Setting up an X.500 Directory Service

Dr Andrew J Findlay

Brunel University
Uxbridge, UK

Andrew.Findlay@brunel.ac.uk

ABSTRACT

One of the more difficult aspects of using electronic mail is finding the address of the person you want to talk to. At present there is no complete solution to this problem: a global directory is needed.

The X.500 standard describes a distributed directory service that has the potential to solve this problem. There are now several implementations of X.500 and the Directory is developing into a useful international service.

This paper introduces the X.500 system, and describes Brunel's experience with setting up and running a directory service based on ISODE/Quipu. Software configuration is briefly discussed and the data-gathering process is described. Suggestions are given for merging data from several sources to form coherent directory entries.

1. The Need for a directory

How often have you come back from a conference half-remembering somebody's name and then wanted to contact them? How many times do you refer to your organisations internal phone book each week? How many people come to you each week wanting to know the electronic mail address of somebody at a site you have never heard of?

Directories are a part of life. The problem (as with standards) is that there are so many to choose from. Just finding out what directories exist can be difficult; a directory of directories would be a useful thing!

Better still would be a single directory service, covering the whole world and listing everything that every existing directory contains. This single Directory would of course be totally user-friendly and would do exactly what was expected rather than what was asked for! This of course is impossible, but it is a reasonable description of what X.500 is trying to do.

2. The Scale of the Problem

To be really useful, the Directory must contain information on every person *in the World.*
Ultimately there will probably be three entries for each person:

- Residential person
- Organisational person
- Job title

Even if the X.500 Directory is restricted to people who can be contacted by some electronic means (telephone, telex, fax, electronic mail) the numbers are staggering.

Consider a few statistics:

Usenet News readership:	250 000
Telephones in the UK:	28 Million
People in the UK:	56 Million
People in the USA:	240 Million
People in China:	1000 Million?

It is obvious that a centralised directory is impossible. Apart from the amount of data, the number of queries must be considered. If each person in the directory is responsible for just one query each day, a central system would be swamped. Only a massively distributed directory system will stand any chance of working.

3. X.500

The X.500 standard was developed jointly by ISO and CCITT.[ISO/CCITT1988] It describes a distributed directory system based on a network of 'co-operating computer systems'. The standard lays down the communications protocols to be used, and defines the overall structure of the information to be held.

3.1. The X.500 Information Model

The X.500 Directory is structured as a tree, very much like the UNIX file system. This structure is referred to as the Directory Information Tree or DIT. Each node in the tree may contain information, in the form of attribute-value pairs. One or more of these attribute-value pairs are taken to be the 'name' of the node and are referred to as the Relative Distinguished Name, or RDN. For example, at the top of the tree there are nodes relating to countries; they have names like *country=GB* and *country=FR*. Below each country there are nodes representing organisations and geographical areas (Figure 1).

