UK Academic Community Directory Group

Preparing data for inclusion an X.500 Directory

Paul Barker

Department of Computer Science
University College London
United Kingdom

Date: April 1990

ABSTRACT

This note is intended to be a practical guide on preparing data for inclusion in the the U.K. Academic X.500 Directory Pilot. While the focus is on the U.K. Pilot, much of the document has general applicability to both academic and commercial sites, in particular those sites running the Quipu X.500 system.

This note is intended to augment the information in the Quipu manual [1], and also to make some of the information therein more digestible. However this note falls far short of being a replacement for the manual, and Directory administrators will still need to refer to the Quipu manual for detail.

Some of the discussion that follows refers to general X.500 issues, while other parts, for example those on access control and EDB files, necessarily refer specifically to the Quipu system. Aspects discussed include: the depth of the Directory Information Tree; choice of Relative Distinguished Names; selected object classes and attribute types; Quipu access control lists; format of EDB files.

A number of example entries are specified to clarify the points discussed.

UK Academic Community Directory Group

Preparing data for inclusion an X.500 Directory

Paul Barker

Department of Computer Science
University College London
United Kingdom

Date: April 1990

1. Introduction

This note is intended to remedy the current dearth of practical information given to sites participating in the U.K Academic Directory Pilot on how they should prepare their data for inclusion in an X.500 Directory. The note follows the X.500 recommendations where they give specific guidance. In addition, where appropriate, the consensus view which has emerged through discussions in the Directory Group is expressed. Furthermore, guidance is given on the use of some Quipu-specific details: for example, object classes and access control lists.

Areas that are discussed are:

- The depth of the Directory Information Tree
- Choice of Relative Distinguished Names
- Selected Object Classes
- Selected Attribute Types, including access control lists

To make the discussion more concrete, a number of examples of Directory entries are given in Appendix A. These are specified in the Entry Data Block (EDB) format used by the Quipu Directory system, currently used in the Pilot.

2. Naming organisations and people in the Directory

The Directory is a distributed database, hierarchically structured in the shape of a tree. Figure 1 gives an example Directory Information Tree (DIT) with a few entries.

Large entities, such as countries or international organisations, appear under the root of the tree. Medium size entities, such as national organisations, appear further down the tree. Entries for individuals and other small entities appear at the leaves of the tree.

Every entry in the Directory has a *distinguished name* (DN) which unambiguously identifies that entry. The distinguished name of an object can be derived from the structure of the DIT by concatenating the *relative distinguished names*, (RDNs) from the root downwards, of the entries higher up the tree. Thus, from figure 1, the leaf entry with

a RDN of "CN=Staff Room" has the distinguished name
"C=GB, O=University College London, OU=Computer Science, CN=Staff Room".

