Network Working Group
Request for Comments: 1274

P. Barker
S. Kille
University College London

November 1991

The COSINE and Internet X.500 Schema

Status of this Memo

This RFC specifies an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

This document suggests an X.500 Directory Schema, or Naming Architecture, for use in the COSINE and Internet X.500 pilots. The schema is independent of any specific implementation. As well as indicating support for the standard object classes and attributes, a large number of generally useful object classes and attributes are also defined. An appendix to this document includes a machine processable version of the schema.

This document also proposes a mechanism for allowing the schema to evolve in line with emerging requirements. Proformas to support this process are included.

Corrections and additions to the schema should be sent to na-update@cs.ucl.ac.uk list, as described within.

1. Introduction

Directory Services are a fundamental requirement of both human and computer communications' systems. Human users need to be able to look up various details about other people: for example, telephone numbers, facsimile numbers and paper mail addresses. Computing systems also need Directory Services for several purposes: for example, to support address look-ups for a variety of services, and to support user-friendly naming and distribution lists in electronic mail systems.

Directory Services have recently been standardised and published as the 1988 CCITT X.500 / ISO IS9594 recommendations [1]. The standard provides a good basis for the provision of real services, and a considerable amount of Directory Service piloting activity is

currently underway. In the U.S., the PSI White Pages Pilot [4] has stimulated use of X.500 on the Internet. In Britain, the U.K. Academic Community Directory Pilot [5] is similarly promoting use of X.500.

2. Motivation and aims of this document

In a number of areas the X.500 standard only provides a basis for services. One such area is the Directory's Schema or Naming Architecture. The standard defines a number of useful object classes, in X.521, and attribute types, in X.520. These are intended to be generally useful across a range of directory applications. However, while these standard definitions are a useful starting point, they are insufficient as a basis for a large scale pilot directory.

While it is possible for directory administrators to define their own sets of additional attribute types and object classes, this is undesirable for some common attributes and objects. The same objects and attribute types would be privately defined many times over. This would result in the directory's generality being diminished as remote systems would be unable to determine the semantics of these privately defined data types.

A number of useful additions to the standard definitions were made in this note's forerunner, "The THORN and RARE Naming Architecture" [2]. These have been heavily used in early X.500 piloting activities. Furthermore, both the THORN and Quipu X.500 implementations have made use of these definitions.

Since the afore-mentioned note was issued, a number of further requirements have come to light as the volume and variety of piloting activity has increased. Yet further requirements seem likely as the scale of X.500 pilot services increases. Thus, it is argued that it is not sufficient to merely reissue an updated version of the original note. The schema is a "living document" that needs procedures for:

- Allowing submission of requests for new attributes and object classes to be added into the schema;

- Allowing groups of object classes and attribute types defined elsewhere to be integrated into the schema.

- Checking for the redundancy of any previously defined attribute types and object classes.

This document attempts to establish procedures to allow for the

continual updating of the schema. Two proformas are set out for this purpose. In addition, descriptive detail is provided for the additional object classes and attribute types defined in the schema. These descriptions follow the style used in X.520 and X.521. Finally, also following the style adopted in the standards documents, appendices will include the entire schema. Plain text versions of the document's appendices are intended to be machine processable to allow derivation of a system's schema tables. Appendix C lists all the schema's object classes and attribute types in their respective ASN.1 macro formats.

The scope and intended remit of this coordination activity should be clearly understood.

- Esoteric and local, highly experimental requirements should continue to be met by private definitions.

- Requirements which have support from more than one site will usually be integrated into the schema. Put in other words, the tendency will be for the inclusion, as opposed to the exclusion, of useful additions to the schema.

- An attempt will be made to avoid duplication of object classes and attribute types for essentially similar real world objects.

3. What conformance to this schema means

It is not reasonable to require that a DSA which supports this schema has specific code to handle each of the defined syntaxes. However, the following requirements are made of a system which claims conformance to this specification:

1. A DSA shall be able to store all of the attributes and object class values specified. (Note that this implies support for all the object classes and attribute types required by strong authentication as defined in X.509.)

2. A DUA shall be able to identify each attribute type and object class to the user, with an appropriate representation (e.g., a string).

3. These statement are qualified for large attributes values (>1kbyte). A conforming DSA does not have to store such attribute values, and a DUA does not have to display such values, although it must indicate their presence.

The following are desirable, but not required:

1. For a DSA to match correctly on the basis of all attribute syntaxes defined

2. For a DSA to enforce the Object Class schema implied by these definitions

3. For a DUA to correctly display the attribute values (syntaxes) defined

4. For DUAs and DSAs to maintain compatibility with a previous version of the schema.

4. Requesting new object classes and attribute types

This section defines procedures for requesting new object classes and attribute types to be added to the schema. Proformas for object classes and attribute types are specified, and examples given of how to use them. A mechanism for making requests for large groups of new object classes and attribute types is described in the next section.

As stated earlier, it is anticipated that the schema will evolve considerably over time. As X.500 is used to support a widening range of applications, there will be requirements for extensions to the schema. This document proposes formalising this procedure by requiring requests for additions to the schema to be submitted as completed proformas. This stipulation will greatly simplify subsequent revisions of the schema.

There is one qualification to the above with respect to requests for modifications to an existing object class. If a modification to an object class merely involves additional, optional attributes, the object class will be enhanced as requested. Systems are expected to be resilient to such changes to the schema. However, requests to modify an object class, such that the mandatory attribute types require altering, will not be met. Instead, a new object class will be created, and the original object class expired following the scheme described in the next main section.

It is anticipated that most requests for modifications to the schema will be met without any need for editorial intervention. Sometimes, however, some discussion between the submitter of a request and the schema's editor may be required. For example, the editor may have to judge the relative merits of two very similar requests and, as a result, one of the parties may not get quite what they want. In cases such as this where the submitter of a request feels aggrieved about an editorial decision, the requestor may appeal to a broader community by explaining their views to the mailing list osi-ds@cs.ucl.ac.uk. Heed will be paid to any consensus that emerges

from discussions on the schema on this list. If it proves that this list is used almost solely for discussions on schema issues, a separate discussion list will be created.

To facilitate the production of the afore-mentioned proformas, tools are included in Appendix B which will verify that a proforma has been correctly formatted.

Completed proformas should be mailed to na-update@cs.ucl.ac.uk

4.1. Object Class proforma

This section gives an example, completed proforma for a new object class, alcoholic drink. A proforma for object class specified in BNF is included in Appendix A.

```
  Object Class: Alcoholic Drink

  Description: The Alcoholic Drink object class is used to define
  entries representing intoxicating beverages.

  ASN1OCMacro: alcoholicDrink OBJECT-CLASS
      SUBCLASS OF drink
      MUST CONTAIN {
          percentAlcohol}
      MAY CONTAIN {
          normalServing,
          hue}
```

An object class description consists of three fields, separated by blank lines. The keywords Object Class, Description and ASN1OCMacro, and their suffixed colons, must be included exactly as above.

The Object Class field should be used for a textual description of the object class. This will be at most three or four words.

The Description field should contain some explanatory text about the intended use of the object class. This can run to a number of lines.

The ASN1OCMacro field should follow the definition of the object class macro as specified in X.501. The above example shows the main features. There are many more examples which can studied in the section defining the Pilot Object Classes.

4.2. Attribute type proforma

This section gives an example completed proforma for a new attribute type, hue (one of the attribute types in the alcoholic drink object

class).

```
  Attribute Type: Hue

  Description: The Hue attribute type specifies the hue of
  an object.  (Note that a description may run to several
  lines.)

  OCMust:

  OCMay: alcoholicDrink

  ASN1ATMacro:hue ATTRIBUTE
      WITH ATTRIBUTE SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-hue))

  ub-hue INTEGER ::= 256
```

An attribute type description consists of five fields, separated by blank lines. The keywords Attribute Type, Description, OCMust, OCMay and ASN1ATMacro, and their suffixed colons, must be included exactly as above.

The Attribute Type field should be used for a textual description of the attribute type. This will be at most three or four words.

The Description field should contain some explanatory text about the intended use of the attribute type. This can run to a number of lines.

The OCMust field should contain a comma-separated list of object classes for which this attribute is mandatory.

The OCMay field should contain a comma-separated list of object classes for which this attribute is optional.

