limit stacksize 16m
```

in order to get FTAM to fully compile.

The *make* command from the top-level directory will cause a complete generation of the system. If all goes well, proceed with the installation. If not, complain, as there ‘‘should be no problems’’ at this step. Some files while compiling may produce a

```
warning: statement not reached
```

or a

```
type ObjectDescriptor: Warning: Can't find file DSE.ph failed
```

message. This is normal. Sometimes when building a loader library, you might see several

```
ranlib: warning: ../libisode.a(aetdbm.o): no symbol table
```

messages. This is also normal. You might also see a few messages like:

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

This is also normal. As a rule, unless *make* says something like

```
*** Error code 1
```

or perhaps

```
Exit
```

then everything is going just fine!

TESTING

Some directories may have a resident test program, e.g., in the **psap/** directory, there is a program called *psaptest*. These programs are for internal testing only, and are not for use by ‘‘mere mortals’’. If you want to test things, after installation run *isode–test* (see the **USER PROGRAMS** section).

INSTALLATION

You will need to be the super–user to install the software. Note that installing the software from an NFS-mounted partition requires that you perform the installation as the super-user on the *target* system after changing to the source directory on the *source* system.

In the directions that follow, reference is made to some of the directories defined in the **CONFIG.make** file. You should substitute in the correct value, for example, if the expression

```
$(SBINDIR)tsapd
```

and if SBINDIR is defined as **/usr/etc/** in the **CONFIG.make** file, then you should type

```
/usr/etc/tsapd
```

instead.

There are two kinds of activities: once–only activities that you perform the first time the software is installed; and each–time activities that you perform every time the software is installed.

The first once–only activity is to verify that the *tsapd* daemon will be run when the machine goes multi–user. On Berkeley UNIX systems, add these lines to the **/etc/rc.local** file:

```
if [ –f $(SBINDIR)tsapd ]; then
    $(SBINDIR)tsapd >/dev/null 2>&1 &
    (echo –n ' tsap') > /dev/console
fi
```

On other systems, a similar procedure is followed. For example, on systems derived from AT&T UNIX, the file **/etc/rc2** script might be edited.

Once you are familiar with the system, you will likely want to run the OSI Directory and use another program, *iaed* to invoke local services. The section **DIRECTORY SERVICES** discusses this in greater detail. (However, if this is your first time, don't skip ahead.)

The next once–only activity is to verify that systems with a native **/etc/services** file contain an entry for the tsap service (if you have configured the ISODE to run over TCP). If not, add the line:

```
tsap        102/tcp
```

to the **/etc/services** file. Alternatively, some systems may have a definition of the form

```
iso-tsap    102/tcp
```

which is also acceptable. If your system does not have such a file, the software automatically compensates for this.

Next, on Berkeley UNIX systems, add a line to the **/usr/lib/crontab** file to invoke a shell-script that will re-cycle the log files. Usually, the line you add looks something like this:

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

which says that the shell-script $(SBINDIR)isologs should be invoked at 4am each morning. On other systems, a similar procedure is followed. For example, on systems derived from AT&T UNIX, the file **/usr/spool/cron/crontabs/root** might be edited followed by the command

```
% crontab root
```

There are two each–time activities:

```
# ./make inst–all
```

which does the installation. This command will try to build all the directories you have specified, using *mkdir*. This means that the parent of each of these directories must exist for the the *mkdir* to succeed.

The second each–time activity, is that if you are already running the ISODE, then you will need to kill and restart the *tsapd* (8c) daemon, otherwise incoming connections will not be initialized correctly. Otherwise, start the daemon now. From the *CShell*, the command might be:

```
# $(SBINDIR)tsapd >& /dev/null
```

The daemon will automatically detach. If you do not redirect the daemon's standard–error, then it will not detach, instead printing messages as to what actions it is taking.

That's about it. This will install everything. To clean-up the source tree as well, then use:

