INTERNATIONAL STANDARD

ISO/IEC 9594-6

Second edition
1995-09-15

Information technology — Open Systems Interconnection — The Directory: Selected attribute types

Technologies de l'information — Interconnexion de systèmes ouverts (OSI) — L'Annuaire: Types d'attributs sélectionnés

CONTENTS

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 9594-6 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 21, *Open systems interconnection, data management and open distributed processing,* in collaboration with ITU-T. The identical text is published as ITU-T Recommendation X.520.

Implementors should note that a defect resolution process exists and that corrections may be applied to this part of ISO/IEC 9594 in the form of technical corrigenda. A list of approved technical corrigenda for this part of ISO/IEC 9594 can be obtained from the subcommittee secretariat. Published technical corrigenda are available from your national standards organization.

This second edition technically revises and enhances ISO/IEC 9594-6:1990. Implementations may still claim conformance to the first edition of this part of ISO/IEC 9594. However, at some point, the first edition will no longer be supported (i.e. reported defects will no longer be resolved). It is recommended that implementations conform to this second edition as soon as possible.

ISO/IEC 9594 consists of the following parts, under the general title *Information technology — Open Systems Interconnection — The Directory*:

— *Part 1: Overview of concepts, models and services*

— *Part 2: Models*

— *Part 3: Abstract service definition*

— *Part 4: Procedures for distributed operation*

— *Part 5: Protocol specifications*

— *Part 6: Selected attribute types*

— *Part 7: Selected object classes*

— *Part 8: Authentication framework*

— *Part 9: Replication*

Annex A forms an integral part of this part of ISO/IEC 9594. Annexes B to E are for information only.

Introduction

This Recommendation I International Standard, together with other Recommendations I International Standards, has been produced to facilitate the interconnection of information processing systems to provide directory services. A set of such systems, together with the directory information which they hold, can be viewed as an integrated whole, called the *Directory*. The information held by the Directory, collectively known as the Directory Information Base (DIB), is typically used to facilitate communication between, with or about objects such as application entities, people, terminals, and distribution lists.

The Directory plays a significant role in Open Systems Interconnection, whose aim is to allow, with a minimum of technical agreement outside of the interconnection standards themselves, the interconnection of information processing systems:

-     from different manufacturers;

-     under different managements;

-     of different levels of complexity; and

-     of different ages.

This Recommendation I International Standard defines a number of attribute types which may be found useful across a range of applications of the Directory, as well as a number of standard attribute syntaxes and matching rules. One particular use for many of the attributes defined herein is in the formation of names, particularly for the classes of object defined in ITU-T Rec. X.521 I ISO/IEC 9594-7.

This second edition technically revises and enhances, but does not replace, the first edition of this Recommendation I International Standard. Implementations may still claim conformance to the first edition.

This second edition (1995) specifies version 1 of the Directory service and protocols. The first edition (1990) also specifies version 1. Differences between the services and between the protocols defined in the two editions are accommodated using the rules of extensibility defined in the second edition of X.519 I ISO/IEC 9594-5.

Annex A, which is an integral part of this Recommendation I International Standard, provides the ASN.1 notation for the complete module which defines the attributes, attribute syntaxes, and matching rules.

Annex B, which is not an integral part of this Recommendation I International Standard, provides a table of attribute types, for easy reference.

Annex C, which is not an integral part of this Recommendation I International Standard, provides suggested upper bounds value constraints used in these Directory Specifications.

Annex D, which is not an integral part of this Recommendation I International Standard, lists alphabetically the attributes and matching rules defined in this Directory Specification.

Annex E, which is not an integral part of this Recommendation I International Standard, lists the amendments and defect reports that have been incorporated to form this edition of this Recommendation I International Standard.

INTERNATIONAL STANDARD

CCITT RECOMMENDATION

INFORMATION TECHNOLOGY – OPEN SYSTEMS INTERCONNECTION – THE DIRECTORY: SELECTED ATTRIBUTE TYPES

SECTION 1 – GENERAL

1 Scope

This Recommendation I International Standard defines a number of attribute types and matching rules which may be found useful across a range of applications of the Directory.

Attribute types and matching rules fall into three categories, as described below.

Some attribute types and matching rules are used by a wide variety of applications or are understood and/or used by the Directory itself.

> NOTE – It is recommended that an attribute type or matching rule defined in this document be used, in preference to the generation of a new one, whenever it is appropriate for the application.

Some attribute types and matching rules are internationally standardized, but are application-specific. These are defined in the standards associated with the application concerned.

Any administrative authority can define its own attribute types and matching rules for any purpose. These are not internationally standardized, and are available to others beyond the administrative authority which created them only by bilateral agreement.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation I International Standard part. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation I International Standard are encouraged to investigate the possibility of applying the most recent editions of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

2.1 Identical Recommendations I International Standards

- ITU-T Recommendation X.500 (1993) I ISO/IEC 9594-1:1995, *Information technology – Open Systems Interconnection – The Directory: Overview of concepts, models and services.*

- ITU-T Recommendation X.501 (1993) I ISO/IEC 9594-2:1995, *Information technology – Open Systems Interconnection – The Directory: Models.*

- ITU-T Recommendation X.511 (1993) I ISO/IEC 9594-3:1995, *Information technology – Open Systems Interconnection – The Directory: Abstract service definition.*

- ITU-T Recommendation X.518 (1993) I ISO/IEC 9594-4:1995, *Information technology – Open Systems Interconnection – The Directory: Procedures for distributed operation.*

- ITU-T Recommendation X.519 (1993) I ISO/IEC 9594-5:1995, *Information technology – Open Systems Interconnection – The Directory: Protocol specifications.*

- ITU-T Recommendation X.521 (1993) I ISO/IEC 9594-7:1995, *Information technology – Open Systems Interconnection – The Directory: Selected object classes.*

- ITU-T Recommendation X.509 (1993) I ISO/IEC 9594-8:1995, *Information technology – Open Systems Interconnection – The Directory: Authentication framework.*

- ITU-T Recommendation X.525 (1993) I ISO/IEC 9594-9:1995, *Information technology – The Directory: Replication.*

- ITU-T Recommendation X.680 (1994) I ISO/IEC 8824-1:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*

- ITU-T Recommendation X.681 (1994) I ISO/IEC 8824-2:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification.*

- ITU-T Recommendation X.682 (1994) I ISO/IEC 8824-3:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*