The ASN1ATMacro field should follow the definition of the attribute macro as specified in X.501. The above example shows some of the features. In particular, please note the format for specifying size constraints.

5. Integrating groups of object classes and attribute types.

This section describes two mechanisms that may be employed to allow the integration of a substantial number of new object classes and attribute types into the schema.

The first mechanism allows for the transition of groups of related, privately defined object classes and attribute types into the schema. An example of when such a transition might be appropriate is when some experimental use of the Directory is widely adopted within the pilot. Such a transition will be made if the following conditions hold:

- The definitions are well structured: i.e., they are not scattered over a multiplicity of object identifier subtrees.

- The definitions are in use at a number of sites, and having to adopt new object identifiers would be unnecessarily disruptive.

A second mechanism allows for the allocation of an object subtree for a group of new definitions. A pilotGroups object identifier has been defined for this purpose. This method will be suitable for an experiment requiring a considerable number of new object identifiers to be defined. This approach allows for flexibility during experimentation and should simplify both the management and the coherence of the pilot's object identifiers.

In both cases, the object classes, attribute types and syntaxes should be defined and described in an RFC. It is suggested that such documents should follow the style used in this document for object class and attribute type definitions. A reference will be given in this schema to the document containing the definitions.

6. Removing "old" object classes and attribute types.

It is also important that object classes and attribute types which are no longer used or useful are removed from the schema. Some object classes and attribute types initially defined as pilot extensions may be included as standard definitions in future versions of the standard. In such a case, it is important that there should be a fairly rapid transition to the standard definitions. Another possibility is that newer, more specific definitions obviate the original definitions.

Two things are essential. First, it is crucial that "old" definitions are retired as gracefully as possible. The intention to retire a definition will be sent to the osi-ds@cs.ucl.ac.uk mail list. In the absence of objections, the definition will be marked for expiry with a given expiry date. The definition will remain in the schema until the expiry date. Users of the schema should ensure that they make the transition to new, alternative definitions in the interim.

Second, users of the schema must have the right to argue for the retention of definitions which they regard as necessary, there being no other definitions which closely meet their requirements. It is clearly impossible to lay down hard and fast rules on this point, as no two instances will ever be quite the same. It is intended that the refereeing on these matters will be sympathetic! As for requests for additions, an aggrieved user can "go to arbitration" by initiating a discussion on the osi-ds@cs.ucl.ac.uk mail list.

7. Object Identifiers

Some additional object identifiers are defined for this schema. These are also reproduced in Appendix C.

```
data OBJECT IDENTIFIER ::= {ccitt 9}
pss OBJECT IDENTIFIER ::= {data 2342}
ucl OBJECT IDENTIFIER ::= {pss 19200300}
pilot OBJECT IDENTIFIER ::= {ucl 100}

pilotAttributeType OBJECT IDENTIFIER ::= {pilot 1}
pilotAttributeSyntax OBJECT IDENTIFIER ::= {pilot 3}
pilotObjectClass OBJECT IDENTIFIER ::= {pilot 4}
pilotGroups OBJECT IDENTIFIER ::= {pilot 10}

iA5StringSyntax OBJECT IDENTIFIER ::= {pilotAttributeSyntax 4}
caseIgnoreIA5StringSyntax OBJECT IDENTIFIER ::=
                                        {pilotAttributeSyntax 5}
```

8. Object Classes

8.1. X.500 standard object classes

A number of generally useful object classes are defined in X.521, and these are supported. Refer to that document for descriptions of the suggested usage of these object classes. The ASN.1 for these object classes is reproduced for completeness in Appendix C.

8.2. X.400 standard object classes

A number of object classes defined in X.400 are supported. Refer to X.402 for descriptions of the usage of these object classes. The ASN.1 for these object classes is reproduced for completeness in Appendix C.

8.3. COSINE/Internet object classes

This section attempts to fuse together the object classes designed for use in the COSINE and Internet pilot activities. Descriptions

are given of the suggested usage of these object classes. The ASN.1 for these object classes is also reproduced in Appendix C.

8.3.1. Pilot Object

The PilotObject object class is used as a sub-class to allow some common, useful attributes to be assigned to entries of all other object classes.

```
  pilotObject OBJECT-CLASS
      SUBCLASS OF top
      MAY CONTAIN {
          info,
          photo,
          manager,
          uniqueIdentifier,
          lastModifiedTime,
          lastModifiedBy,
          dITRedirect,
          audio}
  ::= {pilotObjectClass 3}
```

8.3.2. Pilot Person

The PilotPerson object class is used as a sub-class of person, to allow the use of a number of additional attributes to be assigned to entries of object class person.

```
  pilotPerson OBJECT-CLASS
      SUBCLASS OF person
      MAY CONTAIN {
                  userid,
                  textEncodedORAddress,
                  rfc822Mailbox,
                  favouriteDrink,
                  roomNumber,
                  userClass,
                  homeTelephoneNumber,
                  homePostalAddress,
                  secretary,
                  personalTitle,
                  preferredDeliveryMethod,
                  businessCategory,
                  janetMailbox,
                  otherMailbox,
                  mobileTelephoneNumber,
                  pagerTelephoneNumber,
                  organizationalStatus,
```

```
                    mailPreferenceOption,
                    personalSignature}
    ::= {pilotObjectClass 4}
```

8.3.3. Account

The Account object class is used to define entries representing computer accounts. The userid attribute should be used for naming entries of this object class.

```
  account OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
          userid}
      MAY CONTAIN {
          description,
          seeAlso,
          localityName,
          organizationName,
          organizationalUnitName,
          host}
  ::= {pilotObjectClass 5}
```

8.3.4. Document

The Document object class is used to define entries which represent documents.

```
  document OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
          documentIdentifier}
      MAY CONTAIN {
          commonName,
          description,
          seeAlso,
          localityName,
          organizationName,
          organizationalUnitName,
          documentTitle,
          documentVersion,
          documentAuthor,
          documentLocation,
          documentPublisher}
  ::= {pilotObjectClass 6}
```

8.3.5. Room

The Room object class is used to define entries representing rooms. The commonName attribute should be used for naming pentries of this object class.

```
  room OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
          commonName}
      MAY CONTAIN {
          roomNumber,
          description,
          seeAlso,
          telephoneNumber}
  ::= {pilotObjectClass 7}
```

8.3.6. Document Series

The Document Series object class is used to define an entry which represents a series of documents (e.g., The Request For Comments papers).

```
  documentSeries OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
          commonName}
      MAY CONTAIN {
          description,
          seeAlso,
          telephoneNumber,
          localityName,
          organizationName,
          organizationalUnitName}
  ::= {pilotObjectClass 9}
```

8.3.7. Domain

The Domain object class is used to define entries which represent DNS or NRS domains. The domainComponent attribute should be used for naming entries of this object class. The usage of this object class is described in more detail in [3].

```
  domain OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
          domainComponent}
      MAY CONTAIN {
```

```
            associatedName,
            organizationName,
            organizationalAttributeSet}
    ::= {pilotObjectClass 13}
```

8.3.8. RFC822 Local Part

The RFC822 Local Part object class is used to define entries which represent the local part of RFC822 mail addresses. This treats this part of an RFC822 address as a domain. The usage of this object class is described in more detail in [3].

```
  rFC822localPart OBJECT-CLASS
      SUBCLASS OF domain
      MAY CONTAIN {
          commonName,
          surname,
          description,
          seeAlso,
          telephoneNumber,
          postalAttributeSet,
          telecommunicationAttributeSet}
    ::= {pilotObjectClass 14}
```

8.3.9. DNS Domain

The DNS Domain (Domain NameServer) object class is used to define entries for DNS domains. The usage of this object class is described in more detail in [3].

```
  dNSDomain OBJECT-CLASS
      SUBCLASS OF domain
      MAY CONTAIN {
          ARecord,
          MDRecord,
          MXRecord,
          NSRecord,
          SOARecord,
          CNAMERecord}
    ::= {pilotObjectClass 15}
```

8.3.10. Domain Related Object

The Domain Related Object object class is used to define entries which represent DNS/NRS domains which are "equivalent" to an X.500 domain: e.g., an organisation or organisational unit. The usage of this object class is described in more detail in [3].

```
domainRelatedObject OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        associatedDomain}
::= {pilotObjectClass 17}
```

