Agouti	Alpaca	Anaconda	Queen Anglefish
Anole	Carpenter Ant	Leaf Cutter Ant	Giant Anteater
Two toed Anteater	Antshrike	Aperea	Arapaima
Armadillo	Giant Armadillo	Hairy Armadillo	Six-banded Armadillo
Brazilian Three-banded Armadillo	Axolotl	Fruit Bat	Vampire Bat
Spectacled Bear	Long Haired Beetle	Passionflower Leaf Beetle	Rhinocerous Beetle
Emarald Tree Boa	Red-footed Booby	Brocket	Map Butterfly
Cacique	Dwarf Caiman	Black-headed Caique	Capuchin
Black-capped Capuchin	Capybara	Caracara	Crested Cariama
Cascabel	Andean Cat	Pampas Cat	Little Spotted Cat
Armoured Catfish	Blue Leopard Catfish	Pimelodid Catfish	Southern Mountain Cavy
Weasel Cavy	Cayman	Dragon Finned Characin	Rainbow Characin
Chinchilla	Long-tailed Chinchilla	Short-tailed Chinchilla	Cichlid
Dwarf Cichlid	Oscar Cichlid	Pike Cichlid	Triangle Cichlid
Greater Pichi Ciego	Lesser Pichi Ciego	Coati	Brown Coati
Ring-tailed Coati	False Cobra	Condor	Andean Condor
Boa Constrictor	Giant Cookroach	Giant Coot	Horned Coot
Flightless Cormorant	Land Crab	Rock Crab	Curassow
Black Curassow	Brazilian White-headed Curassow	Darter	Marsh Deer
Pampas Deer	Argentinian Pampas Deer	Swamp Deer	Bush Dog
Small-eared Dog	Amazonian Dolphin	Douroucouli	Inca Dove
Scaly Dove	Muscovy Duck	Torrent Duck	Black-chested Buzzard Eagle
Harpy Eagle	Ornate Hawk Eagle	Black and White Hawk Eagle	Electric Eel
Moray Eel	Plumbeous Forest Falcon	Spotted Silver Dollar Fish	Four-eyed Fish
Hatchet Fish	Hillstream Fish	Leaf Fish	Pampadour Fish
Pencil Fish	Andean Flamingo	James's Flamingo	Andes Fox
Crab-eating Fox	Southern Gray Fox	Greenhouse Frog	Horned Frog
Marsupial Frog	Arrow Poison Frog	Surinam Frog	Gallinule
Gar	Juan Fernandez Island Goat	Yellow Pocket Gopher	Grison
Brazialian Blue Grosbeak	Grunion	Guanaco	Guppy
Galápagos Hawk	Humpback Headstander	Little Blue Heron	Yellow-crowned Night Heron
Hoatzin	Hornero	North Andean Huemul	South Andean Huemul
Small Hummingbird	Hummingbird	Black-chinned Hummingbird	Black Faced Ibis
Scarlet Ibis	Iguana	Jaguar	Jaguarundi
Pileated Jay	Kinkajou	Everglades Kite	Cayenne Lapwing
Teiid Lizard	Llama	Peruvian Longfin	South American Lungfish
Mabuya	Macaw	Military Green Macaw	Hyacinth Macaw
Scarlet Macaw	Manatee	Amazonian Manatee	Caribbean Manatee
Mara	Margay	Buffy-headed Marmoset	Buffy-tufted-ear Marmoset
Common Marmoset	Cotton-top Marmoset	Black Eared Marmoset	Goeldi's Marmoset