- ITU-T Recommendation X.683 (1994) I ISO/IEC 8824-4:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Parametrization of ASN.1 specifications.*

2.2 Paired Recommendations I International Standards equivalent in technical content

- CCITT Recommendation X.200 (1988), *Reference Model of Open Systems Interconnection for CCITT Applications.*

  ISO 7498:1984, *Information processing systems — Open Systems Interconnection — Basic Reference Model.*

2.3 Other references

- CCITT Recommendation E.123 (1988), *Notation for National and International Telephone numbers.*

- CCITT Recommendation E.164 (1991), *Numbering plan for the ISDN era.*

- CCITT Recommendation F.1 (1992), *Operational provisions for the international public telegram service.*

- CCITT Recommendation F.200 (1992), *Teletex service.*

- CCITT Recommendation F.401 (1992), *Message handling services: Naming and addressing for public message handling services.*

- CCITT Recommendation T.30 (1993), *Procedures for document facsimile transmission in the general switched telephone network.*

- CCITT Recommendation T.61 (1993), *Character repertoire and coded character sets for the international teletex service.*

- CCITT Recommendation T.62 (1993), *Control procedures for teletex and Group 4 facsimile services.*

- CCITT Recommendation X.121 (1992), *International numbering plan for public data networks.*

- ISO 3166:1993, *Codes for the representation of names of countries.*

3 Definitions

For the purposes of this CCITT Recommendation I International Standard, the following definitions apply.

The following terms are defined in CCITT Rec. X.501 I ISO/IEC 9594-2:

    a) *attribute type;*

    b) *object class;*

    c) *matching rule.*

4 Conventions

With minor exceptions this Directory Specification has been prepared according to the "Presentation of ITU-T/ISO/IEC common text" guidelines in the Guide for ITU-T and ISO/IEC JTC 1 Cooperation, March 1993.

The term "Directory Specification" (as in "this Directory Specification") shall be taken to mean ITU-T Rec. X.520 I ISO/IEC 9594-6. The term "Directory Specifications" shall be taken to mean the X.500-Series Recommendations and all parts of ISO/IEC 9594.

This Directory Specification uses the term "1988 edition systems" to refer to systems conforming to the previous (1988) edition of the Directory Specifications, i.e. the 1988 edition of the CCITT X.500-Series Recommendations and the ISO/IEC 9594:1990 edition. Systems conforming to the current Directory Specifications are referred to as "1993 edition systems".

Attribute types and matching rules are defined in this Recommendation | International Standard by use of the **ATTRIBUTE** and **MATCHING-RULE** information object classes defined in ITU-T Rec. X.501 | ISO/IEC 9594-2.

Examples of the use of the attribute types are described using an informal notation, where attribute type and value pairs are represented by an acronym for the attribute type, followed by an equals sign ('='), followed by the example value for the attribute.

SECTION 2 – SELECTED ATTRIBUTE TYPES

5   Definition of selected attribute types

This Directory Specification defines a number of attribute types which may be found useful across a range of applications of the Directory.

Many of the attributes defined in this Specification are based on a common ASN.1 syntax:

```
DirectoryString { INTEGER : maxSize } ::= CHOICE {
    teletexString              TeletexString (SIZE (1..maxSize)),
    printableString            PrintableString (SIZE (1..maxSize)),
    universalString            UniversalString (SIZE (1..maxSize)) }
```

Some implementations of the Directory do not support the last of these choices, and will not be able to generate, match, or display attributes having such a syntax.

5.1   System attribute types

5.1.1   Knowledge Information

The Knowledge Information attribute type specifies a human readable accumulated description of knowledge mastered by a specific DSA.

NOTE – This attribute is now obsolete.

```
knowledgeInformation ATTRIBUTE        ::=        {
    WITH SYNTAX                DirectoryString {ub-knowledge-information}
    EQUALITY MATCHING RULE      caseIgnoreMatch
    ID                         id-at-knowledgeInformation }
```

5.2   Labeling attribute types

These attributes type are concerned with information about objects which has been explicitly associated with the objects by a labeling process.

5.2.1   Name

The *Name* attribute type is the attribute supertype from which string attribute types typically used for naming may be formed.

```
name ATTRIBUTE                    ::=        {
    WITH SYNTAX                DirectoryString {ub-name}
    EQUALITY MATCHING RULE      caseIgnoreMatch
    SUBSTRINGS MATCHING RULE    caseIgnoreSubstringsMatch
    ID                         id-at-name }
```

5.2.2   Common Name

The *Common Name* attribute type specifies an identifier of an object. A Common Name is not a directory name; it is a (possibly ambiguous) name by which the object is commonly known in some limited scope (such as an organization) and conforms to the naming conventions of the country or culture with which it is associated.

An attribute value for common name is a string chosen either by the person or organization it describes or the organization responsible for the object it describes for devices and application entities. For example, a typical name of a person in an English-speaking country comprises a personal title (e.g. Mr., Ms, Rd, Professor, Sir, Lord), a first name, middle name(s), last name, generation qualifier (if any, e.g. Jr.) and decorations and awards (if any, e.g. QC).

Examples:

CN = "Mr. Robin Lachlan McLeod BSc(Hons) CEng MIEE";

CN = "Divisional Coordination Committee";

CN = "High Speed Modem".

Any variants should be associated with the named object as separate and alternative attribute values.

Other common variants should also be admitted, e.g. use of a middle name as a preferred first name; use of "Bill" in place of "William", etc.

```
commonName ATTRIBUTE        ::=        {
    SUBTYPE OF                         name
    WITH SYNTAX                        DirectoryString {ub-common-name}
    ID                                 id-at-commonName }
```

5.2.3   Surname

The *Surname* attribute type specifies the linguistic construct which normally is inherited by an individual from the individual's parent or assumed by marriage, and by which the individual is commonly known.

An attribute value for Surname is a string, e.g. "McLeod".

```
surname ATTRIBUTE           ::=        {
    SUBTYPE OF                         name
    WITH SYNTAX                        DirectoryString {ub-name}
    ID                                 id-at-surname }
```

5.2.4   Given Name

The *Given Name* attribute type specifies the linguistic construct which is normally given to an individual by the individual's parent, or is chosen by the individual, or by which the individual is commonly known.