8.3.11. Friendly Country

The Friendly Country object class is used to define country entries in the DIT. The object class is used to allow friendlier naming of countries than that allowed by the object class country. The naming attribute of object class country, countryName, has to be a 2 letter string defined in ISO 3166.

```
friendlyCountry OBJECT-CLASS
    SUBCLASS OF country
    MUST CONTAIN {
        friendlyCountryName}
::= {pilotObjectClass 18}
```

8.3.12. Simple Security Object

The Simple Security Object object class is used to allow an entry to have a userPassword attribute when an entry's principal object classes do not allow userPassword as an attribute type.

```
simpleSecurityObject OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        userPassword }
::= {pilotObjectClass 19}
```

8.3.13. Pilot Organization

The PilotOrganization object class is used as a sub-class of organization and organizationalUnit to allow a number of additional attributes to be assigned to entries of object classes organization and organizationalUnit.

```
pilotOrganization OBJECT-CLASS
    SUBCLASS OF organization, organizationalUnit
    MAY CONTAIN {
                buildingName}
::= {pilotObjectClass 20}
```

8.3.14. Pilot DSA

The PilotDSA object class is used as a sub-class of the dsa object class to allow additional attributes to be assigned to entries for DSAs.

```
  pilotDSA OBJECT-CLASS
      SUBCLASS OF dsa
      MUST CONTAIN {
          dSAQuality}
  ::= {pilotObjectClass 21}
```

8.3.15. Quality Labelled Data

The Quality Labelled Data object class is used to allow the assignment of the data quality attributes to subtrees in the DIT.

See [8] for more details.

```
  qualityLabelledData OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
          dSAQuality}
      MAY CONTAIN {
          subtreeMinimumQuality,
          subtreeMaximumQuality}
  ::= {pilotObjectClass 22}
```

9. Attribute Types

9.1. X.500 standard attribute types

A number of generally useful attribute types are defined in X.520, and these are supported. Refer to that document for descriptions of the suggested usage of these attribute types. The ASN.1 for these attribute types is reproduced for completeness in Appendix C.

9.2. X.400 standard attribute types

The standard X.400 attribute types are supported. See X.402 for full details. The ASN.1 for these attribute types is reproduced in Appendix C.

9.3. COSINE/Internet attribute types

This section describes all the attribute types defined for use in the COSINE and Internet pilots. Descriptions are given as to the suggested usage of these attribute types. The ASN.1 for these

attribute types is reproduced in Appendix C.

9.3.1. Userid

The Userid attribute type specifies a computer system login name.

```
  userid ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-user-identifier))
  ::= {pilotAttributeType 1}
```

9.3.2. Text Encoded O/R Address

The Text Encoded O/R Address attribute type specifies a text encoding of an X.400 O/R address, as specified in RFC 987. The use of this attribute is deprecated as the attribute is intended for interim use only. This attribute will be the first candidate for the attribute expiry mechanisms!

```
  textEncodedORAddress ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
      (SIZE (1 .. ub-text-encoded-or-address))
  ::= {pilotAttributeType 2}
```

9.3.3. RFC 822 Mailbox

The RFC822 Mailbox attribute type specifies an electronic mailbox attribute following the syntax specified in RFC 822. Note that this attribute should not be used for greybook or other non-Internet order mailboxes.

```
  rfc822Mailbox ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreIA5StringSyntax
          (SIZE (1 .. ub-rfc822-mailbox))
  ::= {pilotAttributeType 3}
```

9.3.4. Information

The Information attribute type specifies any general information pertinent to an object. It is recommended that specific usage of this attribute type is avoided, and that specific requirements are met by other (possibly additional) attribute types.

```
  info ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
```

```
            caseIgnoreStringSyntax
            (SIZE (1 .. ub-information))
    ::= {pilotAttributeType 4}
```

9.3.5. Favourite Drink

The Favourite Drink attribute type specifies the favourite drink of an object (or person).

```
  favouriteDrink ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-favourite-drink))
  ::= {pilotAttributeType 5}
```

9.3.6. Room Number

The Room Number attribute type specifies the room number of an object. Note that the commonName attribute should be used for naming room objects.

```
  roomNumber ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-room-number))
  ::= {pilotAttributeType 6}
```

9.3.7. Photo

The Photo attribute type specifies a "photograph" for an object. This should be encoded in G3 fax as explained in recommendation T.4, with an ASN.1 wrapper to make it compatible with an X.400 BodyPart as defined in X.420.

```
  IMPORT G3FacsimileBodyPart FROM { mhs-motis ipms modules
  information-objects }

  photo ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          CHOICE {
              g3-facsimile [3] G3FacsimileBodyPart
              }
      (SIZE (1 .. ub-photo))
  ::= {pilotAttributeType 7}
```

9.3.8. User Class

The User Class attribute type specifies a category of computer user. The semantics placed on this attribute are for local interpretation. Examples of current usage od this attribute in academia are undergraduate student, researcher, lecturer, etc. Note that the organizationalStatus attribute may now often be preferred as it makes no distinction between computer users and others.

```
  userClass ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-user-class))
  ::= {pilotAttributeType 8}
```

9.3.9. Host

The Host attribute type specifies a host computer.

```
  host ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-host))
  ::= {pilotAttributeType 9}
```

9.3.10. Manager

The Manager attribute type specifies the manager of an object represented by an entry.

```
  manager ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          distinguishedNameSyntax
  ::= {pilotAttributeType 10}
```

9.3.11. Document Identifier

The Document Identifier attribute type specifies a unique identifier for a document.

```
  documentIdentifier ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-document-identifier))
  ::= {pilotAttributeType 11}
```

9.3.12. Document Title

The Document Title attribute type specifies the title of a document.

```
  documentTitle ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
      (SIZE (1 .. ub-document-title))
  ::= {pilotAttributeType 12}
```

9.3.13. Document Version

The Document Version attribute type specifies the version number of a document.

```
  documentVersion ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-document-version))
  ::= {pilotAttributeType 13}
```

9.3.14. Document Author

The Document Author attribute type specifies the distinguished name of the author of a document.

```
  documentAuthor ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          distinguishedNameSyntax
  ::= {pilotAttributeType 14}
```

9.3.15. Document Location

The Document Location attribute type specifies the location of the document original.

```
  documentLocation ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-document-location))
  ::= {pilotAttributeType 15}
```

9.3.16. Home Telephone Number

The Home Telephone Number attribute type specifies a home telephone number associated with a person. Attribute values should follow the agreed format for international telephone numbers: i.e., "+44 71 123 4567".

```
    homeTelephoneNumber ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            telephoneNumberSyntax
    ::= {pilotAttributeType 20}
```

9.3.17. Secretary

The Secretary attribute type specifies the secretary of a person. The attribute value for Secretary is a distinguished name.

```
    secretary ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            distinguishedNameSyntax
    ::= {pilotAttributeType 21}
```

9.3.18. Other Mailbox

The Other Mailbox attribute type specifies values for electronic mailbox types other than X.400 and rfc822.

```
    otherMailbox ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            SEQUENCE {
                    mailboxType PrintableString, -- e.g. Telemail
                    mailbox IA5String  -- e.g. X378:Joe
            }
    ::= {pilotAttributeType 22}
```

9.3.19. Last Modified Time

The Last Modified Time attribute type specifies the last time, in UTC time, that an entry was modified. Ideally, this attribute should be maintained by the DSA.

```
    lastModifiedTime ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            uTCTimeSyntax
    ::= {pilotAttributeType 23}
```

9.3.20. Last Modified By

The Last Modified By attribute specifies the distinguished name of the last user to modify the associated entry. Ideally, this attribute should be maintained by the DSA.

```
    lastModifiedBy ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            distinguishedNameSyntax
```

```
    ::= {pilotAttributeType 24}
```

9.3.21. Domain Component

The Domain Component attribute type specifies a DNS/NRS domain. For example, "uk" or "ac".

```
  domainComponent ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreIA5StringSyntax
          SINGLE VALUE
  ::= {pilotAttributeType 25}
```

9.3.22. DNS ARecord

The A Record attribute type specifies a type A (Address) DNS resource record [6] [7].

```
  aRecord ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          DNSRecordSyntax
  ::= {pilotAttributeType 26}
```