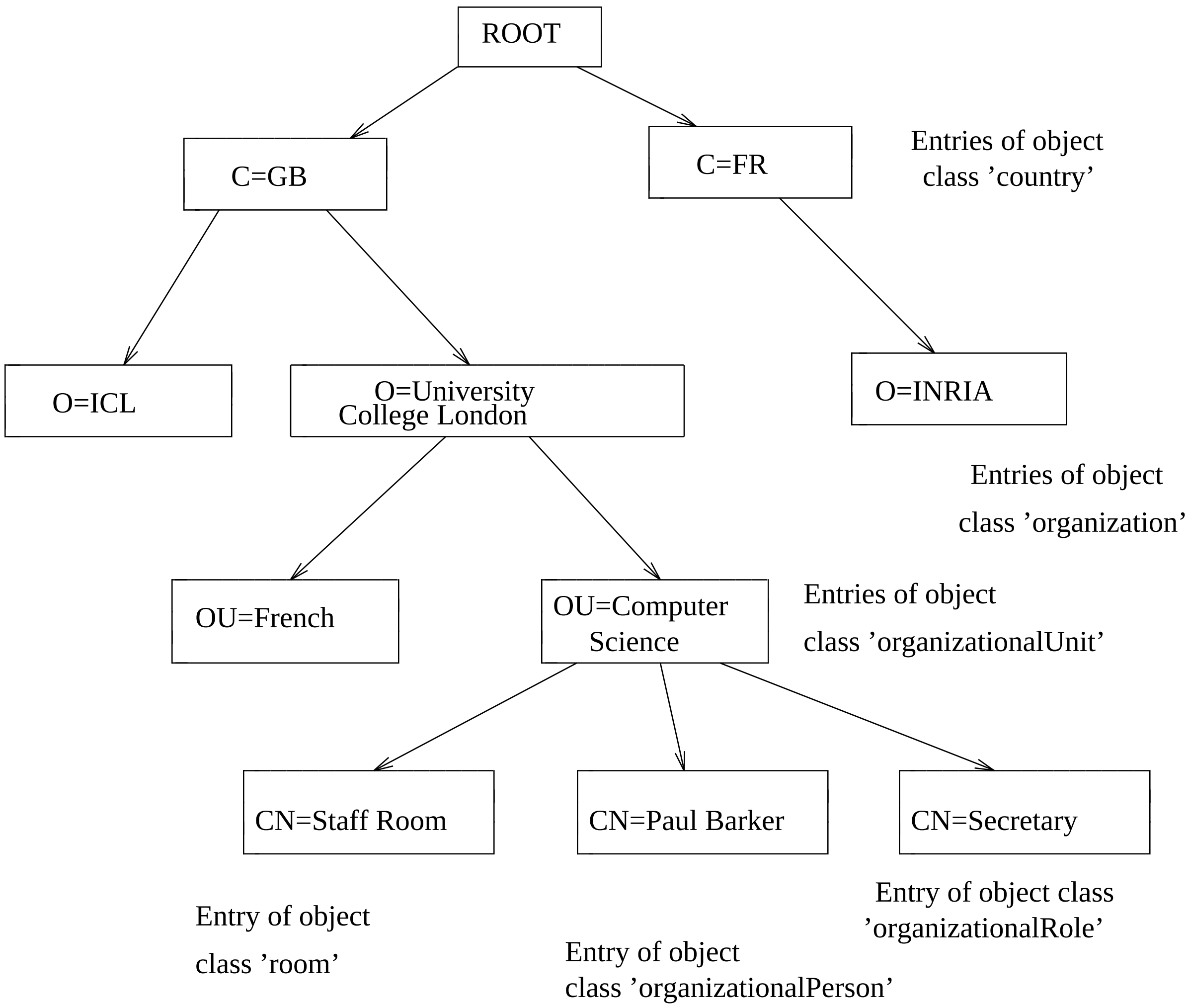

Figure 1: Hypothetical DIT

There are thus two factors which determine the directory name of an object:

- The depth of the Directory Information Tree
- Choice of relative distinguished names

These are now discussed in turn.

2.1. The depth of the Directory Information Tree

The X.500 standard allows a considerable degree of flexibility with regard to the depth of the tree. Some levels in the tree, such as country, may be omitted, while others such as organisational unit, can be used at one level, at several levels, or omitted entirely. Figure 2 shows part of the schema recommended in the X.500 standard.

However, it is strongly advised that U.K. pilot sites follow the more restricted model which is now described.

All U.K. academic sites should appear immediately under the "C=GB" entry. It is inappropriate for them to appear immediately beneath the root of the tree as U.K. universities and polytechnics are not supra-national organisations.