An attribute value for Given Name is a string, e.g. "David", or "Jean Paul".

```
givenName ATTRIBUTE         ::=        {
    SUBTYPE OF                         name
    WITH SYNTAX                        DirectoryString {ub-name}
    ID                                 id-at-givenName }
```

5.2.5   Initials

The *Initials* attribute type contains the initials of some or all of an individual's names, but not the surname(s).

An attribute value for Initials is a string, e.g. "D" or "D." or "J.P.".

```
initials ATTRIBUTE          ::=        {
    SUBTYPE OF                         name
    WITH SYNTAX                        DirectoryString {ub-name}
    ID                                 id-at-initials }
```

5.2.6   Generation Qualifier

The *Generation Qualifier* attribute type contains a string which is used to provide generation information to qualify an individual's name.

An attribute value for Generation Qualifier is a string, e.g. "Jr." or "II".

```
generationQualifier ATTRIBUTE   ::=        {
    SUBTYPE OF                             name
    WITH SYNTAX                            DirectoryString {ub-name}
    ID                                     id-at-generationQualifier }
```

5.2.7　Unique Identifier

The *Unique Identifier* attribute type specifies an identifier which may be used to distinguish between object references when a distinguished name has been reused. It may be, for example, an encoded object identifier, certificate, date, timestamp, or some other form of certification on the validity of the distinguished name.

An attribute value for Unique Identifier is a bit string.

```
uniqueIdentifier ATTRIBUTE        ::=        {
    WITH SYNTAX                      UniqueIdentifier
    EQUALITY MATCHING RULE            bitStringMatch
    ID                               id-at-uniqueIdentifier }
UniqueIdentifier                  ::=        BIT STRING
```

5.2.8　DN Qualifier

The *DN Qualifier* attribute type specifies disambiguating information to add to the relative distinguished name of an entry. It is intended to be used for entries held in multiple DSAs which would otherwise have the same name, and that its value be the same in a given DSA for all entries to which this information has been added.

```
dnQualifier ATTRIBUTE             ::=        {
    WITH SYNTAX                      PrintableString
    EQUALITY MATCHING RULE            caseIgnoreMatch
    ORDERING MATCHING RULE            caseIgnoreOrderingMatch
    SUBSTRINGS MATCHING RULE         caseIgnoreSubstringsMatch
    ID                               id-at-dnQualifier }
```

5.2.9　Serial Number

The *Serial Number* attribute type specifies an identifier, the serial number of a device.

An attribute value for Serial Number is a printable string.

```
serialNumber ATTRIBUTE            ::=        {
    WITH SYNTAX                      PrintableString (SIZE (1..ub-serialNumber))
    EQUALITY MATCHING RULE            caseIgnoreMatch
    SUBSTRINGS MATCHING RULE         caseIgnoreSubstringsMatch
    ID                               id-at-serial-number }
```

5.3　Geographical Attribute Types

These attribute types are concerned with geographical positions or regions with which objects are associated.

5.3.1　Country Name

The *Country Name* attribute type specifies a country. When used as a component of a directory name, it identifies the country in which the named object is physically located or with which it is associated in some other important way.

An attribute value for country name is a string chosen from ISO 3166.

```
countryName ATTRIBUTE             ::=        {
    SUBTYPE OF                       name
    WITH SYNTAX                      PrintableString (SIZE (2)) -- IS 3166 codes only
    SINGLE VALUE                     TRUE
    ID                               id-at-countryName }
```

5.3.2　Locality Name

The *Locality Name* attribute type specifies a locality. When used as a component of a directory name, it identifies a geographical area or locality in which the named object is physically located or with which it is associated in some other important way.

An attribute value for Locality Name is a string, e.g. L = "Edinburgh".

```
localityName ATTRIBUTE            ::=        {
    SUBTYPE OF                       name
    WITH SYNTAX                      DirectoryString {ub-locality-name}
    ID                               id-at-localityName }
```

The *Collective Locality Name* attribute type specifies a locality name for a collection of entries.

```
collectiveLocalityName ATTRIBUTE        ::=        {
    SUBTYPE OF                  localityName
    COLLECTIVE                  TRUE
    ID                          id-at-collectiveLocalityName }
```

5.3.3   State or Province Name

The *State or Province Name* attribute type specifies a state or province. When used as a component of a directory name, it identifies a geographical subdivision in which the named object is physically located or with which it is associated in some other important way.

An attribute value for State or Province Name is a string, e.g. S = "Ohio".

```
stateOrProvinceName ATTRIBUTE::=        {
    SUBTYPE OF                  name
    WITH SYNTAX                 DirectoryString {ub-state-name}
    ID                          id-at-stateOrProvinceName }
```

The *Collective State or Province Name* attribute type specifies a state or province name for a collection of entries.

```
collectiveStateOrProvinceName ATTRIBUTE ::=        {
    SUBTYPE OF                  stateOrProvinceName
    COLLECTIVE                  TRUE
    ID                          id-at-collectiveStateOrProvinceName }
```

5.3.4   Street Address

The *Street Address* attribute type specifies a site for the local distribution and physical delivery in a postal address, i.e. the street name, place, avenue, and the house number. When used as a component of a directory name, it identifies the street address at which the named object is located or with which it is associated in some other important way.

An attribute value for Street Address is a string, e.g. "Arnulfstraße 60".

```
streetAddress ATTRIBUTE         ::=        {
    WITH SYNTAX                 DirectoryString {ub-street-address}
    EQUALITY MATCHING RULE       caseIgnoreMatch
    SUBSTRINGS MATCHING RULE    caseIgnoreSubstringsMatch
    ID                          id-at-streetAddress }
```

The *Collective Street Address* attribute type specifies a street address for a collection of entries.

```
collectiveStreetAddress ATTRIBUTE       ::=        {
    SUBTYPE OF                  streetAddress
    COLLECTIVE                  TRUE
    ID                          id-at-collectiveStreetAddress }
```

5.3.5   House Identifier

The *House Identifier* attribute type specifies a linguistic construct used to identify a particular building, for example a house number or house name relative to a street, avenue, town or city, etc.

An attribute value for House Identifier is a string, e.g. "14".

```
houseIdentifier ATTRIBUTE       ::=        {
    WITH SYNTAX                 DirectoryString {ub-name}
    EQUALITY MATCHING RULE       caseIgnoreMatch
    SUBSTRINGS MATCHING RULE     caseIgnoreSubstringsMatch
    ID                          id-at-houseIdentifier }
```

5.4   Organizational attribute types

These attribute types are concerned with organizations and can be used to describe objects in terms of organizations with which they are associated.