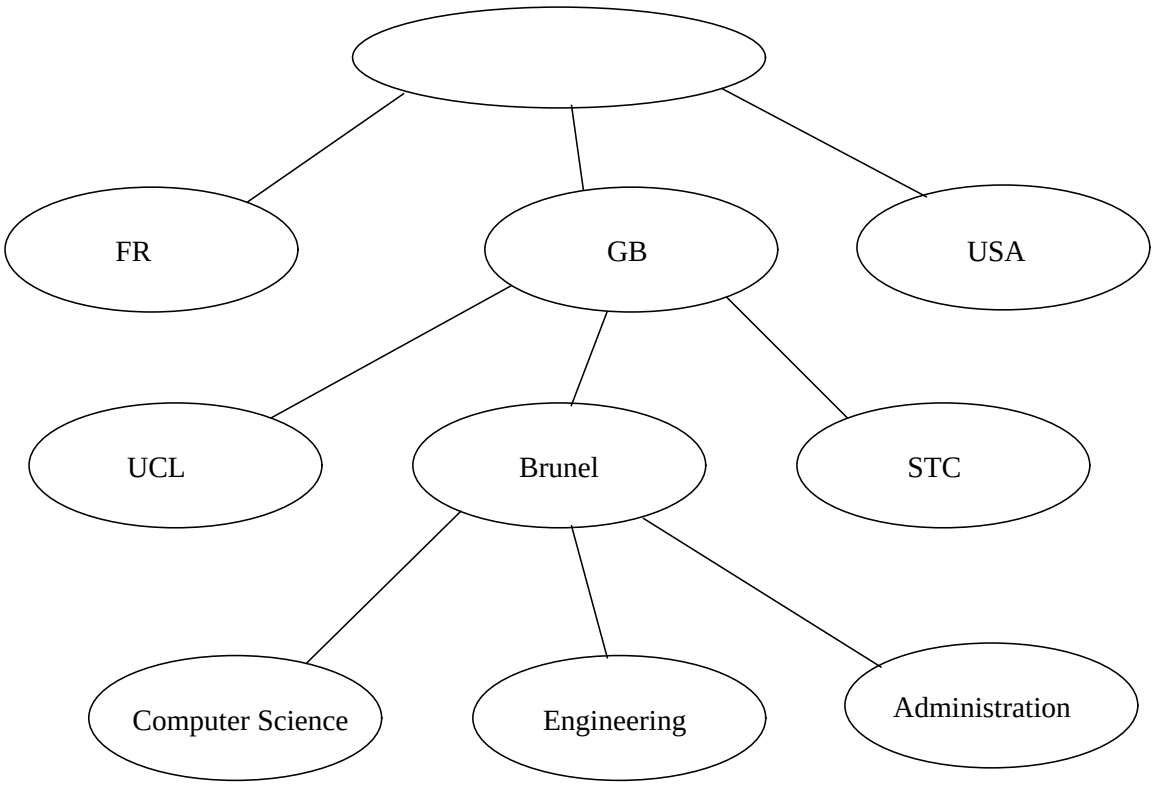

Figure 1: Part of the Directory Information Tree

Organisations can be further divided into *organisational units* nested to any level. Entries for people and

services are generally leaves of the tree.

Each node in the tree contains information relating to one object. For example, part of the entry describing Brunel University is shown in Figure 2.

Attribute name	Value
organizationName	Brunel University
organizationName	Brunel The University of West London
organizationName	Brunel
telephoneNumber	+44 895 74000
postalAddress	Brunel University Kingston Lane Uxbridge UB8 3PH UK
businessCategory	Education
description	Technological University offering sandwich courses
localityName	Uxbridge, West London

Figure 2: Entry describing Brunel University

Note that there are several values given for the attribute *organizationName.* The first one is the official name of the University, and is used to name the node in the DIT.

Further down the tree, entries for people may duplicate some of the information given at *organisation* level. This is because the current version of the X.500 standard does not define any attribute inheritance mechanism so each node has to be treated independently.

So far, the Directory has been described as a pure tree. In fact, it is not quite that restrictive as there are two mechanisms that allow cross-linking. A node may be set up as an *alias* for another node. In UNIX terms this is just like a symbolic link: two or more names can refer to a single object. As with symbolic links, it is possible to have an alias that does not point to a valid node. The other mechanism is the *see also* attribute, which allows a looser association to be established between two nodes. Together with other 'connecting' attributes, these mechanisms allow most organisations to be modelled quite well.

3.2. The X.500 Service Model

In line with other communication standards, X.500 divides the work to be done into 'user-related' functions and 'system' functions. Each user has a 'Directory User Agent' (DUA) which communicates with the directory system on their behalf. (Figure 3)

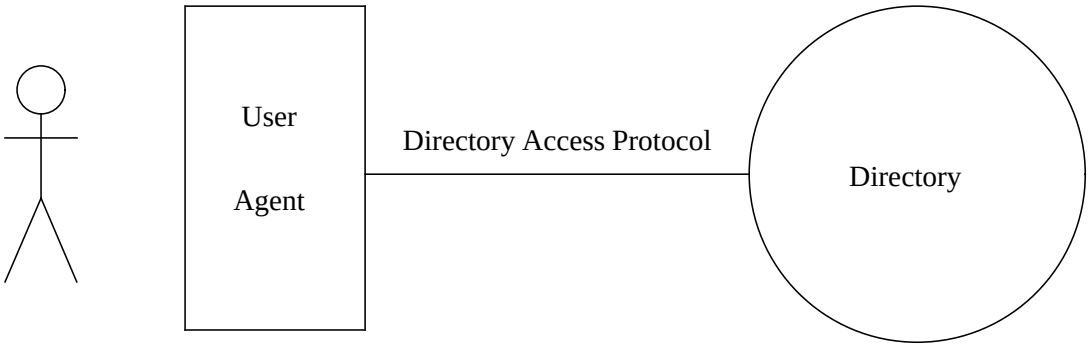

Figure 3: The X.500 service model

The directory system itself is comprised of a number of 'Directory System Agents' (DSAs). The DSAs communicate among themselves using a protocol called DSP (Directory System Protocol). A DUA

normally connects to a single DSA which provides all services required by the user, referring to other DSAs where necessary (Figure 4).

