ISODE Daemons

C. J. Robbins
X-Tel Services Ltd

C.Robbins@uk.co.uk

October 3rd, 1991

What daemons are there?

ros.quipu: The QUIPU DSA process.

dad: Directory Assistance Daemon.

tsapd: This is the ISODE transport daemon.

iaed: Version of tsapd that uses the directory.

dased: Directory interface for programs such a ftam.

tsbridge: Transport Service Bridge. This can be used to give other machines on your ethernet X.25 access.

This is covered in detail elsewhere so details are not repeated in these slides.

These are all found in the `ETCDIR` directory. This is typically `/usr/local/etc` or `/usr/etc`.

Starting the Daemons

Should all be started with the output redirected to a file. If not the programs are in DEBUG mode and will behave differently.

So under the C Shell you might use:

```
daemon_name any_flags >& /dev/null
```

Under the Bourne shell (as in the rc.local file!) you might use:

```
daemon_name any_flags > /dev/null 2>&1
```

The order in which the daemons are started *does not* matter. If the QUIPU DSA is not available, they will wait until it is!

ros.quipu

This is the stand alone QUIPU DSA process.

There should be one `ros.quipu` process per DSA.

All connections over DAP and DSP to the directory connect directly to this one process.

It does not rely on *any* of the other daemons.

It must have it's own unique network address — a different X.25 sub-address to the `tsapd`.

dad — Directory Assistance Daemon

This is an optional deamon and can be used by DUA programs such as `fred` to gain fast access to the directory.

It is also used by the PSI Macintosh and PC DUAs.

When started, it will invoke a `DISH` process and sit waiting for connections on a default TCP address.

To get `fred` to use `dad` rather than access the directory directly you should first start `dad` (from your rc.local!) then add

```
set    server          your_machine_name
```

to your `fredrc` file, found in the `ETCDIR`

tsapd — Transport Service Access Point Daemon (1)

This listens for connections on an NSAP address and for each incomming connection starts a new process to handle the connection.

The NSAP is defined in one of two ways:

Command line arguments:

> The address can be given on the command line, see the `tsapd` manual page for details.

Using defaults:

> For TCP, the `tsap` entry is looked for in the `/etc/services` file, and the associated TCP port used.
> For X.25 the `x25_local_dte` variable in your `isotailor` file defines the DTE.

tsapd — Transport Service Access Point Daemon (2)

When a connection arrives, the Transport Selector (TSEL) from the source address is matched with the TSELs given in the `isoservices` file. A typical line is

```
"tsap/filestore"  #259  iso.ftam -c
```

Which means, when the TSEL "#259" is used, invoke the "tsap/filestore" by running the `iso.ftam` program with the `-c` flag. (`-c` forces a chroot to the initiator's home directory to restrict network access via ftam.)

The prgram `iso.ftam` is expected to be in the `ETCDIR` unless a full path name is used.

iaed — Isode Application Entitiy Daemon (1)

Exactly the same principle as `tsapd`; but instead of static files, the directory is used for configuration. It is used instead of`tsapd`.

To start the daemon, there are generally two parameters. For example at X-Tel we use:

`-u '@c=GB@o=X-Tel Services Ltd@cn=lancaster'`:

> When `iaed` connects to the directory it will use the given DN in the bind argument. A password could also be given with `-p` flag if required.

`-D '@c=GB@o=X-Tel Services Ltd@cn=lancaster'`:

> This defines where in the DIT `iead` should look for configuration info.

Example DIT entries are given on Slides 10 and 11.

iaed — Isode Application Entitiy Daemon (2)

When `iaed` starts, it searchs the given part of the DIT for objects belonging to the object class `isodeApplicationEntity`.

For each object found, the `presentationAddress` and `execVector` attributes are read. `iaed` will listen on the given address.
(Replaces the `isoentity` lookup used by `tsapd`)

When a connection comes in on a given address, the process defined by the `execVector` attribute of the corresponding entry is started and passed control of the connection. (Replaces the `isoservices` lookup used by `tsapd`)

The directory is periodically re-read to look for changes in the attributes.
Sending `iaed` a `SIGHUP` using the `kill` command will also cause the directory to be re-read.

dased — Directory Access Service Element Daemon

This is used by user processes such a `ftam` and by other daemons such as `iead` to find the presentation address of OSI services.

It should be started with a parameter of the following form:

`-u '@c=GB@o=X-Tel Services Ltd@cn=lancaster'`:

> When `dased` connects to the directory it will use the given DN in the bind argument. A password could also be given with `-p` flag if required.

The `dased` daemon listens for connections on the presentation address defined by the `ns_address` address variable in `isotailor`.

The `ns_enable` variable should be set to "on".

Typically, it will be given a UFN and an application context by the user process.
`dased` will resolve the UFN, then search below that point in the DIT for an entry with the given application context in the `supportedApplicationContext` attribute.
Finally, `dased` will return the value of the `presentationAddress` attribute.

Typical DIT layout for dased and iaed (1)

For each service you want to run, you should have an entry in the directory.

Below X-Tel we have a simple entry for a host called 'lancaster'...

```
cn= lancaster
objectClass= applicationProcess & quipuNonLeafObject
seeAlso= c=GB@o=X-Tel Services Ltd@cn=trellis
masterDSA= cn=Vampire Bat
slaveDSA= c=GB@cn=Thin-spined Porcupine
```

Below this entry we have (amongst others)...

```
commonName=file service
commonName=isode miscellany
commonName=pp qmgr
commonName=x40088
```

Typical DIT layout for dased and iaed (2)

The "file service" (FATM) entry looks like:

```
cn= file service
objectClass= quipuObject & iSODEApplicationEntity
presentationAddress= '0103'H/Janet=00002100102999
supportedApplicationContext= iso ftam
execVector= iso.ftam -c
```

This is an `iSODEApplicationEntity` object, so `iaed` listens on the given address. When a connection comes in, on the given TSEL (`'0103'H`) the program `iso.ftam` as defined by `execVector` will be invoked (with a `-c` flag).

The "pp qmgr" entry is slightly different...

```
cn= pp qmgr
objectClass= applicationEntity & quipuObject
presentationAddress= '1001'H/LOCAL-ETHER=...
supportedApplicationContext= 0.9.2342.60200172.201.1
acl=
```

This entry is only an `applicationEntity` and not an `iSODEApplicationEntity`, so `iaed` will not listen on the given address (the PP qmgr is already doing that!). However, the `dased` will still be able to find the service.

Managing the DIT entries

The `bootsvc` program from the `support` directory of the ISODE source tree, can be used as a one off to create the directory entries in the first place. Full details are in the `isode-gen` manual page.

After that, they can be managed by hand using DISH, or by using the DISH scrpits in the others/quipu/tools/scripts directory of the source tree. There is a READ-ME file in this directory explaining their use.