5.4.1   OrganizationName

The *OrganizationName* attribute type specifies an organization. When used as a component of a directory name it identifies an organization with which the named object is affiliated.

An attribute value for OrganizationName is a string chosen by the organization (e.g. O = "Scottish Telecom-munications plc"). Any variants should be associated with the named Organization as separate and alternative attribute values.

```
organizationName ATTRIBUTE    ::=        {
    SUBTYPE OF                      name
    WITH SYNTAX                     DirectoryString {ub-organization-name}
    ID                              id-at-organizationName }
```

The *Collective Organization Name* attribute type specifies an organization name for a collection of entries.

```
collectiveOrganizationName ATTRIBUTE    ::=        {
    SUBTYPE OF                      organizationName
    COLLECTIVE                      TRUE
    ID                              id-at-collectiveOrganizationName }
```

5.4.2   Organizational Unit Name

The *Organizational Unit Name* attribute type specifies an organizational unit. When used as a component of a directory name it identifies an organizational unit with which the named object is affiliated.

The designated organizational unit is understood to be part of an organization designated by an OrganizationName attribute. It follows that if an Organizational Unit Name attribute is used in a directory name, it must be associated with an OrganizationName attribute.

An attribute value for Organizational Unit Name is a string chosen by the organization of which it is part (e.g. OU = "Technology Division"). Note that the commonly used abbreviation "TD" would be a separate and alternative attribute value.

Example:

O = "Scottel", OU = "TD"

```
organizationalUnitName ATTRIBUTE    ::=        {
    SUBTYPE OF                      name
    WITH SYNTAX                     DirectoryString {ub-organizational-unit-name}
    ID                              id-at-organizationalUnitName }
```

The *Collective Organizational Unit Name* attribute type specifies an organizational unit name for a collection of entries.

```
collectiveOrganizationalUnitName ATTRIBUTE    ::=        {
    SUBTYPE OF                      organizationalUnitName
    COLLECTIVE                      TRUE
    ID                              id-at-collectiveOrganizationalUnitName }
```

5.4.3   Title

The *Title* attribute type specifies the designated position or function of the object within an organization.

An attribute value for Title is a string.

Example:

T = "Manager, Distributed Applications"

```
title ATTRIBUTE            ::=        {
    SUBTYPE OF                      name
    WITH SYNTAX                     DirectoryString {ub-title}
    ID                              id-at-title }
```

5.5   Explanatory attribute types

These attribute types are concerned with explanations (e.g. in a natural language) of something about an object.

5.5.1   Description

The *Description* attribute type specifies text which describes the associated object.

For example, the object "Standards Interest" might have the associated description "distribution list for exchange of information about intra-company standards development".

An attribute value for Description is a string.

```
description ATTRIBUTE              ::=       {
    WITH SYNTAX                           DirectoryString {ub-description}
    EQUALITY MATCHING RULE                 caseIgnoreMatch
    SUBSTRINGS MATCHING RULE              caseIgnoreSubstringsMatch
    ID                                    id-at-description }
```

5.5.2   Search Guide

The *Search Guide* attribute type specifies information of suggested search criteria which may be included in some entries expected to be a convenient base-object for the search operation, e.g. country or organization.

Search criteria consist of an optional identifier for the type of object sought and combinations of attribute types and logical operators to be used in the construction of a filter. It is possible to specify for each search criteria item the matching level, e.g. approximate match.

The Search Guide attribute may recur to reflect the various types of requests, e.g. search for a Residential Person or an Organizational Person, which may be fulfilled from the given base-object where the Search Guide is read.

```
searchGuide ATTRIBUTE                   ::=       {
    WITH SYNTAX                              Guide
    ID                                       id-at-searchGuide }

Guide                                       ::=       SET {
    objectClass               [0]            OBJECT-CLASS.&id OPTIONAL,
    criteria                  [1]            Criteria }

Criteria                      ::=            CHOICE {
    type                      [0]            CriteriaItem,
    and                       [1]            SET OF Criteria,
    or                        [2]            SET OF Criteria,
    not                       [3]            Criteria }

CriteriaItem                  ::=            CHOICE {
    equality                  [0]            AttributeType,
    substrings                [1]            AttributeType,
    greaterOrEqual            [2]            AttributeType,
    lessOrEqual               [3]            AttributeType,
    approximateMatch          [4]            AttributeType }
```

Example:

The following is a potential value of the Search Guide attribute that could be stored in entries of object class Locality to indicate how entries of object class Residential Person might be found:

```
residential-person-guide Guide ::= {
    objectClass residentialPerson.&id,
    criteria and : {
        type : substrings : commonName.&id,
        type : substrings : streetAddress.&id }}
```

The construction of a filter from this value of Guide is straightforward.

Step (1) produces the intermediate Filter value

```
intermediate-filter Filter              ::=
    and : {
        item : substrings {
            type commonName.&id,
            strings { any : teletexString : "Dubois" }},
        item : substrings {
            type streetAddress.&id,
            strings { any : teletexString "Hugo" }}}
```

Step (2) produces a filter for matching Residential Person entries in the subtree:

```
residential-person-filter Filter ::=
    and : {
        item : equality : {
            type objectClass.&id,
            assertion residentialPerson.&id },
        intermediateFilter }
```

5.5.3   Enhanced Search Guide

The *Enhanced Search Guide* attribute provides an enhancement of the **searchGuide** attribute, adding information about the recommended search depth for searches among subordinate objects of a given object class.

```
enhancedSearchGuide ATTRIBUTE          ::=      {
    WITH SYNTAX                        EnhancedGuide
    ID                                 id-at-enhancedSearchGuide }

EnhancedGuide              ::=      SEQUENCE {
    objectClass           [0]        OBJECT-CLASS.&id,
    criteria              [1]        Criteria,
    subset                [2]        INTEGER
        { baseObject (0), oneLevel (1), wholeSubtree (2) } DEFAULT oneLevel }
```

5.5.4   Business Category

The *Business Category* attribute type specifies information concerning the occupation of some common objects, e.g. people. For example, this attribute provides the facility to interrogate the Directory about people sharing the same occupation.

```
businessCategory ATTRIBUTE      ::=      {
    WITH SYNTAX                        DirectoryString {ub-business-category}
    EQUALITY MATCHING RULE             caseIgnoreMatch
    SUBSTRINGS MATCHING RULE           caseIgnoreSubstringsMatch
    ID                                 id-at-businessCategory }
```

5.6   Postal Addressing attribute types

These attribute types are concerned with information required for physical postal delivery to an object.