9.3.23. MX Record

The MX Record attribute type specifies a type MX (Mail Exchange) DNS resource record [6] [7].

```
  mXRecord ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          DNSRecordSyntax
  ::= {pilotAttributeType 28}
```

9.3.24. NS Record

The NS Record attribute type specifies an NS (Name Server) DNS resource record [6] [7].

```
  nSRecord ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          DNSRecordSyntax
  ::= {pilotAttributeType 29}
```

9.3.25. SOA Record

The SOA Record attribute type specifies a type SOA (Start of Authority) DNS resorce record [6] [7].

```
    sOARecord ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            DNSRecordSyntax
    ::= {pilotAttributeType 30}
```

9.3.26. CNAME Record

The CNAME Record attribute type specifies a type CNAME (Canonical Name) DNS resource record [6] [7].

```
    cNAMERecord ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            iA5StringSyntax
    ::= {pilotAttributeType 31}
```

9.3.27. Associated Domain

The Associated Domain attribute type specifies a DNS or NRS domain which is associated with an object in the DIT. For example, the entry in the DIT with a distinguished name "C=GB, O=University College London" would have an associated domain of "UCL.AC.UK. Note that all domains should be represented in rfc822 order. See [3] for more details of usage of this attribute.

```
    associatedDomain ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            caseIgnoreIA5StringSyntax
    ::= {pilotAttributeType 37}
```

9.3.28. Associated Name

The Associated Name attribute type specifies an entry in the organisational DIT associated with a DNS/NRS domain. See [3] for more details of usage of this attribute.

```
    associatedName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            distinguishedNameSyntax
    ::= {pilotAttributeType 38}
```

9.3.29. Home postal address

The Home postal address attribute type specifies a home postal address for an object. This should be limited to up to 6 lines of 30 characters each.

```
    homePostalAddress ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
```

```
            postalAddress
            MATCHES FOR EQUALITY
    ::= {pilotAttributeType 39}
```

9.3.30. Personal Title

The Personal Title attribute type specifies a personal title for a person. Examples of personal titles are "Ms", "Dr", "Prof" and "Rev".

```
  personalTitle ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-personal-title))
  ::= {pilotAttributeType 40}
```

9.3.31. Mobile Telephone Number

The Mobile Telephone Number attribute type specifies a mobile telephone number associated with a person. Attribute values should follow the agreed format for international telephone numbers: i.e., "+44 71 123 4567".

```
  mobileTelephoneNumber ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          telephoneNumberSyntax
  ::= {pilotAttributeType 41}
```

9.3.32. Pager Telephone Number

The Pager Telephone Number attribute type specifies a pager telephone number for an object. Attribute values should follow the agreed format for international telephone numbers: i.e., "+44 71 123 4567".

```
  pagerTelephoneNumber ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          telephoneNumberSyntax
  ::= {pilotAttributeType 42}
```

9.3.33. Friendly Country Name

The Friendly Country Name attribute type specifies names of countries in human readable format. The standard attribute country name must be one of the two-letter codes defined in ISO 3166.

```
  friendlyCountryName ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
  ::= {pilotAttributeType 43}
```

9.3.34. Unique Identifier

The Unique Identifier attribute type specifies a "unique identifier" for an object represented in the Directory. The domain within which the identifier is unique, and the exact semantics of the identifier, are for local definition. For a person, this might be an institution-wide payroll number. For an organisational unit, it might be a department code.

```
  uniqueIdentifier ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-unique-identifier))
  ::= {pilotAttributeType 44}
```

9.3.35. Organisational Status

The Organisational Status attribute type specifies a category by which a person is often referred to in an organisation. Examples of usage in academia might include undergraduate student, researcher, lecturer, etc.

A Directory administrator should probably consider carefully the distinctions between this and the title and userClass attributes.

```
  organizationalStatus ATTRIBUTE
          WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-organizational-status))
  ::= {pilotAttributeType 45}
```

9.3.36. Janet Mailbox

The Janet Mailbox attribute type specifies an electronic mailbox attribute following the syntax specified in the Grey Book of the Coloured Book series. This attribute is intended for the convenience of U.K users unfamiliar with rfc822 and little-endian mail addresses. Entries using this attribute MUST also include an rfc822Mailbox attribute.

```
  janetMailbox ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreIA5StringSyntax
          (SIZE (1 .. ub-janet-mailbox))
  ::= {pilotAttributeType 46}
```

9.3.37. Mail Preference Option

An attribute to allow users to indicate a preference for inclusion of their names on mailing lists (electronic or physical). The absence of such an attribute should be interpreted as if the attribute was present with value "no-list-inclusion". This attribute should be interpreted by anyone using the directory to derive mailing lists, and its value respected.

```
  mailPreferenceOption ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX ENUMERATED {
              no-list-inclusion(0),
              any-list-inclusion(1),  -- may be added to any lists
              professional-list-inclusion(2)
                                      -- may be added to lists
                                      -- which the list provider
                                      -- views as related to the
                                      -- users professional inter-
                                      -- ests, perhaps evaluated
                                      -- from the business of the
                                      -- organisation or keywords
                                      -- in the entry.
              }
  ::= {pilotAttributeType 47}
```

9.3.38. Building Name

The Building Name attribute type specifies the name of the building where an organisation or organisational unit is based.

```
  buildingName ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          caseIgnoreStringSyntax
          (SIZE (1 .. ub-building-name))
  ::= {pilotAttributeType 48}
```

9.3.39. DSA Quality

The DSA Quality attribute type specifies the purported quality of a DSA. It allows a DSA manager to indicate the expected level of availability of the DSA. See [8] for details of the syntax.

```
  dSAQuality ATTRIBUTE
          WITH ATTRIBUTE-SYNTAX DSAQualitySyntax
          SINGLE VALUE
  ::= {pilotAttributeType 49}
```

9.3.40. Single Level Quality

The Single Level Quality attribute type specifies the purported data quality at the level immediately below in the DIT. See [8] for details of the syntax.

```
singleLevelQuality ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX DataQualitySyntax
        SINGLE VALUE
::= {pilotAttributeType 50}
```

9.3.41. Subtree Minimum Quality

The Subtree Minimum Quality attribute type specifies the purported minimum data quality for a DIT subtree. See [8] for more discussion and details of the syntax.

```
subtreeMinimumQuality ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX DataQualitySyntax
        SINGLE VALUE
           -- Defaults to singleLevelQuality
::= {pilotAttributeType 51}
```

9.3.42. Subtree Maximum Quality

The Subtree Maximum Quality attribute type specifies the purported maximum data quality for a DIT subtree. See [8] for more discussion and details of the syntax.

```
subtreeMaximumQuality ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX DataQualitySyntax
        SINGLE VALUE
           -- Defaults to singleLevelQuality
::= {pilotAttributeType 52}
```

9.3.43. Personal Signature

The Personal Signature attribute type allows for a representation of a person's signature. This should be encoded in G3 fax as explained in recommendation T.4, with an ASN.1 wrapper to make it compatible with an X.400 BodyPart as defined in X.420.

```
IMPORT G3FacsimileBodyPart FROM { mhs-motis ipms modules
information-objects }

personalSignature ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        CHOICE {
```

```
                g3-facsimile [3] G3FacsimileBodyPart
                }
        (SIZE (1 .. ub-personal-signature))
    ::= {pilotAttributeType 53}
```

9.3.44. DIT Redirect

The DIT Redirect attribute type is used to indicate that the object described by one entry now has a newer entry in the DIT. The entry containing the redirection attribute should be expired after a suitable grace period. This attribute may be used when an individual changes his/her place of work, and thus acquires a new organisational DN.

```
  dITRedirect ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          distinguishedNameSyntax
  ::= {pilotAttributeType 54}
```

9.3.45. Audio

The Audio attribute type allows the storing of sounds in the Directory. The attribute uses a u-law encoded sound file as used by the "play" utility on a Sun 4. This is an interim format.

```
  audio ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
          Audio
      (SIZE (1 .. ub-audio))
  ::= {pilotAttributeType 55}
```

9.3.46. Publisher of Document

The Publisher of Document attribute is the person and/or organization that published a document.

```
  documentPublisher ATTRIBUTE
          WITH ATTRIBUTE SYNTAX caseIgnoreStringSyntax
  ::= {pilotAttributeType 56}
```