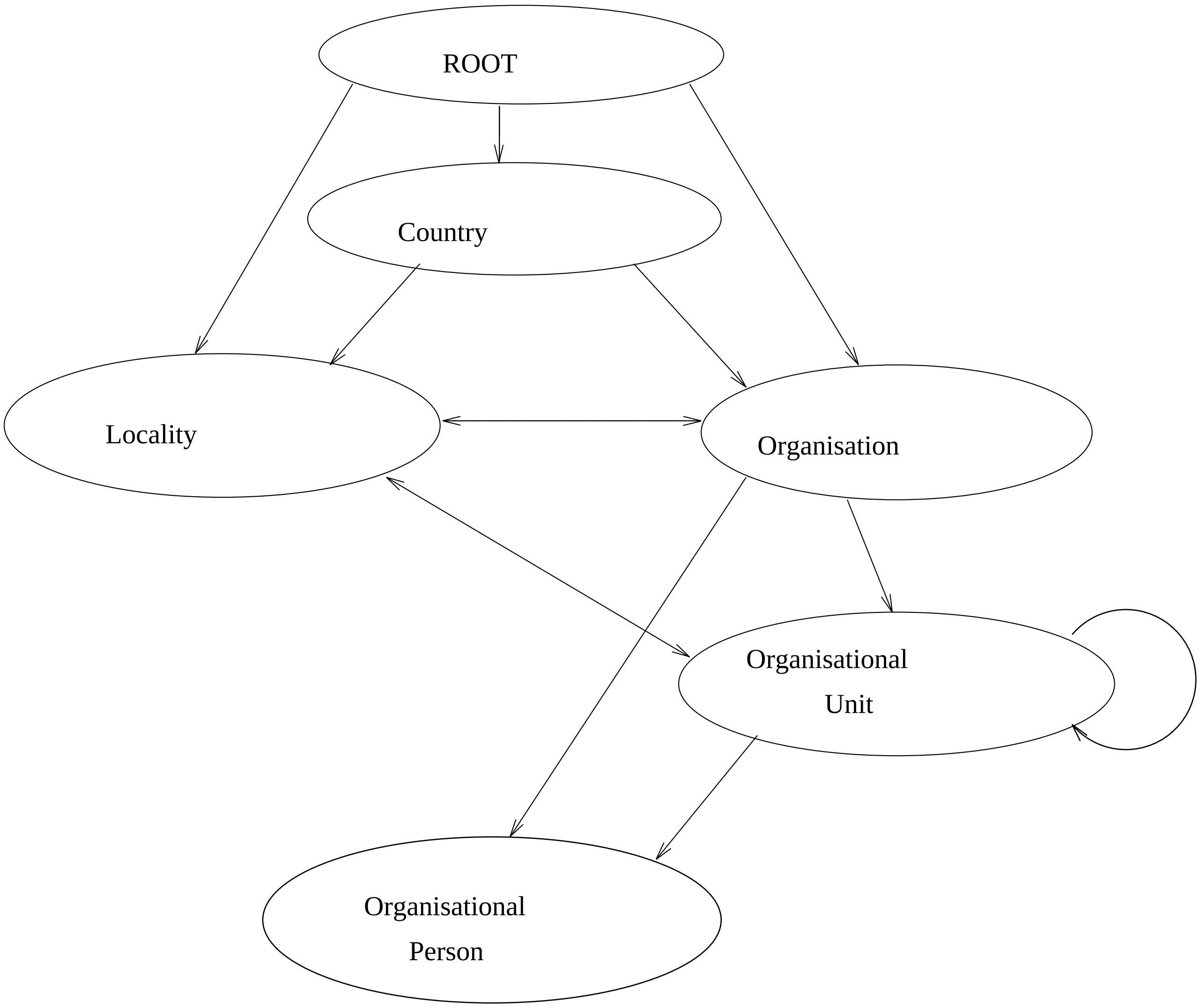

Figure 2: Part of suggested DIT structure recommended in X.500

The general presumption is for a DIT which tends to flatness rather than depth. This implies that usually one, but at most two, levels of organisational unit should be used, although a small institution might consider not using organisational units at all. It may be tempting to organise a university's name space in some highly structured manner, taking account of college, faculty, department, staff/student and even further sub-divisions. However this leads to the construction of very long and unnatural names. It seems reasonable that the hierarchy chosen should be tailored at least as much to the requirements of the Directory user searching for information as it is to the administrative or management structure of the organisation. This criterion militates against a deep DIT structure.

2.2. Choice of relative distinguished names

Just as the shape of the tree should be chosen to facilitate querying of the Directory, the RDNs chosen to name entries should be as meaningful as possible to everyone. The naming strategy agreed by the U.K. Directory Group is that RDNs should usually be the commonly known long name of the object or person. For organisations and organisational units, this means that sets of initials should be avoided, unless the longer name form is not widely known. Long, not commonly used, over-formal names should also be avoided. This policy becomes more important the further down the tree one goes. Initials here have highly specialised, not universally understood, meanings. While people in computer science departments in universities know that CS is short for Computer Science, most people do not. Indeed, people running Computing Services might even dispute this interpretation!

The following examples show various "good" and "bad" RDN choices.