5.6.1   Postal Address

The *Postal Address* attribute type specifies the address information required for the physical delivery of postal messages by the postal authority to the named object.

An attribute value for Postal Address will be typically composed of selected attributes from the MHS Unformatted Postal O/R Address version 1 according to CCITT Recommendation F.401 and limited to 6 lines of 30 characters each, including a Postal Country Name. Normally the information contained in such an address could include an addressee's name, street address, city, state or province, postal code and possibly a Post Office Box number depending on the specific requirements of the named object.

```
postalAddress ATTRIBUTE          ::=      {
    WITH SYNTAX                        PostalAddress
    EQUALITY MATCHING RULE             caseIgnoreListMatch
    SUBSTRINGS MATCHING RULE           caseIgnoreListSubstringsMatch
    ID                                 id-at-postalAddress }

PostalAddress      ::=      SEQUENCE SIZE(1..ub-postal-line) OF DirectoryString {ub-postal-string}
```

The *Collective Postal Address* attribute type specifies a postal address for a collection of entries.

```
collectivePostalAddress ATTRIBUTE          ::=      {
    SUBTYPE OF                         postalAddress
    COLLECTIVE                         TRUE
    ID                                 id-at-collectivePostalAddress }
```

5.6.2   Postal Code

The *Postal Code* attribute type specifies the postal code of the named object. If this attribute value is present it will be part of the object's postal address.

An attribute value for Postal Code is a string.

```
postalCode ATTRIBUTE            ::=             {
    WITH SYNTAX                     DirectoryString {ub-postal-code}
    EQUALITY MATCHING RULE           caseIgnoreMatch
    SUBSTRINGS MATCHING RULE        caseIgnoreSubstringsMatch
    ID                              id-at-postalCode }
```

The *Collective Postal Code* attribute type specifies a postal code for a collection of entries.

```
collectivePostalCode ATTRIBUTE  ::=             {
    SUBTYPE OF                      postalCode
    COLLECTIVE                      TRUE
    ID                              id-at-collectivePostalCode }
```

5.6.3   Post Office Box

The *Post Office Box* attribute type specifies the Post Office Box by which the object will receive physical postal delivery. If present, the attribute value is part of the object's postal address.

```
postOfficeBox ATTRIBUTE         ::=             {
    WITH SYNTAX                     DirectoryString {ub-post-office-box}
    EQUALITY MATCHING RULE           caseIgnoreMatch
    SUBSTRINGS MATCHING RULE        caseIgnoreSubstringsMatch
    ID                              id-at-postOfficeBox }
```

The *Collective Post Office Box* attribute type specifies a post office box for a collection of entries.

```
collectivePostOfficeBox ATTRIBUTE       ::=             {
    SUBTYPE OF                      postOfficeBox
    COLLECTIVE                      TRUE
    ID                              id-at-collectivePostOfficeBox }
```

5.6.4   Physical Delivery Office Name

The Physical Delivery Office Name attribute type specifies the name of the city, village, etc. where a physical delivery office is situated.

An attribute value for Physical Delivery Office Name is a string.

```
physicalDeliveryOfficeName ATTRIBUTE    ::=      {
    WITH SYNTAX                     DirectoryString {ub-physical-office-name}
    EQUALITY MATCHING RULE           caseIgnoreMatch
    SUBSTRINGS MATCHING RULE        caseIgnoreSubstringsMatch
    ID                              id-at-physicalDeliveryOfficeName }
```

The *Collective Physical Delivery Office Name* attribute type specifies a physical delivery office name for a collection of entries.

```
collectivePhysicalDeliveryOfficeName  ATTRIBUTE      ::=      {
    SUBTYPE OF                      physicalDeliveryOfficeName
    COLLECTIVE                      TRUE
    ID                              id-at-collectivePhysicalDeliveryOfficeName }
```

5.7   Telecommunications Addressing attribute types

These attribute types are concerned with addressing information needed to communicate with the object using telecommunication means.

5.7.1 Telephone Number

The *Telephone Number* attribute type specifies a telephone number associated with an object.

An attribute value for Telephone Number is a string that complies with the internationally agreed format for showing international telephone numbers, CCITT Recommendation E.123 (e.g. "+ 44 582 10101").

```
telephoneNumber ATTRIBUTE        ::=        {
     WITH SYNTAX                  PrintableString (SIZE (1..ub-telephone-number))
     EQUALITY MATCHING RULE        telephoneNumberMatch
     SUBSTRINGS MATCHING RULE     telephoneNumberSubstringsMatch
     ID                           id-at-telephoneNumber }
```

The *Collective Telephone Number* attribute type specifies a telephone number for a collection of entries.

```
collectiveTelephoneNumber  ATTRIBUTE        ::=        {
     SUBTYPE OF                   telephoneNumber
     COLLECTIVE                   TRUE
     ID                           id-at-collectiveTelephoneNumber }
```

5.7.2 Telex Number

The *Telex Number* attribute type specifies the telex number, country code, and answerback code of a telex terminal associated with an object.

```
telexNumber ATTRIBUTE            ::=        {
     WITH SYNTAX                  TelexNumber
     ID                           id-at-telexNumber }

TelexNumber ::= SEQUENCE {
     telexNumber                  PrintableString (SIZE (1..ub-telex-number)),
     countryCode                  PrintableString (SIZE (1..ub-country-code)),
     answerback                   PrintableString (SIZE (1..ub-answerback)) }
```

The *Collective Telex Number* attribute type specifies a telex number for a collection of entries.

```
collectiveTelexNumber  ATTRIBUTE        ::=        {
     SUBTYPE OF                   telexNumber
     COLLECTIVE                   TRUE
     ID                           id-at-collectiveTelexNumber }
```

5.7.3 Teletex Terminal Identifier

The *Teletex Terminal Identifier* attribute type specifies the Teletex terminal identifier (and, optionally, parameters) for a teletex terminal associated with an object.