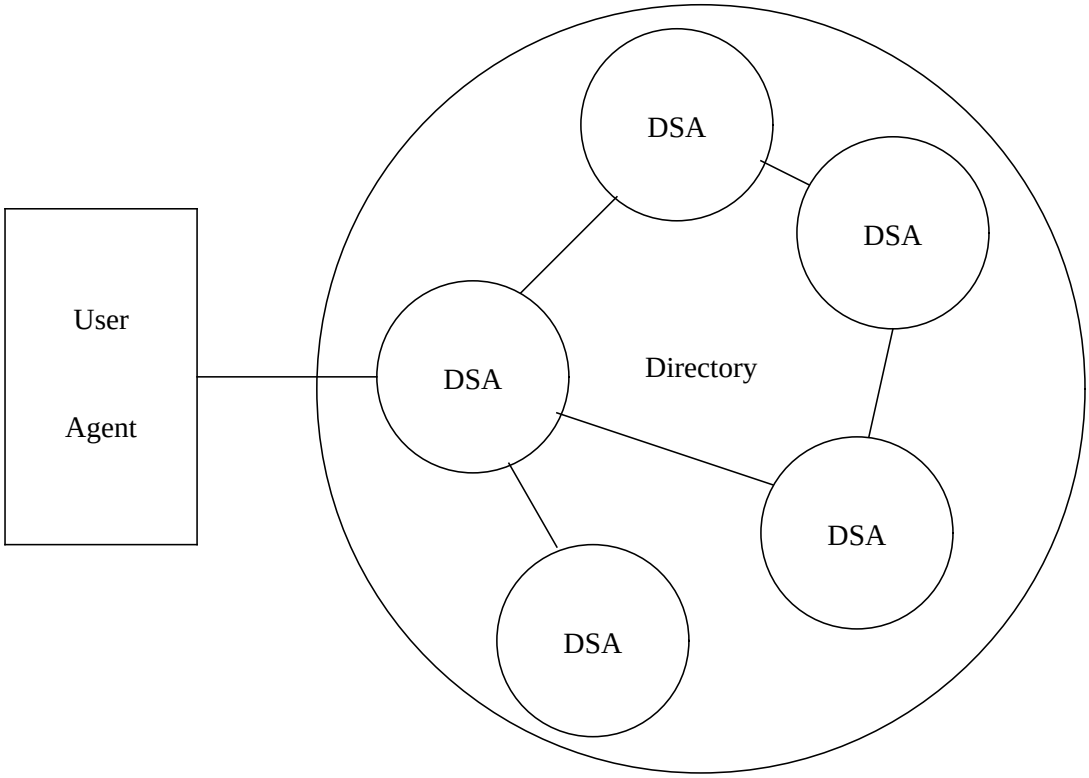

Figure 4: The distributed service model

The Directory Information Tree is distributed among the DSAs so that each subtree is held in a DSA 'near' the objects or people described in that subtree. A large organisation might have several DSAs; each could hold information relating to one or more departments. It is likely that some information would be replicated, copies being held in two or more DSAs to improve performance and reliability.

4. Quipu

Quipu is a freely-available implementation of X.500 which was designed at University College London and is distributed as part of the ISODE package.[Kille1988, Rose1990] Quipu will run on most Bsd-like UNIX systems, and can communicate using TCP/IP as well as X.25. The design places great emphasis on flexibility so that experiments can be carried out easily.

Rather than having an interface to a conventional database system, Quipu holds all data in memory. This means that searches can be carried out on any attribute with equal efficiency, as there are no index files. The disadvantage of this approach is that DSA processes use a lot of virtual memory and can cause heavy paging on small machines. Good performance can be obtained though, and response times below 1s have been demonstrated.

5. Configuration

For such a large package, ISODE is remarkably easy to build. A set of configuration templates are provided covering a wide range of machines. At Brunel we use Sun–3 Sun–4 and Pyramid machines, and the only changes needed to the standard files concern our preferences for filesystem layout.

ISODE is large: the source code takes up about 16MB, and at least 60MB more is needed to build the system. The installation process uses about another 20MB so this is not a game for sites with a disk shortage!