9.4. Generally useful syntaxes

```
  caseIgnoreIA5StringSyntax ATTRIBUTE-SYNTAX
          IA5String
          MATCHES FOR EQUALITY SUBSTRINGS
```

```
iA5StringSyntax ATTRIBUTE-SYNTAX
    IA5String
    MATCHES FOR EQUALITY SUBSTRINGS


-- Syntaxes to support the DNS attributes

DNSRecordSyntax ATTRIBUTE-SYNTAX
        IA5String
        MATCHES FOR EQUALITY


NRSInformationSyntax ATTRIBUTE-SYNTAX
        NRSInformation
        MATCHES FOR EQUALITY


NRSInformation ::=  SET {
                [0] Context,
                [1] Address-space-id,
                routes [2] SEQUENCE OF SEQUENCE {
                Route-cost,
                Addressing-info }
        }
```

9.5. Upper bounds on length of attribute values

```
ub-document-identifier INTEGER ::= 256

ub-document-location INTEGER ::= 256

ub-document-title INTEGER ::= 256

ub-document-version INTEGER ::= 256

ub-favourite-drink INTEGER ::= 256

ub-host INTEGER ::= 256

ub-information INTEGER ::= 2048

ub-unique-identifier INTEGER ::= 256

ub-personal-title INTEGER ::= 256

ub-photo INTEGER ::= 250000
```

```
ub-rfc822-mailbox INTEGER ::= 256

ub-room-number INTEGER ::= 256

ub-text-or-address INTEGER ::= 256

ub-user-class INTEGER ::= 256

ub-user-identifier INTEGER ::= 256

ub-organizational-status INTEGER ::= 256

ub-janet-mailbox INTEGER ::= 256

ub-building-name INTEGER ::= 256

ub-personal-signature ::= 50000

ub-audio INTEGER ::= 250000
```

References

[1] CCITT/ISO, "X.500, The Directory - overview of concepts, models and services, CCITT /ISO IS 9594.

[2] Kille, S., "The THORN and RARE X.500 Naming Architecture, in University College London, Department of Computer Science Research Note 89/48, May 1989.

[3] Kille, S., "X.500 and Domains", RFC 1279, University College London, November 1991.

[4] Rose, M., "PSI/NYSERNet White Pages Pilot Project: Status Report", Technical Report 90-09-10-1, published by NYSERNet Inc, 1990.

[5] Craigie, J., "UK Academic Community Directory Service Pilot Project, pp. 305-310 in Computer Networks and ISDN Systems 17 (1989), published by North Holland.

[6] Mockapetris, P., "Domain Names - Concepts and Facilities", RFC 1034, USC/Information Sciences Institute, November 1987.

[7] Mockapetris, P., "Domain Names - Implementation and Specification, RFC 1035, USC/Information Sciences Institute, November 1987.

[8] Kille, S., "Handling QOS (Quality of service) in the

Directory," publication in process, March 1991.

APPENDIX A - Object Class and Attribute Type proformas

These are specified in BNF. First some useful definitions, common to both proformas.

```
EOL ::= -- the end of line character(s)

BlankLine ::= -- a line consisting solely of an EOL character

String ::= <anychar> | <String> <anychar>

anychar ::= --any character occurring in general text, excluding
            -- the end of line character

lString ::= <lowercase> <otherstring>

lowercase ::= [a-z]

UString ::= <uppercase> <otherstring>

uppercase ::= [A-Z]

otherstring ::= <otherchar> | <otherstring> <otherchar>

otherchar ::= <lowercase> | <uppercase> | <digit>

Integer ::= <digit> | <Integer> <digit>

digit ::= [0-9]
```

1. Object Class

```
OCProforma ::= <ObjectClassName> <BlankLine> <Description> \
               <BlankLine> <OCMacro>

ObjectClassName ::= "ObjectClass:" <String> <EOL>

Description ::= "Description:" <DescriptiveText> <EOL>

DescriptiveText ::= <String> | <DescriptiveText> <EOL> <String>

OCMacro ::= "ASN1OCMacro:" <ObjectClassMacro>

-- The definition of ObjectClassMacro is adapted from
-- that given in X.501
```

```
ObjectClassMacro ::= <OCname> "OBJECT-CLASS" <SubclassOf> \
                     <MandatoryAttributes> <OptionalAttributes>

OCName ::= <lString>

SubclassOf ::= "SUBCLASS OF" Subclasses | <empty>

Subclasses ::= <Subclass> | <Subclass> "," <Subclasses>

Subclass ::= <OCName>

MandatoryAttributes ::= "MUST CONTAIN {" <Attributes> "}" \
                        | <empty>
OptionalAttributes ::= "MAY CONTAIN {" <Attributes> "}" | <empty>

Attributes ::= <AttributeTerm> | <AttributeTerm> "," <Attributes>

AttributeTerm ::= <Attribute> | <AttributeSet>

Attribute ::= <lString>

AttributeSet ::= <lString>
```

2. Attribute Type

```
ATProforma ::= <AttributeTypeName> <BlankLine> <Description> \
               <BlankLine> <OCMust> <Blankline> <OCMay> \
               <BlankLine> <ATMacro>

AttributeTypeName ::= "Attribute Type:" <String> <EOL>

Description ::= "Description:" <DescriptiveText> <EOL>

DescriptiveText ::= <String> | <DescriptiveText> <EOL> <String>

OCMust ::= "OCMust:" <OCList> <EOL>

OCList ::= <OCName> | <OCList> "," <OCName> | <empty>

OCMay ::= "OCMay:" <OCList> <EOL>

ATMacro ::= "ASN1ATMacro:" <AttributeTypeMacro>

-- The definition of AttributeTypeMacro is adapted from that
-- given in X.501

AttributeTypeMacro ::= <ATname> "ATTRIBUTE" <AttributeSyntax> \
```

```
                          <Multivalued> | <empty>

     ATName ::= <lString>

     AttributeSyntax ::= "WITH ATTRIBUTE SYNTAX" SyntaxChoice

     SyntaxChoice ::= <Syntax> <Constraint> | <ASN1Type> <MatchTypes>

     Syntax ::= <lString>

     Constraint ::= "(" ConstraintAlternative ")" | <empty>

     ConstraintAlternative ::= StringConstraint | IntegerConstraint

     StringConstraint ::= "SIZE" "("  SizeConstraint ")"

     SizeConstraint ::= SingleValue | Range

     SingleValue ::= <Integer>

     Range ::= <Integer> ".." <Integer>

     IntegerConstraint ::= Range

     ASN1Type ::= <UString>
     -- one of ASN.1's base types: e.g. PrintableString,
     -- NumericString, etc.

     MatchTypes ::= "MATCHES FOR" Matches | <empty>

     Matches ::= Match | Matches Match

     Match ::= "EQUALITY" | "SUBSTRINGS" | "ORDERING"

     Multivalued ::= "SINGLE VALUE" | "MULTI VALUE" | <empty>
```

APPENDIX B - Format checking tools

This section includes the source for format checking tools for the two proformas. The tools are written as Bourne shell scripts for UNIX systems.

1. Object class format checker

```
     sed 's/ *: */:/' |
     awk '
     BEGIN {
```

```
        state = "initial"
}

/^$/ {
        next
}

/^Object Class:/ {
        n = index($0, ":")
        if (state != "initial")
        {
                print "Already got object class " oc
                print "Got another object class " substr($0, n+1)
                state = "notOK"
                exit 1
        }
        oc = substr($0, n+1)
        state = "gotOC"
        next
}

/^Description:/ {
        n = index($0, ":")
        if (state != "gotOC")
        {
                print "Got Description: " substr($0, n+1)
                for (i = 0; i < 2 && getline > 0; i++)
                        print $0
                print "..."
                if (state == "initial")
                        print "Expecting Object Class:"
                else
                        print "Expecting ASN1OCMacro:"
                state = "notOK"
                exit 1
        }
        while (getline > 0)
                if (length($0) > 0)
                        continue
                else
                        break
        state = "gotDesc"
        next
}

/^ASN1OCMacro:/ {
        n = index($0, ":")
        if (state != "gotDesc")
```

```
                {
                        print "Got ASN1Macro: " substr($0, n+1)
                        for (i = 0; i < 2 && getline > 0; i++)
                                print $0
                        print "..."
                        if (state == "initial")
                                print "Expecting Object Class:"
                        else
                                print "Expecting Description:"
                        state = "notOK"
                        exit 1
                }
                state = "OK"
                exit 0
        }

        {
                print "Parsing has got confused on seeing line: " $0
                state = "notOK"
                exit 1
        }

        END {
                if (state == "OK")
                        print "Input looks OK"
        }
```