An attribute value for Teletex Terminal Identifier is a string which complies with CCITT Recommendation F.200 and an optional set whose components are according to CCITT Recommendation T.62.

```
teletexTerminalIdentifier ATTRIBUTE        ::=        {
     WITH SYNTAX                  TeletexTerminalIdentifier
     ID                           id-at-teletexTerminalIdentifier }

TeletexTerminalIdentifier ::= SEQUENCE {
     teletexTerminal              PrintableString (SIZE(1..ub-teletex-terminal-id)),
     parameters                   TeletexNonBasicParameters OPTIONAL}
```

The *Collective Teletex Terminal Identifer* attribute type specifies a teletex terminal identifier for a collection of entries.

```
collectiveTeletexTerminalIdentifier  ATTRIBUTE        ::=        {
     SUBTYPE OF                   teletexTerminalIdentifier
     COLLECTIVE                   TRUE
     ID                           id-at-collectiveTeletexTerminalIdentifier }
```

5.7.4 Facsimile Telephone Number

The Facsimile Telephone Number attribute type specifies a telephone number for a facsimile terminal (and optionally its parameters) associated with an object.

An attribute value for the facsimile telephone number is a string that complies with the internationally agreed format for showing international telephone numbers, CCITT Recommendation E.123 (e.g. "+81 3 347 7418") and an optional bit string (formatted according to CCITT Recommendation T.30).

```
facsimileTelephoneNumber ATTRIBUTE        ::=       {
    WITH SYNTAX                   FacsimileTelephoneNumber
    ID                            id-at-facsimileTelephoneNumber }

FacsimileTelephoneNumber         ::=    SEQUENCE {
    telephoneNumber                   PrintableString (SIZE (1..ub-telephone-number)),
    parameters                        G3FacsimileNonBasicParameters  OPTIONAL }
```

The *Collective Facsimile Telephone Number* attribute type specifies a facsimile telephone number for a collection of entries.

```
collectiveFacsimileTelephoneNumber  ATTRIBUTE     ::=      {
    SUBTYPE OF                   facsimileTelephoneNumber
    COLLECTIVE                   TRUE
    ID                           id-at-collectiveFacsimileTelephoneNumber }
```

5.7.5 X.121 Address

The *X.121 Address* attribute type specifies an address as defined by CCITT Recommendation X.121 associated with an object.

```
x121Address ATTRIBUTE            ::=      {
    WITH SYNTAX                  NumericString (SIZE (1..ub-x121-address))
    EQUALITY MATCHING RULE        numericStringMatch
    SUBSTRINGS MATCHING RULE     numericStringSubstringsMatch
    ID                           id-at-x121Address }
```

5.7.6 International ISDN Number

The *International ISDN Number* attribute type specifies an International ISDN Number associated with an object.

An attribute value for International ISDN Number is a string which complies with the internationally agreed format for ISDN addresses given in CCITT Recommendation E.164.

```
internationalISDNNumber ATTRIBUTE        ::=       {
    WITH SYNTAX                  NumericString (SIZE (1..ub-international-isdn-number))
    EQUALITY MATCHING RULE        numericStringMatch
    SUBSTRINGS MATCHING RULE     numericStringSubstringsMatch
    ID                           id-at-internationalISDNNumber }
```

The *Collective International ISDN Number* attribute type specifies an international ISDN number for a collection of entries.

```
collectiveInternationalISDNNumber ATTRIBUTE       ::=      {
    SUBTYPE OF                   internationalISDNNumber
    COLLECTIVE                   TRUE
    ID                           id-at-collectiveInternationalISDNNumber }
```

5.7.7 Registered Address

The Registered Address attribute type specifies a mnemonic for an address associated with an object at a particular city location. The mnemonic is registered in the country in which the city is located and is used in the provision of the Public Telegram Service (according to CCITT Recommendation F.1).

```
registeredAddress ATTRIBUTE     ::=      {
    SUBTYPE OF                   postalAddress
    WITH SYNTAX                  PostalAddress
    ID                           id-at-registeredAddress }
```

5.7.8 Destination Indicator

The *Destination Indicator* attribute type specifies (according to CCITT Recommendation F.1 and CCITT Recommendation F.31) the country and city associated with the object (the addressee) needed to provide the Public Telegram Service.

An attribute value for Destination Indicator is a string.

```
destinationIndicator ATTRIBUTE   ::=      {
      WITH SYNTAX                          PrintableString (SIZE (1..ub-destination-indicator))
                                           -- alphabetical characters only
      EQUALITY MATCHING RULE                caseIgnoreMatch
      SUBSTRINGS MATCHING RULE             caseIgnoreSubstringsMatch
      ID                                   id-at-destinationIndicator }
```

5.8 Preferences attribute types

These attribute types are concerned with the preferences of an object.

5.8.1 Preferred Delivery Method

The *Preferred Delivery Method* attribute type specifies the object's priority order regarding the method to be used for communicating with it.

```
preferredDeliveryMethod ATTRIBUTE        ::= {
      WITH SYNTAX                         SEQUENCE OF INTEGER {
         ny-delivery-method                  (0),
         mhs-delivery                        (1),
         physical-delivery                   (2),
         telex-delivery                      (3),
         teletex-delivery                    (4),
         g3-facsimile-delivery               (5),
         g4-facsimile-delivery               (6),
         ia5-terminal-delivery               (7),
         videotex-delivery                   (8),
         telephone-delivery                  (9) }
      SINGLE VALUE                        TRUE
      ID                                  id-at-preferredDeliveryMethod }
```

5.9 OSI Application attribute types

These attribute types are concerned with information regarding objects in the OSI Application Layer.

5.9.1 Presentation Address

The *Presentation Address* attribute type specifies a presentation address associated with an object representing an OSI application entity.

An attribute value for Presentation Address is a presentation address as defined in ISO 7498.

```
presentationAddress ATTRIBUTE   ::=       {
      WITH SYNTAX                          PresentationAddress
      EQUALITY MATCHING RULE                presentationAddressMatch
      SINGLE VALUE                         TRUE
      ID                                   id-at-presentationAddress }

PresentationAddress ::= SEQUENCE {
      pSelector           [0]             OCTET STRING OPTIONAL,
      sSelector           [1]             OCTET STRING OPTIONAL,
      tSelector           [2]             OCTET STRING OPTIONAL,
      nAddresses          [3]             SET SIZE (1..MAX)  OF  OCTET STRING }
```

5.9.2 Supported Application Context

The *Supported Application Context* attribute type specifies the object identifier(s) of application context(s) that the object (an OSI application entity) supports.

```
supportedApplicationContext ATTRIBUTE    ::=    {
    WITH SYNTAX                 OBJECT IDENTIFIER
    EQUALITY MATCHING RULE       objectIdentifierMatch
    ID                          id-at-supportedApplicationContext }
```

5.9.3 Protocol Information

The *Protocol Information* attribute type associates protocol information with each network address in the Presentation Address attribute.