Golden Lion Marmoset	Tassel-eared Marmoset	White Marmoset	Molly
Brown Howler Monkey	Red Howler Monkey	Black Spider Monkey	Brown-headed Spider Monkey
Geoffroy's Spider Monkey	Long-haired Spider Monkey	Woolly Spider Monkey	Squirrel Monkey
Red Titi Monkey	Woolly Monkey	Yellow-tailed Woolly Monkey	Morpho
Cypri's Morpho	Anopheles Mosquito	Motmot	Roufous Motmot
American Cane Mouse	Red Nosed Mouse	Ocelot	Oilbird
Wolly Opossum	American Oriole	Baltimore Oriole	Giant Brazilian Otter
Giant Otter	Marine Otter	La Plata Otter	Southern River Otter
Roufous Ovenbird	Burrowing Owl	Long Moustached Owl	Spectacled Owl
Paca	Pacarana	American Paddlefish	Pangolin
Parrakeet	Quaker Parrakeet	Yellow-fronted Amazon Parrot	Blue-headed Parrot
Red-fan Parrot	White-fronted Parrot	Chacoan Peccary	Collared Peccary
Brown Pelican	Galápagos Penguin	Humboldt's Penguin	Rockhopper Penguin
Peropatus	Guinea Pig	Fruit Pigeon	Piranah
Thin-spined Porcupine	Tree Porcupine	Puculet	Nothern Pudu
Southern Pudu	Sooty Capped Puffbill	Puffbird	Puma
Forrest Rabbit	Crab-eating Raccoon	Clapper Rail	Giant Wood Rail
Chinchilla Rat	Spiny Rat	Rhea	Cock of the Rock
Saki	Southern Bearded Saki	Monk Saki	White-nosed Saki
Scalare	Screamer	Horned Screamer	South American Sea-lion
Elephant Seal	Juan Fernandez Fur Seal	Galápagos Fur Seal	Sloth
Maned Sloth	Brazilian three-toed Sloth	Blind Snake	Coral Snake
Canana Rat Snake	Urutu Snake	Pygmy Seed Snipe	Solendon
Atlantic Spadefish	Couch's Spadefoot	Bird-eating Spider	Silk Spider
Roseato Spoonbill	South American Red Squirrel	Variegated Squirrel	Jaribu Stork
Sunbittern	Black-necked Swan	Coscorba Swan	Bare-face Tamarin
Cotton-top Tamarin	Emperor Tamarin	Golden Lion Tamarin	Golden-headed Lion Tamarin
Golden-rumped Lion Tamarin	Red Mantled Tamarin	Pied Tamarin	Black and Red Tamarin
White-footed Tamarin	Tanager	Violet headed Tanager	Central American Tapir
Young South American Tapir	Mountain Tapir	Tarpon	Buenos Aires Tetra
Bentos Tetra	Cardinal Tetra	Bleeding Heart Tetra	Small Scaled Tetra
Tinamou	Masked Titi	Burrowing Toad	Columbian Toad
Darwin's Toad	Giant Tortoise	Galápagos Giant Tortoise	Yellow-legged Tortoise
Toucan	Red Billed Toucan	Saffron Toucan	Trogon
Violaceous Trogon	Ocellated Turkey	Black-headed Uakari	Red Uakari
Red and White Uakari	Vicuna	Viscacha	Black Widow
Maned Wolf	Woodcreaper	Black Cheeked Woodpecker	Bar Winged Wood Wren
Yapok	Zebu		