```
% ./make clean
```

at this point. Note that if you are planning on generating or installing FTAM or VT or QUIPU (described below), then you should not clean-up the source tree until after you are finished dealing with these.

If your system is configured for TCP/IP, and you are not already running an SNMP agent, then you are **URGED** to immediately install the SNMP agent distributed with the ISODE. Consult the **NETWORK MANAGEMENT** section below.

Finally, if you are interested in discussing the ISODE with others running the software, drop a note to the Internet mailbox ‘‘ISODE–Request@NIC.DDN.MIL’’, and ask to be added to the ‘‘ISODE@NIC.DDN.MIL’’ list.

TAILORING

If you create a file called **$(ETCDIR)isotailor**, then you can customize the behavior of the programs which use the ISODE when they start. Consult the **support/isotailor.5** file for further information.

USER PROGRAMS

By default, two services are installed.

The first service, having programs *isoc* and *isod*, is used to test out the installation of the ISODE on your system:

```
% ./make test
```

which runs the *isode–test* script.

The second service, having programs *imisc* and *ros.imisc*, is a small demo service supporting things like *finger*, *who* and so forth.

There are additional programs in the **others/** directory. These aren't integral parts of the system and assume that the ISODE is already installed. Use at your own discretion.

REGISTERING OSI APPLICATION SERVICES

Earlier releases of the ISODE relied on static tables to keep track of the OSI application services offered on an end-system. This is a problematic exercise in keeping local and remote tables synchronized. In this release of the ISODE, the OSI Directory can be used to manage this information, thereby automating the synchronization process.

Preparation

Once you have installed the ISODE, you must bring up a DSA. The procedures for doing this varies, depending on your location; consult the section "Setting up an Initial DSA" in Volume 5 of the *User's Manual*.

Once your DSA is running, you should build the DMD for your organization. Underneath the entry for your organization, you should select an area where your end-system's application entities will reside in the DIT. For example, the OSI application services available in PSI's Santa Clara office reside somewhere under:

```
c=US
  @o=Performance Systems International
  @ou=Research and Development
  @ou=Santa Clara
```

Note that this area may very well be different than the value of the ‘‘local_DIT’’ in your dsaptailor file. In general, all the end-systems at a site will have the same "local_DIT" value, but each of those end-systems offering OSI application services will place those services at a different portion in the DIT (usually somewhere underneath the ‘‘local_DIT’’ value).

By convention, all the OSI application services offered by a given end-system are placed in the same location in the DIT, under an applicationProcess entry with the short name of the end-system, e.g., ‘‘cn=cheetah’’. So, using the example above, the entry

```
c=US
  @o=Performance Systems International
  @ou=Research and Development
  @ou=Santa Clara
```

```
@cn=cheetah
```

would contain all the entries of interest.

Once-only Installation

The *bootsvc* script will generate a shell script that will create an applicationProcess entry and then an entry for each of the OSI services provided by the ISODE. So, you must first select the RDN for the applicationProcess entry.

Run *bootsvc* to create a script:

```
% support/bootsvc <<aP-name>> > run.sh
```

e.g.,

```
% support/bootsvc cheetah > run.sh
```

Note that the first line of this script is used to define the network address where *iaed* listens for incoming connections. By default, only the address for the Internet community (RFC1006) is set. If the end-system is configured for other OSI communities, then this line should be changed accordingly, e.g.,

```
A="Internet=‘hostname‘|NS+aabbcc"
```

Next, start *dish* in the background, bind as the manager, move to the location in the DIT where the services are to be registered and run the script, e.g.,

```
% setenv DISHPROC "127.0.0.1 ‘expr $$ + 32768‘"
% bind -u <<DN of DSA Manager>>
% moveto "ou=Research and Development@ou=Santa Clara"
% sh run.sh
```

Following this, you need to arrange for *iaed* rather than *tsapd* to run when the machine goes multi–user. On Berkeley UNIX systems, replace these lines to the **/etc/rc.local** file:

```
if [ –f $(SBINDIR)tsapd ]; then
  $(SBINDIR)tsapd >/dev/null 2>&1 &
  (echo –n ’ tsap’) > /dev/console
fi
```

with:

```
if [ –f $(SBINDIR)iaed ]; then
  $(SBINDIR)iaed -D ’ou=Research and ...@cn=services’ >/dev/null 2>&1 &
  (echo –n ’ iae’) > /dev/console
fi
```

On other systems, a similar procedure is followed.

When *iaed* starts, it will connect to the Directory, find the services contained therein, and start listening as appropriate.

Finally, when the Directory software was installed, this included a program called *dased*. If you have not already done so, edit the **$(ETCDIR)isotailor** file to have these two lines:

```
ns_enable: on
```

```
ns_address: Internet=domain-name+17006
```

where ''domain-name'' is the DNS name or IP-address of the machine which is running *dased*. This can be a different machine than the one running the DSA, but it's probably best to have the local DSA and *dased* running on the same machine.

Next, arrange for *dased* to be started when the machine goes multi-user. On Berkeley UNIX systems, add these lines to the **/etc/rc.local** file:

```
if [ –f $(SBINDIR)dased ]; then
   $(SBINDIR)dased >/dev/null 2>&1 &
   (echo –n ' dase') > /dev/console
fi
```

On other systems, a similar procedure is followed.

When *dased* starts, it will listen for incoming connections from initiator ISODE programs. (By default, the initiator programs aren't loaded with the user-friendly nameservice and the DAP, owing to the code size--instead, they talk to *dased*.)

For your other systems, edit the **$(ETCDIR)isotailor** file to have these two lines:

```
ns_enable: on
ns_address: Internet=domain-name+17006
```

where ''domain-name'' is the DNS name or IP-address of the machine which is running *dased*.

To test the system:

```
% isode-test -iaed
```

If all goes well, users should be able to type things such as

```
% ftam cheetah,sc,psi,us
```

and ''the right thing'' will happen (i.e., local users can access remote services, even if they have not been entered into the entities database).

Adding New Services

The installation procedures need be performed only once. If you decide to disable a service, simply remove the corresponding entry from the Directory. To add a new service, see the Section ''Defining New Services'' in the *User's Manual*.

FTAM/FTP gateway

Because the FTAM/FTP gateway is meant to appear as an FTAM entity, the entry for this service must be placed in a different portion of the DIT than the regular FTAM service. As such, the *bootsvc* script will not install this service.

Hence, if you wish to run such a service, you will have to install it manually. The entry might be something like this:

```
objectClass= top & quipuObject &\
        applicationEntity & iSODEApplicationEntity
cn= <<whatever you want>>
presentationAddress= <<unique transport selector>>/<<end-system's NSAP>>
supportedApplicationContext= iso ftam
```

```
acl=
execVector= iso.ftam-ftp
```

Look in your part of the Directory to see some examples of what these entries look like. The are some scripts described in *others/quipu/tools/scripts/READ-ME* which can be used to maintain such entries.

FILE TRANSFER, ACCESS AND MANAGEMENT

In addition, if you are running the ISODE on a Berkeley or AT&T System V UNIX system, then there is also an implementation of the ISO FTAM. FTAM, which stands for File Transfer, Access and Management, is the OSI file service. The implementation provided is fairly complete in the context of the particular file services it offers. It is a minimal implementation in as much as it offers only four core services: transfer of text files, transfer of binary files, directory listings, and file management.

To generate FTAM, go to the **isode–7.0** directory and type:

```
% ./make all-ftam
```

This will cause a complete generation of the FTAM libraries and programs. If all goes well, proceed with the installation. If not, complain as there ‘‘should be no problems’’ at this step.

You will need to be the super-user to install FTAM:

```
# ./make install–ftam
```

That's about it. This will install everything and then clean–up the source tree. Note that if you are planning on generating or installing the FTAM/FTP gateway (described below), then you should not clean-up the source tree until after you are finished dealing with the gateway. In this case, or if you just want an installation and no clean–up, then use:

```
# ./make inst–ftam
```

instead.

FTAM/FTP GATEWAY

In addition, if you are running the ISODE on a Berkeley or AT&T System V UNIX system, there is also an implementation of an FTAM/FTP application gateway. The gateway is actually two programs: one which acts as an ftam responder and an ftp client, and the other which acts as an ftp server and an ftam initiator. Note that the gateway currently resides at a different location than the standard FTAM responder and FTP server. (This may be corrected in a future release.) Read the manual entries for *ftamd-ftp* (8c) and *ftpd-ftam* (8c) for the details.

To generate the FTAM/FTAM gateway, go to the **isode–7.0** directory and type:

```
% ./make all-ftam-ftp
```

This will cause a complete generation of the gateway. If all goes well, proceed with the installation. If not, complain as there ‘‘should be no problems’’ at this step.

You will need to be the super-user to install the FTAM/FTP gateway:

```
# ./make install–ftam-ftp
```

This will install everything and then clean–up the source tree. If you just want an installation and no clean–up, then use:

```
# ./make inst–ftam-ftp
```

instead.

Regardless of the command you use, on 4.2BSD-derived systems, add this line to your **/etc/servers** file:

```
ftp-ftam    tcp    $(SBINDIR)in.ftpd-ftam
```

On 4.3BSD-derived systems, add this line to your **/etc/inetd.conf** file:

```
ftp-ftam    stream    tcp    nowait    root    $(SBINDIR)in.ftpd-ftam    in.ftpd-ftam
```

Finally, add this line to your **/etc/services** file:

```
ftp-ftam      531/tcp
```

VIRTUAL TERMINAL

In addition, if you are running the ISODE on a Berkeley UNIX system, there is also an implementation of the ISO VT. VT is the OSI terminal service. The implementation provided is roughly comparable to an average telnet implementation.

To generate the VT system, go to the **isode–7.0** directory and type:

```
% ./make all-vt
```

This will cause a complete generation of the VT initiator and responder programs. If all goes well, proceed with the installation. If not, complain as there ‘‘should be no problems’’ at this step.

You will need to be the super-user to install VT:

```
# ./make install–vt
```

That's about it. This will install everything and then clean–up the source tree. If you just want an installation and no clean–up, then use:

```
# ./make inst–vt
```

instead.

DIRECTORY SERVICES

In addition, if you are running the ISODE on a Berkeley UNIX or AT&T System V UNIX system, there is also an implementation of the OSI Directory, called QUIPU. If you're not interested in running a Directory, skip this text and go to the section entitled **GENERATING DOCUMENTATION**.

Each host using the OSI directory implicitly runs a Directory User Agent (DUA). Additionally, you may wish to run a Directory System Agent (DSA) on some hosts. As such, the instructions which follow indicate which activities are necessary in both instances, as appropriate.

QUIPU GENERATION

To generate QUIPU, go to the **isode–7.0** directory and type:

```
% ./make all-quipu
```

This will cause a complete generation of the DSAP library and the DSA. If all goes well, proceed with the installation. If not, complain as there ‘‘should be no problems’’ at this step.

QUIPU INSTALLATION

You will need to be the super-user to install QUIPU:

```
# ./make install–quipu
```

This will install everything and then clean–up the source tree. If you just want an installation and no clean–up, then use:

```
# ./make inst–quipu
```

instead. After either command, there is one once-only activity.

The QUIPU DSA is a ''static responder''. This means that it accepts new associations and managing old ones as necessary. Hence, if you intend to run a local DSA, it is necessary to start the *ros.quipu* daemon when the machine goes multi-user. On Berkeley UNIX systems, add these lines to the **/etc/rc.local** file:

```
if [ –f $(SBINDIR)ros.quipu ]; then
   (cd /usr/etc/quipu-db; $(SBINDIR)ros.quipu >/dev/null 2>&1) &
   (echo –n ' quipu') > /dev/console
fi
```

(This assumes your database is in the directory **/usr/etc/quipu-db** - it need not be) On other systems, a similar procedure is followed.

QUIPU DATABASE

If you intend to run a local DSA, then you will need to build a Directory database. (If you are already running QUIPU 5.0 or later, then you've done this before and so you can skip to the next section on **QUIPU TAILORING**.) The database directory, by default, lives in the ETCDIR area (usually **/usr/etc/**) under the name of **quipu-db/**. Three prototype databases can be found in the directory **others/quipu/quipu-db/**. These database files should be protected as they contain Directory passwords and other sensitive information. The DSA needs to be able to read this information, and so performs a setuid on execution to the UID of the owner of the database directory.

Now customize the chosen prototype database under **/usr/etc/quipu-db/**. The details of this database are explained in Volume 5 of the users manual. However you should be able to derive a minimal database by following the example structure defined for University College London in the GB branch of the Directory tree. Then delete the example structure for O=University College London.

QUIPU TAILORING

If you choose to run a local DSA, now configure it. The DSA tailors itself at runtime by reading the file **$(ETCDIR)quiputailor**. A prototype of this file will be installed during the normal ISODE installation process. Only one entry in the file usually needs to be changed:

```
mydsaname        CN=toucan
```

Substitute the name of the DSA as it occurs in the Directory for ''CN=toucan''. See the section in Volume 5 for a description of the full range of tailoring options in the **$(ETCDIR)quiputailor** file.

Now configure the various DUA programs. These tailor themselves at runtime by reading the file **$(ETCDIR)dsaptailor**. A prototype of this file will be installed during the normal ISODE installation process. Only one entry in the file usually needs to be changed:

```
dsa_address     toucan  localHost=17003
```

Substitute the name of your ''primary'' DSA for ''toucan'' and its corresponding presentation address for the '''0101'H/Internet+...'' string. This information can be found in the Directory on the host which is running the DSA.

Do not confuse the *dsa_address* used in this file with the *ns_address* used in the **$(ETCDIR)isotailor** file. These are separate services and must live at different addresses. See *quiputailor* (5) for a description of the full range of tailoring options in the **$(ETCDIR)dsaptailor** file.

QUIPU ONCE-ONLY

Having tailored QUIPU, you can now start the DSA. However, if you are already running QUIPU, then you will need to kill and restart the QUIPU DSA.

Start the DSA now. From the *CShell*, the command might be:

```
# $(SBINDIR)ros.quipu >& /dev/null
```

The daemon will automatically detach. If you do not redirect the daemon's standard–error, then it will not detach, instead printing messages as to what actions it is taking.

NETWORK MANAGEMENT

In addition, if you are running the ISODE on a Berkeley UNIX system, there is also an implementation of the SNMP. Although this is not the OSI network management service, Inasmuch as the continued survival of the Internet hinges on all nodes becoming network manageable, this package was developed using the ISODE and is being freely distributed with releases of Berkeley UNIX.

It must be stressed that this package is not a complete network management system. In particular, whilst *snmpd* provides a minimal agent functionality, there are no Network Operation Center (NOC) tools--*snmpi* is a debugging aid only.

To generate the SNMP system, go to the **isode–7.0** directory and type:

```
% ./make all-snmp
```

This will cause a complete generation of the SNMP agent and the minimal SNMP initiator program. If all goes well, proceed with the installation. If not, complain as there ‘‘should be no problems’’ at this step.

There are two once–only activities which must be performed prior to installation. First, check your **/etc/services** file, and verify that these three lines are present:

```
snmp        161/udp
snmp-trap   162/udp
smux        199/tcp
```

If not, add them.

Second, add these lines to the **/etc/rc.local** file:

```
if [ –f $(SBINDIR)snmpd ]; then
    $(SBINDIR)snmpd >/dev/null 2>&1 &
    (echo –n ’ snmp’) > /dev/console
fi
if [ –f $(SBINDIR)smux.unixd –a –f $(SBINDIR)snmpd ]; then
    $(SBINDIR)smux.unixd >/dev/null 2>&1 &
    (echo –n ’ smux-unix’) > /dev/console
fi
```

You will need to be the super-user to install SNMP:

```
# ./make install–snmp
```

This will install everything and then clean–up the source tree. If you just want an installation and no clean–up, then use:

```
# ./make inst–snmp
```

instead.

Regardless of the command you use, read the comments in the **$(ETCDIR)snmpd.rc** file which will tell you how to tailor the agent for your installation.

Finally, if you are already running the SNMP, then you will need to kill and restart the *snmpd* (8c) and SMUX UNIX daemons. (It is best to kill *smux.unixd* first, and then *snmpd*.) Otherwise, start the daemons now. From the *CShell*, the command might be:

```
# $(SBINDIR)snmpd >& /dev/null
# $(SBINDIR)smux.unixd >& /dev/null
```

The daemon will automatically detach. If you do not redirect the daemon's standard–error, then it will not detach, instead printing messages as to what actions it is taking.

LIGHTWEIGHT PRESENTATION PROTOCOL

In addition, if you are running the ISODE on a Berkeley UNIX system, there is also an implementation of RFC1085, the lightweight presentation protocol for TCP/IP-based internets.

To generate the LPP system, go to the **isode–7.0** directory and type:

```
% ./make all–lpp
```

This will cause a complete generation of the LPP library and support programs. If all goes well, proceed with the installation. If not, complain as there ‘‘should be no problems’’ at this step.

You will need to be the super-user to install the LPP system. There are two kinds of activities: once–only activities that you perform the first time the software is installed; and each–time activities that you perform every time the software is installed.

The first once–only activity is to verify that the *lppd* daemon will be run when the machine goes multi–user. On Berkeley UNIX systems, add these lines to the **/etc/rc.local** file:

```
if [ –f $(SBINDIR)lppd ]; then
   $(SBINDIR)lppd >/dev/null 2>&1 &
   (echo –n ' lpp') > /dev/console
fi
```

On other systems, a similar procedure is followed.

The next once–only activity is to verify that systems with a native **/etc/services** file contain an entry for the miscellany service. This is used when the ISODE miscellaneous services is run using the LPP. If not, add the line:

```
miscellany    17002/lpp
```

to the **/etc/services** file. If your system does not have such a file, the software automatically compensates for this.

There are two each–time activities:

```
# ./make install–lpp
```

This will install everything and then clean–up the source tree. If you just want an installation and no clean–up, then use:

```
# ./make inst–lpp
```

instead.

Regardless of the command you use, the second each–time activity, is that if you are already running the LPP system, then you will need to kill and restart the *lppd* (8c) daemon, otherwise incoming connections will not be initialized correctly. Otherwise, start the daemon now. From the *CShell*, the command might be:

```
# $(SBINDIR)lppd >& /dev/null
```

The daemon will automatically detach. If you do not redirect the daemon's standard–error, then it will not detach, instead printing messages as to what actions it is taking.

That's about it.

GENERATING DOCUMENTATION

The directory **doc/** contains the documentation set for this release. Consult the file **doc/READ–ME** for a description of each document. The directory **doc/ps/** contains PostScript versions of each document. Usually it is easier to print the files in this directory than generate the documentation from scratch as the sources to these documents are in either LaTeX (for papers) or SLiTeX (for presentations).

If you received this distribution from the network, then the directory **doc/ps/** does not contain any PostScript files. There should be a separate compressed *tar* file, containing only PostScript files, available on the machine where you retrieved this distribution.

FILES

Too numerous to mention. Honest.

SEE ALSO

The ISO Development Environment: User's Manual

AUTHOR

Marshall T. Rose
with assistance from a cast of thousands (read the **Preface** in the *User's Manual*)