Good choice	Bad Choice
University College London	UCL
Brunel University	Brunel
Crudthorpe University	The Royal and Most Pompously Benificent University of Crudthorpe
Computer Science	CS
Paul Barker	Paul John Fred Bert Sid Barker

3. Object classes and naming attributes

Every entry in the Directory has to have an *object class* attribute. This defines what sort of object the entry purports to describe. This "typing" of objects determines what sort of attributes the entry can have. While figure 1 shows some hypothetical entries, each with a single object class, in practice an entry will often have multiple object classes. This allows the entry to have a number of sets of attributes. It is suggested that entries created should have certain Quipu and THORN object classes, as well as the basic object class defined in the standard. This allows the use of non-standard attributes, such as access control lists, photos and favourite drinks. Furthermore, a site can experiment with additional attributes and object classes by adding their own local definitions to the *oidtables*. This topic is beyond the scope of this document. See section 13.10 of the Quipu manual for details of how to do this.

In addition to the object class attribute, every entry must have a RDN. The RDN is simply an attribute value of the entry used for naming. The standard recommends that certain attributes should be used as the naming attribute for each object class. The Directory Pilot is following these recommendations.

The following table adds detail to the above discussion by showing which object classes and naming attributes should be used when creating entries for some typical real world objects.

Object type	Basic Object Class	Recommended additional object classes	Naming attribute
University	organization	quipuNonLeafObject	organizationName
University department	organizationalUnit	quipuNonLeafObject	organizationalUnitName
Person	organizationalPerson	thornPerson quipuObject	commonName
Role	organizationalRole	quipuObject	commonName
Room	room	thornObject quipuObject	commonName

Table 1: Objects, object classes and naming attributes

NOTE: Aliases should not be used for the purpose of attributing a role to a personal entry. Aliases should only be used when one wants to be able to reference an entry in 2 separate naming contexts. If a person split their work equally between two departments within an organisation, it would be desirable to have two directory names for that person. One of these names would correspond to the person’s entry, and the other would be an alias which references that entry.

Aliasing could also be used to reference an entry outside the scope of a single organisation. For example, "c=gb@o=networkshop@cn=a.findlay" could be an alias to "c=gb@o=brunel University@ou=Manufacturing and Engineering Systems@Dr A J Findlay"

4. Attributes

The object class or classes selected for an entry define the set of attributes that the entry both must and may contain. The following table gives some indication of the range of attributes that can be assigned to the objects detailed in Table 1, if the suggested object classes are used.

Object type	Must contain attribute	May contain attribute
University	organizationName, objectClass	acl, description, seeAlso postalAddress, postalCode, streetAddress, telephoneNumber, facsimileTelephoneNumber, telexNumber
University department	organizationalUnitName objectClass	as for University
Person	commonName objectClass surname	acl, localityName, streetAddress postalAddress, postalCode telephoneNumber, facsimileTelephoneNumber, description, seeAlso, userPassword title, info, photo, favouriteDrink, userid, userClass, secretary, homePhone, homePostalAddress, textEncodedORAddress, rfc822Mailbox
Role	commonName objectClass	acl, roleOccupant, seeAlso, description postalAddress, postalCode, streetAddress, telephoneNumber, facsimileTelephoneNumber
Room	commonName objectClass	acl, info, photo, seeAlso telephoneNumber, description

Table 2: Objects, mandatory attributes, and (some of) the optional atttributes

The usage of the majority of the above attributes should be self-explanatory. However, the usage of a few of them, in particular the *access control list,* merits some elucidation or comment.

The *acl,* or *access control list* attribute, is used to specify the type of access allowed, on which entries or attributes, by which user or group of users. It is categorised above as an optional attribute. This is slightly misleading. It IS optional to include the attribute within an entry in the EDB file. However, if it is not present, then the DSA will automatically add a default access control list attribute.

It is important to note the semantics of default access control. It gives public read access to all attributes, and write access to the owner of the entry.

In fact things are a little more complicated than this. Before write access is allowed, the owner of an entry has to authenticate her/himself to the Directory. Authentication requires that a user provides a password, which is compared with the value of the password attribute in the users's entry. Thus write access is only possible on entries with a password attribute. However, as soon as you add a password attribute, the default acl is no longer sufficient, as this allows public read access to all attributes! Thus, if you add userPassword attributes to a personal entry, you **must** also have an explicit acl attribute. This should allow compare access to all for the password attribute, but **not** read access!