Table 14.1: Endangered South American Wildlife

Having located its own entry, a DSA will know its network address by looking at the `presentionAddress` attribute, hence it can start listening for operations.

But first it must load the database. This process is described in full in Section 14.8.

In its own entry it will find a `edbInfo` attribute, this describes which EDBs the DSA is expected to hold. The format of the `edbInfo` attribute is described fully in Section 14.10, but essentially the first parameter supplied says "load this EDB". Thus the attribute

```
edbInfo = ##
edbInfo = c=US@o=The Wollongong Group ##
edbInfo = c=GB ##
```

requests that the EDB files

quipu-db/EDB

(signified by no data before the first "`#`" sign),

quipu-db/c=US/o=The Wollongong Group/EDB

and

quipu-db/c=GB/EDB

are loaded.

There are a few other important attributes your DSAs entry should have, the "toucan" entry in Figure 14.2 on page 138 gives examples of the other attributes. They are briefly described below.

`description:` A textual message describing the DSA, and the wildlife!

`quipuVersion:` This should contain the version number of the QUIPU software you are using. This can be found by using the DISH command `squid -version`. You only have to set this once. From then on the DSA will manage it, updating the entry when required.

`manager:` The DN of the manager of the DSA. This user will not be blocked by access control when modifying the local database over DAP.

`supportedApplicationContext:` This should always have the value

```
QuipuDSP & X500DSP & X500DAP & InternetDSP
```

for this version of QUIPU. It is used by remote DSAs to decide which protocols your DSA supports, and thus how best to contact it.

`objectclass:` This must contain "`quipuDSA`".

`listenAddress:` Sometimes, (particularly in the X.25 world, or when using a transport service bridge), the address a server should listen on, is not the same as the address clients should call. The presentationAddress attribute defines the address clients will always call. If, and only if, this is not the address the server should listen on, the address for the server should be given in the listernAddress attribute.

`acl:` A DSAs entry, like any other, can be protected by access control lists. Care should be taken to make sure all remote DSAs can see all the attributes they need. The following ACL is recommended for use in a DSAs entry. The string "`MANAGER DN`" should be replaced with the distinguished name of the DSA manager as defined by the "manager" attribute.

```
acl= group # MANAGER DN # write # entry
acl= group # MANAGER DN # write # default
acl= others # read # entry
acl= others # read # default
acl= others # read # child
```

A little explaination may help as this might seem somewhat cryptic. The `others # read # child` line is to prevent anybody from adding children to the entry; it is the only line referencing the children, so nobody is given write access. The userpassword attribute in the entry does not need protecting for two reasons. Firstly nothing can make use of it, self has only got read access to the entry (as defined by the others clause). Secondly, the password will probably be in a highly replicated EDB file, so despite the pseudo crypting, is easily breakable.

`relayDSA:` If your DSA is not connected to one of the major networks (Internet, IPSS...), it may from time to time get references to a DSA that it cannot connect to directly. For the operation to succeed, your DSA will need to chain the operation to a DSA that can progress the query.

This attribute is the DN of a DSA or DSAs that are connected to both your network and the major networks you are not connected to. There needs to be an agreement between the managers of the two DSAs because the relay DSA will be asked to perform operations on your behalf. For example, if the DSA "x" has access to IPSS and Internet, but DSA "y" only has IPSS access, DSA "y" might add a relayDSA attribute to it own entry, with the DN of DSA "x" as the value. Then, when DSA "y" gets a reference to an Internet based DSA, it will chain the operation the DSA "x". Clearly, if every DSA chooses the same relay DSA, that DSA will soon become overloaded and reject your association attempts with a "busy" error. So some care is needed in choosing the "right" DSA (The QUIPU team recognise that there needs to be some form of "relay authorisation" and are looking at possible solutions for future versions of QUIPU). Section 14.3 describes how to prevent your DSA (e.g., DSA "x") from chaining operations on behalf of other DSAs.

`photo:` A picture of the wildlife !

Section 14.7 describes how you can add data to your DSA by extending the supplied textual database to include your own data or by sending data to the directory via the DUA modify operations. This may be done independently from connecting to the global directory.

14.3 Tailoring

The previous sections have described how to start a basic DSA, and connect a DUA to it. Having done this, there are various configuration options you can set, which are described in this section.

On startup the DSA first consults a runtime tailor file (the default DSA tailor file is called *quiputailor* in the ISODE `ETCDIR` directory, but can be

changed with a `-t` parameter to *ros.quipu*; consult *quipu*(8c) for details), which indicates such things as:

- name of the DSA.
- location of the database.
- location and level of the logs that the DSA will produce.
- location of any other DSAs for initial bootstrap

At startup, the *isotailor*(5n) file is also consulted to configure the system-wide ISODE parameters.