2. Attribute Type format checker

```
        sed 's/ *: */:/' |
        awk '
        BEGIN {
                state = "initial"
        }

        /^$/ {
                next
        }

        /^Attribute Type:/ {
                n = index($0, ":")
                if (state != "initial")
                {
                        got = "Attribute Type:"
                        exit 1
                }
```

```
                state = "gotAT"
                next
        }

        /^Description:/ {
                n = index($0, ":")
                if (state != "gotAT")
                {
                        got = "Description:"
                        exit 1
                }
                while (getline > 0)
                        if (length($0) > 0)
                                continue
                        else
                                break
                state = "gotDesc"
                next
        }

        /^OCMust:/ {
                n = index($0, ":")
                if (state != "gotDesc")
                {
                        got = "OCMust:"
                        exit 1
                }
                state = "gotOCMust"
                next
        }

        /^OCMay:/ {
                n = index($0, ":")
                if (state != "gotOCMust")
                {
                        got = "OCMay:"
                        exit 1
                }
                state = "gotOCMay"
                next
        }

        /^ASN1ATMacro:/ {
                n = index($0, ":")
                if (state != "gotOCMay")
                {
                        got = "ASN1ATMacro:"
                        exit 1
```

```
                }
                state = "OK"
                exit 0
        }

        {
                print "Parsing has got confused on seeing line: " $0
                state = "notOK"
                exit 1
        }

        END {
                if (state == "initial")
                        print "Expecting Attribute Type:"
                else if (state == "gotAT")
                        print "Expecting Description:"
                else if (state == "gotDesc")
                        print "Expecting OCMust:"
                else if (state == "gotOCMust")
                        print "Expecting OCMay:"
                else if (state == "gotOCMay")
                        print "Expecting ASN1ATMacro:"
                if (state != "OK")
                        print "Got " got
                else
                        print "Input looks OK"
        }
```