A further modification to the default access control will almost certainly be desirable, if users are allowed to modify their own entries. There will probably be certain attributes which should only be modified by a directory administrator; for example, commonName, surname, objectClass and the access control list itself.

It is worth noting that the term *directory administrator* used above is a rather imprecise one. It could mean one of two things. First, each DSA has a manager, identified by the *manager* attribute of the DSA entry. This person has the right to read, create, modify and delete any entry or attribute within that DSA. This is analagous to the root login on a UNIX system. It is also possible to give additional access privileges to other users, if they are explicitly named in entries' access control lists. This allows separation of the roles of DSA management and data management.

For the sake of both clarity and brevity, it is considered that the best detailed explanation of how to set up access control lists is by the use of examples. Appendix A contains sample acls for the following cases:

- Entries are to be read-only for all users
- Entries are to be read-only for all users, except a named administrator or group of administrators.
- Users will be able to modify some, but not all, attributes in their entries.

It is suggested that you follow the templates initially until you feel you need to do anything more complicated.

This is not the place for a full tutorial on access control. For those of you who wish to delve deeper, access control is explained at greater length in section 10.2 of the Quipu manual.

seeAlso is used to specify the DNs of other closely related objects. For example, when used as an attribute of a room object, seeAlso could be used to indicate the room's usual occupant(s).

postalAddress should be the full postal name and address as used for the physical delivery of paper mail and parcels (up to 6 lines of 30 characters each).

telephoneNumber, facsimileTelephoneNumber and homePhone should be stored using the full international format: i.e. +44 123 456789

description and *info* can be used for arbitrary descriptive text about the object.

title should be used to describe the position or function of an object within an organisation.

favouriteDrink is not a mandatory attribute, but should be!

userClass may be used to distinguish various categories of system user. For example, at UCL we differentiate between academic staff, research staff, support staff, 1st year students, MSc students, postgraduate students, etc.

secretary should be the DN of a person's secretary.

photo is a G3 fax encoded photograph. Consult sections 13.5.3 and 13.5.4 of the Quipu manual for more details.

NOTE: The assigning of some attributes, for example, both to a personal and a role entry can lead to a violation of the relational model, with concomitant updating problems. The crux of whether to duplicate the information within, in this case, the role entry depends on whether the role actually has an attribute associated with it in its own right. Consider the case of a departmental secretary. If the person filling the role left the organisation, the phone number associated with the role would remain the same. Thus the role should have a telephone number attribute. The same would not be true of a departmental library representative. As different individuals filled the role, so the phone number associated with that role would change. In this case, the phone number should not be an attribute of the role entry.

5. Quipu EDB files

Until data bulk loaders and data merging tools are produced, the simplest method of getting some information into the directory is to edit the DSA's EDB (Entry Data Block) files. These files are hierarchically organised within the UNIX file system, in a manner which corresponds to the structure of the DIT. This is explained at greater length in section 13.1.3 of the Quipu manual.

The first task when setting up a DSA for an organisation is to create an entry for the organisation and for the DSA as stated in the U.K. Pilot's Directory System Configuration Guide [2]. APPENDIX A contains templates for both these entries. When you have prepared them, you are should mail these entries to quipu-support, to become registered as part of the pilot.

You will probably now need to create some organizationalUnit entries to reflect the structure of your organisation. (It is assumed here that most universities are too large to consider putting all their entries immediately under the organisation entry.) Finally, you will need to create entries for people, rooms and roles. Example entries can be found in APPENDIX A.

6. References

[1] Stephen E. Kille, Colin Robbins, Michael Roe, Alan Turland, *Quipu*, in Volume 5 of Marshall T. Rose's "The ISO Development Environment Users's Manual (Version 6.0)", January 1990.

[2] Paul Barker, *Directory System Configuration Guide*, in University College London, Department of Computer Science Research Note 90/7, January 1990.

APPENDIX A

This appendix contains extracts of sample EDB files which illustrate the points above made in the text.

Each EDB file created should begin with the following two lines. The first says that the EDB is a master, as opposed to a replicated SLAVE copy. The second line is the UTC time when the EDB file was created. This format can be achieved with the following UNIX incantation: date +%y%m%d%H%M%SZ

```
MASTER
900323184046Z
```

Entries consist of a number of attribute=value pairs. In all cases, the first attribute type-value pair is the RDN. Each attribute type-value(s) pair must be on a single physical line (no line-wrapping). Multi-valued attributes can be represented in one of two ways. First, they can be represented as a number of separate lines of type-value pairs. Second, they can be represented by a sequence of "&" separated values on one physical line.