For each nAddress, the protocol component identifies the protocol or profile for the network and transport layers.

```
protocolInformation ATTRIBUTE   ::=    {
    WITH SYNTAX                 ProtocolInformation
    EQUALITY MATCHING RULE       protocolInformationMatch
    ID                          id-at-protocolInformation }

ProtocolInformation             ::=    SEQUENCE {
    nAddress                    OCTET STRING,
    profiles                    SET OF OBJECT IDENTIFIER }
```

5.10 Relational attribute types

These attribute types are concerned with information regarding the objects which are related to a particular object in certain ways.

5.10.1 Distinguished Name

The *Distinguished Name* attribute type is an attribute for specifying the name of an object.

```
distinguishedName ATTRIBUTE     ::=    {
    WITH SYNTAX                 DistinguishedName
    EQUALITY MATCHING RULE       distinguishedNameMatch
    ID                          id-at-distinguishedName }
```

5.10.2 Member

The *Member* attribute type specifies a group of names associated with the object.

An attribute value for Member is a distinguished name.

```
member ATTRIBUTE                ::=    {
    SUBTYPE OF                  distinguishedName
    ID                          id-at-member }
```

5.10.3 Unique Member

The *Unique Member* attribute type specifies a group of unique names associated with an object. A unique name is a name that is optionally disambiguated by the inclusion of its unique identifier.

An attribute value for Unique Member is a distinguished name accompanied by an optional unique identifier.

```
uniqueMember ATTRIBUTE          ::=    {
    WITH SYNTAX                 NameAndOptionalUID
    EQUALITY MATCHING RULE       uniqueMemberMatch
    ID                          id-at-uniqueMember }

NameAndOptionalUID              ::=    SEQUENCE {
    dn                          DistinguishedName,
    uid                         UniqueIdentifier OPTIONAL }
```

5.10.4 Owner

The *Owner* attribute type specifies the name of some object which has some responsibility for the associated object.

An attribute value for Owner is a distinguished name (which could represent a group of names) and can recur.

```
owner ATTRIBUTE              ::=        {
      SUBTYPE OF                        distinguishedName
      ID                                id-at-owner }
```

5.10.5 Role Occupant

The *Role Occupant* attribute type specifies the name of an object which fulfills an organizational role.

An attribute value for Role Occupant is a distinguished name.

```
roleOccupant ATTRIBUTE       ::=        {
      SUBTYPE OF                        distinguishedName
      ID                                id-at-roleOccupant }
```

5.10.6 See Also

The *See Also* attribute type specifies names of other Directory objects which may be other aspects (in some sense) of the same real world object.

An attribute value for See Also is a distinguished name.

```
seeAlso ATTRIBUTE            ::=        {
      SUBTYPE OF                        distinguishedName
      ID                                id-at-seeAlso }
```

SECTION 3 – MATCHING RULES

6 Definition of matching rules

NOTE – For definitions of **objectIdentifierMatch** and **distinguishedNameMatch**, see ITU-T Recom-mendation X.501 | ISO/IEC 9594-2.

6.1 String matching rules

In the matching rules specified in 7.1.1 through 7.1.11, the following spaces are regarded as not significant:

- leading spaces (i.e. those preceding the first printing character);
- trailing spaces (i.e. those following the last printing character);
- multiple consecutive internal spaces (these are taken as equivalent to a single space character).

In the matching rules to which these apply, the strings to be matched shall be matched as if the insignificant spaces were not present in either string.

6.1.1 Case Ignore Match

The *Case Ignore Match* rule compares for equality a presented string with an attribute value of type **DirectoryString**, without regard to the case (upper or lower) of the strings (e.g. "Dundee" and "DUNDEE" match).

```
caseIgnoreMatch MATCHING-RULE::=       {
      SYNTAX                            DirectoryString {ub-match}
      ID                                id-mr-caseIgnoreMatch }
```

The rule returns TRUE if the strings are the same length and corresponding characters are identical except possibly with regard to case.

Where the strings being matched are of different ASN.1 syntax, the comparison proceeds as normal so long as the corresponding characters are in both character sets. Otherwise matching fails.

6.1.2   Case Ignore Ordering Match

The *Case Ignore Ordering Match* rule compares the collation order of a presented string an attribute value of type **DirectoryString**, without regard to the case (upper or lower) of the strings.

```
caseIgnoreOrderingMatch MATCHING-RULE ::=        {
    SYNTAX                          DirectoryString {ub-match}
    ID                              id-mr-caseIgnoreOrderingMatch }
```

The rule returns TRUE if the attribute value is "less" or appears earlier than the presented value, when the strings are compared using the normal collation order for their syntax after lower-case letters in both strings have been replaced by their upper-case equivalents.

Where the strings being matched are of different ASN.1 syntax, the comparison proceeds as normal so long as the corresponding characters are in both character sets. Otherwise matching fails.

6.1.3   Case Ignore Substrings Match

The *Case Ignore Substrings Match* rule determines whether a presented value is a substring of an attribute value of type **DirectoryString**, without regard to the case (upper or lower) of the strings.

```
caseIgnoreSubstringsMatch MATCHING-RULE           ::=        {
    SYNTAX                          SubstringAssertion
    ID                              id-mr-caseIgnoreSubstringsMatch }

SubstringAssertion ::= SEQUENCE OF CHOICE {
    initial             [0]         DirectoryString {ub-match},
    any                 [1]         DirectoryString {ub-match},
    final               [2]         DirectoryString {ub-match} }
    -- at most one initial and one final component
```

The rule returns TRUE if there is a partitioning of the attribute value (into portions) such that:

– the specified substrings (**initial**, **any**, **final**) match different portions of the value in the order of the **strings** sequence;

– **initial**, if present, matches the first portion of the value;

– **final**, if present, matches the last portion of the value;

– **any**, if present, matches some arbitrary portion of the value.