APPENDIX C - Summary of all Object Classes and Attribute Types

```
        -- Some Important Object Identifiers

        data OBJECT IDENTIFIER ::= {ccitt 9}
        pss OBJECT IDENTIFIER ::= {data 2342}
        ucl OBJECT IDENTIFIER ::= {pss 19200300}
        pilot OBJECT IDENTIFIER ::= {ucl 100}

        pilotAttributeType OBJECT IDENTIFIER ::= {pilot 1}
        pilotAttributeSyntax OBJECT IDENTIFIER ::= {pilot 3}
        pilotObjectClass OBJECT IDENTIFIER ::= {pilot 4}
        pilotGroups OBJECT IDENTIFIER ::= {pilot 10}

        iA5StringSyntax OBJECT IDENTIFIER ::= {pilotAttributeSyntax 4}
        caseIgnoreIA5StringSyntax OBJECT IDENTIFIER ::=
                                                {pilotAttributeSyntax 5}
```

```
-- Standard Object Classes

top OBJECT-CLASS
    MUST CONTAIN {
        objectClass}
::= {objectClass 0}


alias OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        aliasedObjectName}
::= {objectClass 1}


country OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        countryName}
    MAY CONTAIN {
        description,
        searchGuide}
::= {objectClass 2}


locality OBJECT-CLASS
    SUBCLASS OF top
    MAY CONTAIN {
        description,
        localityName,
        stateOrProvinceName,
        searchGuide,
        seeAlso,
        streetAddress}
::= {objectClass 3}


organization OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        organizationName}
    MAY CONTAIN {
        organizationalAttributeSet}
::= {objectClass 4}
```

```
organizationalUnit OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        organizationalUnitName}
    MAY CONTAIN {
        organizationalAttributeSet}
::= {objectClass 5}


person OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName,
        surname}
    MAY CONTAIN {
        description,
        seeAlso,
        telephoneNumber,
        userPassword}
::= {objectClass 6}


organizationalPerson OBJECT-CLASS
    SUBCLASS OF person
    MAY CONTAIN {
        localeAttributeSet,
        organizationalUnitName,
        postalAttributeSet,
        telecommunicationAttributeSet,
        title}
::= {objectClass 7}


organizationalRole OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName}
    MAY CONTAIN {
        description,
        localeAttributeSet,
        organizationalUnitName,
        postalAttributeSet,
        preferredDeliveryMethod,
        roleOccupant,
        seeAlso,
        telecommunicationAttributeSet}
::= {objectClass 8}
```

```
groupOfNames OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName,
        member}
    MAY CONTAIN {
        description,
        organizationName,
        organizationalUnitName,
        owner,
        seeAlso,
        businessCategory}
::= {objectClass 9}


residentialPerson OBJECT-CLASS
    SUBCLASS OF person
    MUST CONTAIN {
        localityName}
    MAY CONTAIN {
        localeAttributeSet,
        postalAttributeSet,
        preferredDeliveryMethod,
        telecommunicationAttributeSet,
        businessCategory}
::= {objectClass 10}


applicationProcess OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName}
    MAY CONTAIN {
        description,
        localityName,
        organizationalUnitName,
        seeAlso}
::= {objectClass 11}


applicationEntity OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName,
        presentationAddress}
    MAY CONTAIN {
        description,
        localityName,
```

```
            organizationName,
            organizationalUnitName,
            seeAlso,
            supportedApplicationContext}
    ::= {objectClass 12}


    dSA OBJECT-CLASS
        SUBCLASS OF applicationEntity
        MAY CONTAIN {
            knowledgeInformation}
    ::= {objectClass 13}


    device OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
            commonName}
        MAY CONTAIN {
            description,
            localityName,
            organizationName,
            organizationalUnitName,
            owner,
            seeAlso,
            serialNumber}
    ::= {objectClass 14}


    strongAuthenticationUser OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
            userCertificate}
    ::= {objectClass 15}


    certificationAuthority OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
            cACertificate,
            certificateRevocationList,
            authorityRevocationList}
        MAY CONTAIN {
            crossCertificatePair}
    ::= {objectClass 16}
```

```
-- Standard MHS Object Classes

mhsDistributionList OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName,
        mhsDLSubmitPermissions,
        mhsORAddresses}
    MAY CONTAIN {
        description,
        organizationName,
        organizationalUnitName,
        owner,
        seeAlso,
        mhsDeliverableContentTypes,
        mhsdeliverableEits,
        mhsDLMembers,
        mhsPreferredDeliveryMethods}
::= {mhsObjectClass 0}


mhsMessageStore OBJECT-CLASS
    SUBCLASS OF applicationEntity
    MAY CONTAIN {
        description,
        owner,
        mhsSupportedOptionalAttributes,
        mhsSupportedAutomaticActions,
        mhsSupportedContentTypes}
::= {mhsObjectClass 1}


mhsMessageTransferAgent OBJECT-CLASS
    SUBCLASS OF applicationEntity
    MAY CONTAIN {
        description,
        owner,
        mhsDeliverableContentLength}
::= {mhsObjectClass 2}


mhsOrganizationalUser OBJECT-CLASS
    SUBCLASS OF organizationalPerson
    MUST CONTAIN {
        mhsORAddresses}
    MAY CONTAIN {
        mhsDeliverableContentLength,
        mhsDeliverableContentTypes,
```

```
            mhsDeliverableEits,
            mhsMessageStoreName,
            mhsPreferredDeliveryMethods }
    ::= {mhsObjectClass 3}


    mhsResidentialUser OBJECT-CLASS
        SUBCLASS OF residentialPerson
        MUST CONTAIN {
            mhsORAddresses}
        MAY CONTAIN {
            mhsDeliverableContentLength,
            mhsDeliverableContentTypes,
            mhsDeliverableEits,
            mhsMessageStoreName,
            mhsPreferredDeliveryMethods }
    ::= {mhsObjectClass 4}


    mhsUserAgent OBJECT-CLASS
        SUBCLASS OF applicationEntity
        MAY CONTAIN {
            mhsDeliverableContentLength,
            mhsDeliverableContentTypes,
            mhsDeliverableEits,
            mhsORAddresses,
            owner}
    ::= {mhsObjectClass 5}




    -- Pilot Object Classes

    pilotObject OBJECT-CLASS
        SUBCLASS OF top
        MAY CONTAIN {
            info,
            photo,
            manager,
            uniqueIdentifier,
            lastModifiedTime,
            lastModifiedBy,
            dITRedirect,
            audio}
    ::= {pilotObjectClass 3}
```

```
pilotPerson OBJECT-CLASS
    SUBCLASS OF person
    MAY CONTAIN {
                userid,
                textEncodedORAddress,
                rfc822Mailbox,
                favouriteDrink,
                roomNumber,
                userClass,
                homeTelephoneNumber,
                homePostalAddress,
                secretary,
                personalTitle,
                preferredDeliveryMethod,
                businessCategory,
                janetMailbox,
                otherMailbox,
                mobileTelephoneNumber,
                pagerTelephoneNumber,
                organizationalStatus,
                mailPreferenceOption,
                personalSignature}
::= {pilotObjectClass 4}


account OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        userid}
    MAY CONTAIN {
        description,
        seeAlso,
        localityName,
        organizationName,
        organizationalUnitName,
        host}
::= {pilotObjectClass 5}


document OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        documentIdentifier}
    MAY CONTAIN {
        commonName,
        description,
        seeAlso,
        localityName,
```

```
            organizationName,
            organizationalUnitName,
            documentTitle,
            documentVersion,
            documentAuthor,
            documentLocation,
            documentPublisher}
    ::= {pilotObjectClass 6}


    room OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
            commonName}
        MAY CONTAIN {
            roomNumber,
            description,
            seeAlso,
            telephoneNumber}
    ::= {pilotObjectClass 7}


    documentSeries OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
            commonName}
        MAY CONTAIN {
            description,
            seeAlso,
            telephoneNumber,
            localityName,
            organizationName,
            organizationalUnitName}
    ::= {pilotObjectClass 9}


    domain OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
            domainComponent}
        MAY CONTAIN {
            associatedName,
            organizationName,
            organizationalAttributeSet}
    ::= {pilotObjectClass 13}
```

```
rFC822localPart OBJECT-CLASS
    SUBCLASS OF domain
    MAY CONTAIN {
        commonName,
        surname,
        description,
        seeAlso,
        telephoneNumber,
        postalAttributeSet,
        telecommunicationAttributeSet}
::= {pilotObjectClass 14}


dNSDomain OBJECT-CLASS
    SUBCLASS OF domain
    MAY CONTAIN {
        ARecord,
        MDRecord,
        MXRecord,
        NSRecord,
        SOARecord,
        CNAMERecord}
::= {pilotObjectClass 15}


domainRelatedObject OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        associatedDomain}
::= {pilotObjectClass 17}


friendlyCountry OBJECT-CLASS
    SUBCLASS OF country
    MUST CONTAIN {
        friendlyCountryName}
::= {pilotObjectClass 18}


simpleSecurityObject OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        userPassword }
::= {pilotObjectClass 19}


pilotOrganization OBJECT-CLASS
    SUBCLASS OF organization, organizationalUnit
```

```
        MAY CONTAIN {
                    buildingName}
    ::= {pilotObjectClass 20}


    pilotDSA OBJECT-CLASS
        SUBCLASS OF dsa
        MUST CONTAIN {
            dSAQuality}
    ::= {pilotObjectClass 21}


    qualityLabelledData OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
            dSAQuality}
        MAY CONTAIN {
            subtreeMinimumQuality,
            subtreeMaximumQuality}
    ::= {pilotObjectClass 22}




    -- Standard Attribute Types

    objectClass ObjectClass
        ::= {attributeType 0}


    aliasedObjectName AliasedObjectName
        ::= {attributeType 1}


    knowledgeInformation ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreString
        ::= {attributeType 2}


    commonName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-common-name))
        ::= {attributeType 3}


    surname ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-surname))
```

```
        ::= {attributeType 4}


    serialNumber ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX printableStringSyntax
        (SIZE (1..ub-serial-number))
        ::= {attributeType 5}


    countryName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX PrintableString
        (SIZE (1..ub-country-code))
        SINGLE VALUE
        ::= {attributeType 6}


    localityName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-locality-name))
        ::= {attributeType 7}


    stateOrProvinceName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-state-name))
        ::= {attributeType 8}


    streetAddress ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-street-address))
        ::= {attributeType 9}


    organizationName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-organization-name))
        ::= {attributeType 10}


    organizationalUnitName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-organizational-unit-name))
        ::= {attributeType 11}


    title ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
```

```
        (SIZE (1..ub-title))
        ::= {attributeType 12}


    description ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-description))
        ::= {attributeType 13}


    searchGuide ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX Guide
        ::= {attributeType 14}


    businessCategory ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-business-category))
        ::= {attributeType 15}


    postalAddress ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX PostalAddress
        MATCHES FOR EQUALITY
        ::= {attributeType 16}


    postalCode ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-postal-code))
        ::= {attributeType 17}


    postOfficeBox ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-post-office-box))
        ::= {attributeType 18}


    physicalDeliveryOfficeName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
        (SIZE (1..ub-physical-office-name))
        ::= {attributeType 19}


    telephoneNumber ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX telephoneNumberSyntax
        (SIZE (1..ub-telephone-number))
```

```
        ::= {attributeType 20}


    telexNumber ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX TelexNumber
        (SIZE (1..ub-telex))
        ::= {attributeType 21}


    teletexTerminalIdentifier ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX TeletexTerminalIdentifier
        (SIZE (1..ub-teletex-terminal-id))
        ::= {attributeType 22}