Entries are separated by one or more blank lines. Lines beginning with a "#" character are regarded as comment lines.

N.B. You may notice that the representations of some values in the EDB files are subtly altered after the files have been written back by the DSA when entries are modified. In particular, "@" characters will appear as \40 (the hexadecimal equivalent) in email addresses. This is because the "@" character is treated specially in some cases (within distinguished names) and the DSA is trying too hard to be careful not to confuse this character with a meta-character. Both forms are equally acceptable.

The remainder of the appendix gives examples of entries for DSAs, organisations, organisational units, people, roles, rooms and aliases.

DSA entry

```
# your chosen DSA name - must be South American wildlife
cn= llama
# address of the DSA - for detail on how to represent multiple addresses
# for multiple network types, see other (UCL's, for example) DSA entries once
# you can connect to the Directory
presentationAddress= Janet=12345678901234
# this stays as is
edbinfo= #cn=giant tortoise##
# self-explanatory
description= Description of your animal
description= Descriptions are often more than one line
# this stays as is
objectClass= quipuDSA
# The distinguished name of the dsa manager.  This can be a special entry or
#  be a user with special privileges
manager= c=gb@o=University of Life@cn=dsaManager
# userPassword is a mandatory attribute for DSAs (intended for DSA
#  authentication), but is currently unused! Put your DSA name here for now
```

userPassword= llama
# fill in the DSA manager name in the ACLS as appropriate
acl= group # c=gb@o=University of Life@cn=dsaManager # write # entry
acl= others # read # entry
acl= group # c=gb@o=University of Life@cn=dsaManager # write # default
acl= others # read # default
# The current value in ˜isode/isode*/quipu/quipuvrsn.c
quipuVersion=6.0 Distribution
# this stays as is
supportedApplicationContext=QuipuDSP & X500DSP & X500DAP

Organisation entry

# the name of your organisation
o= University of Life
# the position in the DIT of the entry for your DSA
masterDSA= c=GB@cn=llama
# fill in the DSA manager name as appropriate
acl= group # c=gb@o=University of Life@cn=dsaManager # write # child
acl= others # read # child
acl= group # c=gb@o=University of Life@cn=dsaManager # write # entry
acl= others # read # entry
acl= group # c=gb@o=University of Life@cn=dsaManager # write # default
acl= others # read # default
# this stays as is
treeStructure= quipuNonLeafObject & quipuDSA & applicationEntity & applicationProcess & person & organizationalUnit
# Organisations DTE
x121Address= 234219200300
# These should be obvious - note the formats
telephoneNumber= +44 71-387-7050
postalCode= WC1E 6BT
businessCategory= University
streetAddress= Gower Street
stateOrProvinceName= London
l= London
# this stays as is
objectClass= quipuNonLeafObject & quipuObject & organization & top
# alternate name form for the organisation - useful for searching
o= UL

Organisational Unit entry

# this entry shows a simple entry for an organizational unit, in this case
#   a university department.
# the omission of an acl attribute means that the default acl is implied
# default acl means that this entry can only be modified by the DSA manager
#   but the attributes may be read by all
# note the use of the "&" character in the objectClass attribute to separate
#   the multiple values of the attribute
ou=Electronic and Electrical Engineering
masterDSA= c=GB@cn=Giant Armadillo
objectClass= quipuNonLeafObject & quipuObject & organizationalUnit & top

```
# this is a more complicated entry for an organizational unit, showing a
#  large list of possible attributes
ou=Computer Science
slaveDSA= c=GB@cn=Giant Armadillo
masterDSA= c=GB@cn=Vicuna
lastModifiedBy= c=GB@o=University College London@ou=Computer Science@cn=incads
lastModifiedTime= 900423134417Z
seeAlso= c=GB@o=University College London@ou=Computer Centre
x121Address= 234219200300 & 000005111600
facsimileTelephoneNumber= +44 71-387-1397
telexNumber= 28722 $ G $ UCLPHYS
telephoneNumber= +44 71-387-7050 x 7214
telephoneNumber= +44 71-380-7214
postalCode= WC1E 6BT
postalAddress= Dept. of Compter Science $ University College London $ Gower Street $ London $ WC1E 6BT
businessCategory= Computer Science
searchGuide= person#cn$APPROX
searchGuide= person#cn$SUBSTR
searchGuide= person#uid$EQ
searchGuide= person#sn$EQ
ou= cs
streetAddress= Gower Street
stateOrProvinceName= London
l= pearson building
objectClass= quipuNonLeafObject & quipuObject & organizationalUnit & top
```