Building and installing Quipu is relatively easy; the hard part is tailoring it to your local conditions. The ISODE distribution contains several distinct modules. It is advisable to build and test the simpler ones before starting on Quipu. The command `./make all` will build the core libraries and the *imisc* 'little services' (OSI versions of *finger, write, rdate,* etc.) Once these simple services have been installed, they can be tested by hand or by using the automatic test program. At this stage, any glaring errors in the *isotailor* file can be sorted out.

With the simple services working, it is reasonable to start compiling Quipu with the command `./make all-quipu` This is usually straightforward, and Quipu can then be installed. If you have access to the Internet or an established X.25 network the Directory User Agents can be tested by connecting to somebody else's DSA e.g.: `dish -c giant`

The next stage is to set up a skeleton database for your own DSA. It is best to start from one of the example configurations and modify the entries one at a time with frequent tests. When you have a DSA with an entry for your own organisation, you can ask the manager of your nearest country-level DSA to add your details to the master database. Quipu DSAs usually update important data every six hours so a new DSA is soon widely recognised.

6. Data

A large organisation is a very complex thing. It may have many levels of hierachy and complex relationships between the individual parts. For example, a university might have several faculties, each containing several departments. The departments may have internal structure, being divided into research groups, teaching groups, administrative functions, and so on. To make matters worse, many people will be members of more than one group or even members of two departments.

When designing the structure of the DIT for an organisation, it is necessary to decide how closely the organisational structure will be modelled. Where an organisation has a well-defined structure, it is tempting to make the DIT match that structure exactly. This is probably a mistake. The main purpose of a directory is to allow people to locate other people and anything that might make that more difficult should be avoided. Although the structure of an organisation is usually well-understood by its employees, outsiders are likely to be confused by it. In any case, many organisations regard their internal structure as confidential information and are not willing to publish it. From a pragmatic point of view, the DIT structure should be easy to derive automatically from the available sources of data.

The DIT structure chosen for Brunel University is very simple, and follows the 'wide flat tree' approach used at higher levels. There is a single level of 'organisational units', which represent the departments. People appear at the next level down (Figure 5)

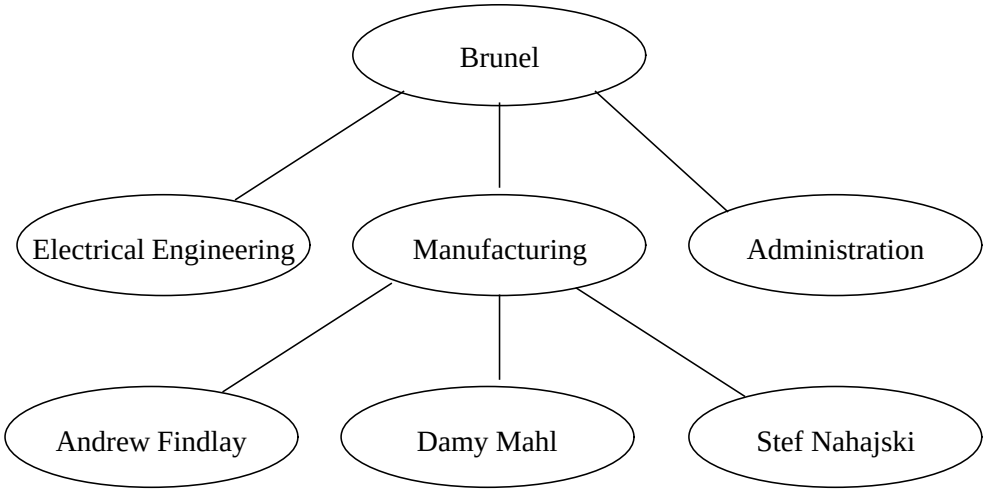

Figure 5: The DIT structure for Brunel University

So far, no attempt has been made to create entries for 'organisational roles' as there is no central database that can supply that information. People have been entered as 'organisational person' objects.

A few other object types have been used. For example, the ‘room’ type defined by the THORN[Kille1989] project is used where the telephone number for a room is known. Interestingly, the current definition of a room does not permit a ‘room number’ attribute! Presumably the room number was expected to be part of the name. Some OSI services are also registered in the Directory, including the Directory itself.

6.1. Sources of Data

There are four major sources of data for Brunel’s part of the DIT:

Staff Database
: This is derived from the University’s personnel database. Each person’s entry contains their surname, forenames, initials, title, and department.