The format is identical to the DUA *tailor* file described in Section 13.2, with the addition of the following options:

`mydsaname:` The distinguished name of the DSA. For example,

```
mydsaname   cn=Axolotl
```

declares this DSA to have the common name of `Axolotl`. Quotes will be needed in the name contains a space characer, for example

```
mydsaname   "c=GB@cn=Long Moustached Owl"
```

`parent:` This entry consists of a name/address pair of a parent DSA. 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. For example,

```
parent C=GB@CN=vicuna Internet=vs1.ucl.cs.ac.uk+50987
```

declares the parent DSA to be C=GB@CN=vicuna at the indicated address. If more than one `parent` tailor entries are found, the DSA will chose which DSA to contact. The first DSA in the list is taken as the master reference, with the subsequent entries as slave references.

If your DSA holds a copy of the ROOT EDB file, this parameter should have just one value which is a reference to the DSA holding the master copy of the root EDB file (currently cn=Giant Tortoise).

`stats`: The value has the same format as the `dsaplog` entry described in Section 13.2, and is used to control the level of DSA statistical logging.

`treedir`: Defines the directory in which the textual database is stored. For example,

```
treedir     /usr/etc/quipu-db/
```

declares the directory */usr/etc/quipu-db/* to contain the local part of the directory information tree. Be sure to remember the trailing slash "`/`".

`shadow`: When searching, often a large part of the time is involved with chaining off to other DSAs to search aliases. To enhance performance it is sometimes useful to have a cached copy of the alias locally, this tailor variable allows such attributes to be "spot shadowed". The value of this variable is an attribute type. The attribute value associated with this type should have DN syntax. When the database has been loaded, if any instance of this attribute references a DN that is not held locally your DSA will "spot shadow" that entry. That is from time to time it will read the entry from a remote DSA, and cache the result.

`optimize_attr`: If TURBO_INDEX has been defined, this option specifies the attribute types to index. Each attribute type must be on a separate line. For example the lines:

```
optimize_attr commonName
optimize_attr surname
```

would arrange to index both the commonName and surname attributes. Only string attributes are allowed. *Note*: this option must come before any `index_subtree` or `index_siblings` options.

`index_subtree`: If TURBO_INDEX has been defined, this option specifies the distinguished name of a subtree to index for subtree searches. Multiple subtrees may be specified by multiple lines. For example, the lines

```
index_subtree "c=US@o=your org"
index_subtree "c=US@o=your org@ou=really big OU"
```

arrange to have two indexes built, one for each subtree specified. Be aware that the indexes take up extra space in core, so care should be taken to index things sparingly.

`index_siblings:` If TURBO_INDEX has been defined, this option specifies the distinguished name of the parent of a group of siblings to index. For example, the line

```
index_siblings "c=US@o=your org@ou=really big OU"
```

arranges to have an index built for one-level searches directly below the specified entry. Multiple indexes may be specified by multiple lines. Be aware that the indexes take up extra space in core, so care should be taken to index things sparingly.

`optimized_only:` If TURBO_INDEX has been defined, the value **on** tells the DSA to refuse any searches that do not consist entirely of optimized attributes and filters. Any such non-indexed queries will be rejected with an "unwilling to perform" service error.

`update:` The value **on** tells the DSA to update slave and cached EDB files when it starts up. See Section 14.10 for further details; by default this parameter is **off**.

`searchlevel:` Defines the level below which users will be able to search the DIT, for example, default 2 (e.g., below organisations). If they try to search from higher up, a n"unwilling to perform" service error will result.

`lastmodified:` If the value is **off**, the attributes `lastmodifiedby` and `lastmodifiedtime` will not be added by the DSA when an entry is altered.

`readonly:` Bring the DSA up in read only mode, that is, prevent user modification. Slave updates are still allowed.

`dspchaining:` The value **on** tells the DSA it is allowed to chain DSP requests to other DSAs if necessary. The default mode of operation is to return a DSA referral, unless a chain is needed due to a disjoint network. The full set of issues deciding whether to use chaining or referrals is discussed in the QUIPU design document ([Kil89b]), and [BR89].

`adminsize:` The administrative size limit for use on search and list operations.

`admintime:` The maximum time allowed to spend on a user query.

`cachetime:` The length of time to keep / use "cached" information.