People, Room and Role entries

```
# this entry shows some of the considerable range of attribute types that
# can be associated with personal entries
# personal entries should have a commonName as the RDN and must also have a
# surname attribute
#
# there is no general mechanism for multi-line string attributes (the
# newline or return character is not included in the set of characters defined
# in printableString
#
# see the address field for an example of how to represent a multi-line
# postal address attribute.  This attribute has a special syntax will allows
# for multiple lines
#
# look very carefully at the access control attribute
# the first 2 lines of the acl allow all to compare a purported password with
#   the value in the entry (this is required by the bind operation), but
#   only the owner to modify it
# the third line contains a list of attributes which can only be read by all,
#   including the owner of the entry
# all attributes not mentioned in this list as read-only may still be written
#   by self (this is 'inherited' from the default acl)
cn=Paul Barker
homePostalAddress= 34 $ Gateley Rd $ London $ SW9 9SZ
secretary= c=GB@o=University College London@ou=Computer Science@cn=Maria Widdowson
```

```
userClass= csresstaff
roomNumber= G21
drink= guinness
mail= P.Barker@cs.ucl.ac.uk
textEncodedORaddress= /I=P/S=Barker/Ou=cs/O=ucl/Prmd=uk.ac/admd=gold 400/c=gb/
uid= paulb
telephoneNumber= +44 71-380-7366
sn= Barker
cn= Paul Barker
userPassword= SECRET
acl= others # compare # attributes # userPassword
acl= self # write # attributes # userPassword
acl= others # read # attributes # objectClass$accessControlList$surname$ etc.
objectClass= organizationalPerson & thornPerson & quipuObject

# another personal entry with read only access control
# the simplest way to achieve this is to omit the acl and userPassword
#   attributes
# use the seeAlso attribute to associate the entry associated with the role
#   performed by the person
# the title attribute should be used to describe a person's function within
#   an organisation
cn=Natalie May
userClass= csresstaff
roomNumber= G06
sn= May
title= Departmental Secretary
seeAlso= c=GB@o=University College London@ou=Computer Science@cn=Departmental Secretary
objectClass= person & top & thornPerson & thornObject & quipuObject


# yet another personal entry with read only access control, but with the
#   addition of a manager (other than the DSA manager) who is able to modify
#   the entry
cn=Ferdinand Bloggs
userClass= freeloader
roomNumber= 123
sn= Bloggs
acl = group # c=GB@o=XYZ University@cn=Joe Soap # write # entry
acl= group # c=GB@o=XYZ University@cn=Joe Soap # write # default
acl= other # read # entry
acl= other # read # default
objectClass= person & top & thornPerson & thornObject & quipuObject


# this is a typical room entry.
# room entries should have a commonName as the RDN.
# Note the use of the seeAlso attribute to associate a person with the room
cn=Staff Common Room
telephoneNumber= +44 71-387-7050 x 3681
cn= Staff Common Room
seeAlso=c=GB@o=University College London@ou=Computer Science@cn=Jon Crowcroft
objectClass= room & thornObject & quipuObject
```

```
# an example of a role entry.
# role entries should have a commonName as the RDN.
# Note the use of the roleOccupant attribute to associate a person (or
#   persons) with that role
cn=Departmental Secretary
roleOccupant=c=GB@o=University College London@ou=Computer Science@cn=Natalie May
telephoneNumber= +44 71-380-7214
cn=Departmental Secretary
objectClass=organizationalRole & quipuObject
```

Alias entry

```
# an example of an alias entry, illustrating required object classes
cn=Andrew Findlay
aliasedObjectName=c=gb@o=brunel University@ou=Manufacturing and Engineering Systems@Dr A J Findlay
objectClass=alias & quipuObject
```