Student Database
: This is derived from the Registry database. It contains the same information as the staff database, plus the course of study and yeargroup.

Telephone Database
: This is supplied by the University telephone exchange. It contains entries for both people and rooms. For people, surname, initials, title, department, and phone number are listed. A room number is shown for most people. Entries for rooms show the name of the room, the department that it belongs to, the room number, and the phone number.

Electronic Mail Database
: The tables that drive the electronic mail system are processed to supply surname, initials, one forename, department, class (staff or student), and RFC822 mailname.

The data from all sources is converted to a standard ‘attribute-value’ format by a collection of shell scripts. An example of this format derived from the staff database is shown in Figure 6.

```
Surname=FINDLAY
Initials=A
PersonalTitle=Mr
Department=Manf and Eng Systems Group
Status=Staff
Room=EC 21
Phone=2512

Surname=FINN
Initials=A
PersonalTitle=Mr
Department=Electrical Engineering
Status=Staff
Room=E 409
Phone=2946
```

Figure 6: Example showing the intermediate data format

6.2. Building the Directory

Brunel’s part of the DIT was constructed in two phases. In the first phase, a list of Departments was generated from the input data files and the appropriate *organizationalUnit* nodes were created. The second phase created entries for people and rooms under the departmental nodes.

Almost all the processing involved in loading the Directory is done with shell scripts. Where access to the Directory is needed, *dish* commands are used. Only two programs had to be written in ‘C’ and both are small. The rest of the work is done by *sh*, *awk*, and *sed.*

Interesting problems were posed by the variable quality of the input data. For example, the three databases supplied by the University administration all used different conventions for naming departments. Some databases were upper-case only, and there were several variations in the way initials were represented. The two 'C' programs mentioned above were used to sort out the initials and attempt to restore correct upper-case and lower-case letters in names. As the administration data came from a system with fixed-width fields, many department names were truncated so they could not be used directly. After an abortive experiment with X.500 aliases, it was decided that the *info* attribute would be hijacked to identify the departments that these 'shortnames' referred to. Thus, when each department's entry was created, the *info* attribute was set up with a list of shortnames as well as a list of *organizationalUnit* names (see Figure 7).

Attribute	Value
organizationalUnitName	Computing and Media Services
organizationalUnitName	Computer Centre
organizationalUnitName	Media Services
info	Shortname: Computing and Media Services
info	Source: DEPARTMENTS-from-Admin
info	Shortname: Computer Centre
info	Shortname: Media Services
treeStructure	room & thornPerson & thornObject & quipuNonLeafObject & quipuObject & applicationProcess & organizationalPerson & organizationalUnit & alias
objectClass	thornObject & quipuNonLeafObject & quipuObject & organizationalUnit & top

Figure 7: Example of a Department entry

Note the *objectClass* and *treeStructure* attributes. They define the type of the node, and the types of nodes that may appear beneath this one in the DIT, respectively.

The scripts that handle creation of entries for people and rooms are designed to be a general-purpose data merging tool. The same scripts are used for all data sources, working from the standard intermediate file format. Figure 8 shows the algorithm used on each entry in the input file.

Build standard fields as they will appear in the Directory

Find the correct departmental entry using the 'Shortname' fields
and make that the 'current position'.

Search this department for a node matching the surname exactly and
the full name approximately.
If more than one entry is found, report an error and stop processing this entry.
If no entry is found, create one using the available information.

Retrieve the information for the entry.

Add any fields that are supplied by the input file but are missing in the Directory entry.

If any changes were made in the previous step, write the entry back to the Directory.

Figure 8: The data-merge algorithm

Although all sources of data are handled in the same way, it is important to process the 'best' sources first. For this purpose, a good data source is defined as one which is likely to have complete and correct information for the fields it defines. Thus, a database derived from the payroll is 'better' than one derived from a UNIX password file because people are more likely to complain if their entry on the payroll is wrong! The payroll file is also more likely to have the complete name and the correct department.

The algorithm described is not very good when dealing with poor-quality data. Where the data sources are not consistent, it is possible that several directory entries get created for a single person. For example, the electronic-mail database is often wrong about the department that a user belongs to. This would cause two entries to be created for the person concerned: one containing only an e-mail address, while the full entry with phone number and postal address appear in a different department. It is difficult to be certain about how many errors are caused by this algorithm, but a rough figure can be calculated: There are 1200 people listed in the staff database, and 4300 in the student database. The Brunel DSA reports 6800 entries, so it is possible that 1300 of these are duplicates! In fact there are less duplicates, because the student database does not include most of the part-time and external students who have computer accounts and therefore e-mail addresses. Other interesting facts emerge from scanning the log of the build process: some people that are apparently members of staff are not on the payroll, and quite a few people who are neither staff nor student have e-mail addresses. Taking these factors into account it seems likely that less than 10% of the entries in the Brunel directory are duplicates. There is scope here for a tool to identify likely duplicate entries.