`conntime:` The length of time to hold a unused connection open.

`nsaptime:` The length of time to wait before deciding a connection cannot be established for a given NSAP.

`retrytime:` The length of time before deciding it is worth attempting to connect to a DSA that could not be contacted earlier.

`slavetime:` The length of time between attempting to update slave EDB files.

`preferdsa:` When a DSA has creates a reference to other DSAs it tries to discriminate, and choose the "best" DSA to contact. This variable gives you a handle on controlling the choice. For example the lines

```
preferDSA c=GB@cn=Vicuna
preferDSA "cn=Giant Tortoise"
```

tell the DSA to use either c=GB@cn=Vicuna or cn=Giant Tortoise in preference to any other DSA. Furthermore, use the DSA c=GB@cn=Vicuna rather than cn=Giant Tortoise if possible.

`cainfo:` For authentication.

`secretkey:` For authentication.

`bindwindow`: For authentication.

`isode`: The argument is an `isodevariable isodevalue` pair as would be found in *isotailor*. This is used to "override" isotailor settings.

`authentication`: The argument describes the minimum level of authentication required to bind to the DSA over DAP. The value should be one of:

`none`: Anonymous binds are accepted.

`dn`: A DN must be supplied.

`simple`: Simple authentication must be used.

`protected`: Protected simple authentication must be used.

`strong`: Strong authentication must be used.

`reject_prefix`: Do not allow DAP bind operations from DNs with the specified prefix.

`reject_length`: Do not allow DAP bind operations with DNs having less that n components.

`relay_for`: Only DSP chain operations for the specified DSAs.

`accept_prefix`: Allow DAP binds from the given prefix ONLY.

`getedb_size`: For a getEDB operation, an EDB is split into slices of n entries, default 10. This parameter allows you to change this number.

`edbtmp_path`: The directory used to store temporary fragments of EDB during a getEDB operation — default */tmp*.

14.3.1 Tailoring a Running DSA

The tailoring described above is performed when the DSA is booted. It is sometimes useful to alter the tailor settings after the DSA has started without having to bring the DSA down and reboot. This can be done via the special *dsacontrol* command in the DISH program as described in Section 4.5 on page 46.

The DUA invokes a modify operation, with a single special attribute `dSAControl` defined in [Kil89b]. The DSA recognises the special attribute, and provided you are bound as the manager of the DSA, passes the attribute value to the appropriate routine.

[Kil89b] describes this process in full.

14.4 Modifying your DSA's Entry

A DSA manages its own entry in the DIT. Generally the master EDB in which your DSAs entry resides is not held by your DSA. For security reasons, this means it is difficult for a DSA to modify its own entry directly, so, for example it cannot keep its version number up to date. However, a DSA holding the master EDB knows that any QUIPU DSA[4] wants to manage its own entry. To allow this to happen, the DSA holding the master EDB "spot shadows" the remote DSA entry. That is, from time to time it connects to the DSA in question, reads its entry, and writes the result back into the master EDB file. So modify operations on the DSA's entry can now take place in your local DSA. This has the advantage that attributes such as the version number are kept up to date.

To perform this shadowing, the DSA has to read its own entry across an un-authenticated DSP link; thus it cannot read any attributes that are protected by ACLs. So all important attributes in the DSAs entry *must* be publicly readable (this includes the unused userPassword attribute). If they are not readable the shadowing operation will fail.

To keep the database consistent with the master EDB, when a modify takes place, your DSA cannot rewrite the information back to its slave copy EDB. So it writes the entry in a file called *DSA.real* in the top level of your database. If there is an inconsistency found, it is this file that is "trusted".

There is also a file called *DSA.pseudo*, which contains some attributes managed by the DSA, that it does not need to make public (such as which cached EDB files it holds). You should *never* need to edit this file.

There is a problem when modifying the presentation address of a DSA. You must make sure the DSA with the master EDB reads the new address, *before* you move the DSA. If not, it will always attempt to connect to the

[4] Any QUIPU DSA for which the version number is 6.8 or greater