    facsimileTelephoneNumber ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX FacsimileTelephoneNumber
        ::= {attributeType 23}


    x121Address ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX NumericString
        (SIZE (1..ub-x121-address))
        ::= {attributeType 24}


    internationaliSDNNumber ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX NumericString
        (SIZE (1..ub-isdn-address))
        ::= {attributeType 25}


    registeredAddress ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX PostalAddress
        ::= {attributeType 26}


    destinationIndicator ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX PrintableString
        (SIZE (1..ub-destination-indicator))
        MATCHES FOR EQUALITY SUBSTRINGS
        ::= {attributeType 27}


    preferredDeliveryMethod ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX deliveryMethod
        ::= {attributeType 28}
```

```
presentationAddress ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX PresentationAddress
    MATCHES FOR EQUALITY
    ::= {attributeType 29}


supportedApplicationContext ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
    ::= {attributeType 30}


member ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX distinguishedNameSyntax
    ::= {attributeType 31}


owner ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX distinguishedNameSyntax
    ::= {attributeType 32}


roleOccupant ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX distinguishedNameSyntax
    ::= {attributeType 33}


seeAlso ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX distinguishedNameSyntax
    ::= {attributeType 34}


userPassword ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX Userpassword
    ::= {attributeType 35}


userCertificate ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX UserCertificate
    ::= {attributeType 36}


cACertificate ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX cACertificate
    ::= {attributeType 37}


authorityRevocationList ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX AuthorityRevocationList
```

```
        ::= {attributeType 38}


    certificateRevocationList ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX CertificateRevocationList
        ::= {attributeType 39}


    crossCertificatePair ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX CrossCertificatePair
        ::= {attributeType 40}




    -- Standard MHS Attribute Types

    mhsDeliverableContentLength ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX integer
        ::= {mhsAttributeType 0}


    mhsDeliverableContentTypes ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX oID
        ::= {mhsAttributeType 1}


    mhsDeliverableEits ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX oID
        ::= {mhsAttributeType 2}


    mhsDLMembers ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX oRName
        ::= {mhsAttributeType 3}


    mhsDLSubmitPermissions ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX dLSubmitPermission
        ::= {mhsAttributeType 4}


    mhsMessageStoreName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX dN
        ::= {mhsAttributeType 5}
```

```
mhsORAddresses ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX oRAddress
    ::= {mhsAttributeType 6}


mhsPreferredDeliveryMethods ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX deliveryMethod
    ::= {mhsAttributeType 7}


mhsSupportedAutomaticActions ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX oID
    ::= {mhsAttributeType 8}


mhsSupportedContentTypes ATTRIBUTE

    WITH ATTRIBUTE-SYNTAX oID
    ::= {mhsAttributeType 9}


mhsSupportedOptionalAttributes ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX oID
    ::= {mhsAttributeType 10}




-- Pilot Attribute Types

userid ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
        (SIZE (1 .. ub-user-identifier))
::= {pilotAttributeType 1}


textEncodedORAddress ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
    (SIZE (1 .. ub-text-encoded-or-address))
::= {pilotAttributeType 2}


rfc822Mailbox ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreIA5StringSyntax
        (SIZE (1 .. ub-rfc822-mailbox))
```

```
    ::= {pilotAttributeType 3}


    info ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            caseIgnoreStringSyntax
            (SIZE (1 .. ub-information))
    ::= {pilotAttributeType 4}


    favouriteDrink ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            caseIgnoreStringSyntax
            (SIZE (1 .. ub-favourite-drink))
    ::= {pilotAttributeType 5}


    roomNumber ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            caseIgnoreStringSyntax
            (SIZE (1 .. ub-room-number))
    ::= {pilotAttributeType 6}


    photo ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            CHOICE {
                g3-facsimile [3] G3FacsimileBodyPart
                }
        (SIZE (1 .. ub-photo))
    ::= {pilotAttributeType 7}


    userClass ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            caseIgnoreStringSyntax
            (SIZE (1 .. ub-user-class))
    ::= {pilotAttributeType 8}


    host ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            caseIgnoreStringSyntax
            (SIZE (1 .. ub-host))
    ::= {pilotAttributeType 9}
```

```
manager ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        distinguishedNameSyntax
::= {pilotAttributeType 10}


documentIdentifier ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
        (SIZE (1 .. ub-document-identifier))
::= {pilotAttributeType 11}


documentTitle ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
    (SIZE (1 .. ub-document-title))
::= {pilotAttributeType 12}


documentVersion ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
        (SIZE (1 .. ub-document-version))
::= {pilotAttributeType 13}


documentAuthor ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        distinguishedNameSyntax
::= {pilotAttributeType 14}


documentLocation ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
        (SIZE (1 .. ub-document-location))
::= {pilotAttributeType 15}


homeTelephoneNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        telephoneNumberSyntax
::= {pilotAttributeType 20}


secretary ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
```

```
            distinguishedNameSyntax
        ::= {pilotAttributeType 21}


        otherMailbox ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX
                SEQUENCE {
                        mailboxType PrintableString, -- e.g. Telemail
                        mailbox IA5String  -- e.g. X378:Joe
                }
        ::= {pilotAttributeType 22}


        lastModifiedTime ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX
                uTCTimeSyntax
        ::= {pilotAttributeType 23}


        lastModifiedBy ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX
                distinguishedNameSyntax
        ::= {pilotAttributeType 24}


        domainComponent ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX
                caseIgnoreIA5StringSyntax
                SINGLE VALUE
        ::= {pilotAttributeType 25}


        aRecord ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX
                DNSRecordSyntax
        ::= {pilotAttributeType 26}


        mXRecord ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX
                DNSRecordSyntax
        ::= {pilotAttributeType 28}


        nSRecord ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX
                DNSRecordSyntax
        ::= {pilotAttributeType 29}
```

```
sOARecord ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        DNSRecordSyntax
::= {pilotAttributeType 30}


cNAMERecord ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        iA5StringSyntax
::= {pilotAttributeType 31}


associatedDomain ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreIA5StringSyntax
::= {pilotAttributeType 37}


associatedName ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        distinguishedNameSyntax
::= {pilotAttributeType 38}


homePostalAddress ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        postalAddress
        MATCHES FOR EQUALITY
::= {pilotAttributeType 39}


personalTitle ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
        (SIZE (1 .. ub-personal-title))
::= {pilotAttributeType 40}


mobileTelephoneNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        telephoneNumberSyntax
::= {pilotAttributeType 41}


pagerTelephoneNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        telephoneNumberSyntax
::= {pilotAttributeType 42}
```

```
friendlyCountryName ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
::= {pilotAttributeType 43}


uniqueIdentifier ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
        (SIZE (1 .. ub-unique-identifier))
::= {pilotAttributeType 44}


organizationalStatus ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
        (SIZE (1 .. ub-organizational-status))
::= {pilotAttributeType 45}


janetMailbox ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreIA5StringSyntax
        (SIZE (1 .. ub-janet-mailbox))
::= {pilotAttributeType 46}


mailPreferenceOption ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX ENUMERATED {
            no-list-inclusion(0),
            any-list-inclusion(1),  -- may be added to any lists
            professional-list-inclusion(2)
                                    -- may be added to lists
                                    -- which the list provider
                                    -- views as related to the
                                    -- users professional inter-
                                    -- ests, perhaps evaluated
                                    -- from the business of the
                                    -- organisation or keywords
                                    -- in the entry.
            }
::= {pilotAttributeType 47}


buildingName ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
        (SIZE (1 .. ub-building-name))
```

```
    ::= {pilotAttributeType 48}


    dSAQuality ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX DSAQualitySyntax
            SINGLE VALUE
    ::= {pilotAttributeType 49}


    singleLevelQuality ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX DataQualitySyntax
            SINGLE VALUE


    subtreeMinimumQuality ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX DataQualitySyntax
            SINGLE VALUE
               -- Defaults to singleLevelQuality
    ::= {pilotAttributeType 51}


    subtreeMaximumQuality ATTRIBUTE
            WITH ATTRIBUTE-SYNTAX DataQualitySyntax
            SINGLE VALUE
               -- Defaults to singleLevelQuality
    ::= {pilotAttributeType 52}


    personalSignature ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            CHOICE {
                g3-facsimile [3] G3FacsimileBodyPart
                }
        (SIZE (1 .. ub-personal-signature))
    ::= {pilotAttributeType 53}


    dITRedirect ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            distinguishedNameSyntax
    ::= {pilotAttributeType 54}


    audio ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
            Audio
        (SIZE (1 .. ub-audio))
    ::= {pilotAttributeType 55}
```

```
documentPublisher ATTRIBUTE
        WITH ATTRIBUTE SYNTAX caseIgnoreStringSyntax
::= {pilotAttributeType 56}



-- Generally useful syntaxes


caseIgnoreIA5StringSyntax ATTRIBUTE-SYNTAX
        IA5String
        MATCHES FOR EQUALITY SUBSTRINGS


iA5StringSyntax ATTRIBUTE-SYNTAX
    IA5String
    MATCHES FOR EQUALITY SUBSTRINGS


-- Syntaxes to support the DNS attributes

DNSRecordSyntax ATTRIBUTE-SYNTAX
        IA5String
        MATCHES FOR EQUALITY


NRSInformationSyntax ATTRIBUTE-SYNTAX
        NRSInformation
        MATCHES FOR EQUALITY


NRSInformation ::=  SET {
                [0] Context,
                [1] Address-space-id,
                routes [2] SEQUENCE OF SEQUENCE {
                Route-cost,
                Addressing-info }
        }
```

```
-- Upper bounds on length of attribute values

ub-document-identifier INTEGER ::= 256

ub-document-location INTEGER ::= 256

ub-document-title INTEGER ::= 256

ub-document-version INTEGER ::= 256

ub-favourite-drink INTEGER ::= 256

ub-host INTEGER ::= 256

ub-information INTEGER ::= 2048

ub-unique-identifier INTEGER ::= 256

ub-personal-title INTEGER ::= 256

ub-photo INTEGER ::= 250000

ub-rfc822-mailbox INTEGER ::= 256

ub-room-number INTEGER ::= 256

ub-text-or-address INTEGER ::= 256

ub-user-class INTEGER ::= 256

ub-user-identifier INTEGER ::= 256

ub-organizational-status INTEGER ::= 256

ub-janet-mailbox INTEGER ::= 256

ub-building-name INTEGER ::= 256

ub-personal-signature ::= 50000

ub-audio INTEGER ::= 250000
```

Security Considerations

Security issues are not discussed in this memo.

10. Authors' Addresses

Paul Barker
Department of Computer Science
University College London
Gower Street
London WC1E 6BT
England

Phone: +44 71-380-7366
EMail: P.Barker@cs.ucl.ac.uk

Steve Kille
Department of Computer Science
University College London
Gower Street
London WC1E 6BT
England

Phone: +44 71-380-7294
EMail: S.Kille@cs.ucl.ac.uk

Or send comments to the discussion group: <osi-ds@cs.ucl.ac.uk>.