Several important areas of data-handling have not yet been addressed. The most significant one is the need to delete obsolete entries. A university has a very large annual turnover of people: about 25% of the students graduate each year and are replaced by freshers. Staff come and go, or even move from one department to another. Telephone numbers change, and so do personal roles. If a directory is to be respected by its users, it must be kept up-to-date. The 'proper' way to do this is to provide an X.500 interface onto the central administrative database, but it will be some time yet before this is possible. (It helps to **have** a central administrative database to start with!) In the interim, much effort and ingenuity will be needed to derive update information from multiple sources of data.

Parallel work at UCL[Barker1990] addresses some of the problems described. The UCL update system is based on comparison of the latest information from each source with the previous version. The list of differences generated is then converted into a list of updates to be performed on the Directory. The advantages of this approach are that fewer Directory operations are needed, and deletions are easier to cope with. The disadvantage is that it can be very difficult to recover from the failure of an update.

7. Use of the Directory

Within Brunel there are now a few people who use the X.500 Directory instead of the printed phone book. The main factor that discourages online information services at present is the widespread use of DOS for administrative jobs. Staff are not willing to drop out of their wordprocessor or spreadsheet package to load a directory-browsing application. It is hoped that the spread of X-terminals (and even PC windowing systems) will enable people to keep a directory application on their 'desktop' in the same way that they might keep a clock icon.

Brunel is involved in the design of Directory User Interfaces for both the X Window System and MS-Windows.[Findlay1989] In the early stages of this work, a curses-based interface called *sd* was produced. This has been made available as a public service on the UK academic network (JANET) as well as on our local UNIX machines. The public-access service has handled about 100 calls per month, mostly from people looking for data on their own institution. *sd* is the most heavily used of all directory interfaces within the University because it works on ordinary character terminals, though the use of our X-based interfaces *xd* and *pod* is increasing.

The Directory is already proving useful as a wide-area service. In the UK there is a pilot project funded by the Joint Network Team, which is aiming to get X.500 services into all universities. So far, about 10 sites have significant amounts of real data in their DSAs, and a further 10 are expected soon. The UK pilot is linked to similar exercises in other European countries, and to the US and Australian networks. In June 1990 it was estimated that there were over 200 000 entries in the DIT.

8. The future

The Directory is expanding rapidly, though at present the number of queries is small. When mail systems start to use X.500 for routing and mailing list expansion, the level of use will increase dramatically.

At Brunel, we hope to use the X.500 directory as the master source of telephone data and the printed phone book will be generated automatically from it. With the new user interfaces that we are developing, it should be possible to make departmental secretaries responsible for maintaining directory entries. This will ensure that entries are provided by those people who have the most up-to-date information.

There is still a lot of work to be done on merging data from different sources into the Directory, and then keeping track of changes. A lot of this will be specific to individual sites, but it is hoped that general methods can be developed for common operations.

References

Barker1990.
Paul Barker, *Managing Data Derived from Multiple Sources in an X.500 Directory*, UCL, London, June 1990. UCL Computer Science Research Note RN/90/6

Findlay1989.
Andrew Findlay and Damanjit Mahl, ‘‘Designing an X.500 User Interface: The Early Stages,’’ *Proceedings of the UKUUG and UKnet Winter Technical Meeting*, UKUUG, Cardiff, December 1989.

ISO/CCITT1988.
ISO/CCITT, *Recommendation X.500: The Directory - Overview of Concepts, Models and Services*, Geneva, March 1988. ISO 9594 is technically aligned with X.500

Kille1988.
Steve Kille, *The Design of QUIPU*, UCL, London, July 1988. UCL Research Note RN/88/32

Kille1989.
Steve Kille, *The THORN and RARE X.500 Naming Architecture*, UCL, London, April 1989. UCL-64

Rose1990.
Marshall T Rose, *The ISO Development Environment: Users Manual*, Performance Systems International, Inc., Mountain View, CA., January 1990. Version 6.0