There shall be at most one **initial**, and at most one **final** in **strings**. If **initial** is present, it shall be the first element of **strings**. If **final** is present, it shall be the last element of **strings**. There shall be zero or more **any** in **strings**.

For a component of substrings to match a portion of the attribute value, corresponding characters must be identical, except in regard to case. Where the strings being matched are of different ASN.1 syntax, the comparison proceeds as normal so long as the corresponding characters are in both character sets. Otherwise matching fails.

6.1.4   Case Exact Match

The *Case Exact Match* rule compares for equality a presented string with an attribute value of type **DirectoryString**.

```
caseExactMatch MATCHING-RULE            ::=        {
    SYNTAX                          DirectoryString {ub-match}
    ID                              id-mr-caseExactMatch }
```

The rule is identical to the **caseIgnoreMatch** rule except that case is not ignored.

6.1.5   Case Exact Ordering Match

The *Case Exact Ordering Match* rule compares the collation order of a presented string with an attribute value of type **DirectoryString**.

```
caseExactOrderingMatch MATCHING-RULE ::=        {
    SYNTAX                          DirectoryString {ub-match}
    ID                              id-mr-caseExactOrderingMatch }
```

The rule is identical to the **caseIgnoreOrderingMatch** rule except that lower-case letters are not replaced by upper-case letters.

6.1.6 Case Exact Substrings Match

The *Case Exact Substrings Match* rule determines whether a presented value is a substring of an attribute value of type **DirectoryString**.

```
caseExactSubstringsMatch MATCHING-RULE ::=       {
    SYNTAX                          SubstringAssertion -- only the PrintableString choice
    ID                              id-mr-caseExactSubstringsMatch }
```

The rule is identical to the **caseIgnoreSubstringsMatch** rule except that case is not ignored.

6.1.7 Numeric String Match

The *Numeric String Match* rule compares for equality a presented numeric string with an attribute value of type **NumericString**.

```
numericStringMatch MATCHING-RULE       ::=       {
    SYNTAX                          NumericString
    ID                              id-mr-numericStringMatch }
```

The rule is identical to the **caseIgnoreMatch** rule except that all space characters are skipped during comparison (case is irrelevant as characters are numeric).

6.1.8 Numeric String Ordering Match

The *Numeric String Ordering Match* rule compares the collation order of a presented string with an attribute value of type **NumericString**.

```
numericStringOrderingMatch MATCHING-RULE       ::=       {
    SYNTAX                          NumericString
    ID                              id-mr-numericStringOrderingMatch }
```

The rule is identical to the **caseIgnoreOrderingMatch** rule except that all space characters are skipped during comparison (case is irrelevant as characters are numeric).

6.1.9 Numeric String Substrings Match

The *Numeric String Substrings Match* rule determines whether a presented value is a substring of an attribute value of type **NumericString**.

```
numericStringSubstringsMatch MATCHING-RULE       ::=       {
    SYNTAX                          SubstringAssertion
    ID                              id-mr-numericStringSubstringsMatch }
```

The rule is identical to the **caseIgnoreSubstringsMatch** rule except that all space characters are skipped during comparison (case is irrelevant as characters are numeric).

6.1.10 Case Ignore List Match

The *Case Ignore List Match* rule compares for equality a presented sequence of strings with an attribute value which is a sequence of **DirectoryString**s, without regard to the case (upper or lower) of the strings.

```
caseIgnoreListMatch MATCHING-RULE       ::=       {
    SYNTAX                          SEQUENCE OF DirectoryString {ub-match}
    ID                              id-mr-caseIgnoreListMatch }
```

The rule returns TRUE if and only if the number of strings in each is the same, and corresponding strings match. The latter matching is as for the **caseIgnoreMatch** matching rule.

6.1.11 Case Ignore List Substrings Match

The *Case Ignore List Substring* rule compares a presented substring with an attribute value which is a sequence of **DirectoryString**s, but where the case (upper or lower) is not significant for comparison purposes.

```
caseIgnoreListSubstringsMatch MATCHING-RULE       ::=       {
    SYNTAX                          SubstringAssertion
    ID                              id-mr-caseIgnoreListSubstringsMatch }
```

A presented value matches a stored value if and only if the presented value matches the string formed by concatenating the strings of the stored value. This matching is done according to the **caseIgnoreSubstringsMatch** rule; however, none of the **initial, any,** or **final** values of the presented value are considered to match a substring of the concatenated string which spans more than one of the strings of the stored value.

6.2   Syntax-based matching rules

6.2.1   Boolean Match

The *Boolean Match* rule compares for equality a presented Boolean value with an attribute value of type **BOOLEAN**.

```
booleanMatch MATCHING-RULE  ::=        {
    SYNTAX                         BOOLEAN
    ID                             id-mr-booleanMatch }
```

The rule returns TRUE if the values are the same, i.e. both are TRUE or both are FALSE.

6.2.2   Integer Match

The *Integer Match* rule compares for equality a presented integer value with an attribute value of type **INTEGER**.

```
integerMatch MATCHING-RULE  ::=        {
    SYNTAX                         INTEGER
    ID                             id-mr-integerMatch }
```

The rule returns TRUE if the integers are equal.

6.2.3   Integer Ordering Match

The *Integer Ordering Match* rule compares a presented integer value with an attribute value of type **INTEGER**.

```
integerOrderingMatch MATCHING-RULE      ::=     {
    SYNTAX                         INTEGER
    ID                             id-mr-integerOrderingMatch }
```

The rule returns TRUE if the attribute value is less than the presented value.

6.2.4   Bit String Match

The *Bit String Match* rule compares a presented bit string with an attribute value of type **BIT STRING**.

```
bitStringMatch MATCHING-RULE::=        {
    SYNTAX                         BIT STRING
    ID                             id-mr-bitStringMatch }
```

The rule returns TRUE if the attribute value has the same number of bits as the presented value and the bits match on a bitwise basis.

6.2.5   Octet String Match

The *Octet String Match* rule compares for equality a presented octet string with an attribute value of type **OCTET STRING**.

```
octetStringMatch MATCHING-RULE         ::=     {
    SYNTAX                         OCTET STRING
    ID                             id-mr-octetStringMatch }
```

The rule returns TRUE if and only if the strings are the same length and corresponding octets are identical.

6.2.6   Octet String Ordering Match

The *Octet String Ordering* rule compares the collation order of a presented octet string with an attribute value of type **OCTET STRING**.

```
octetStringOrderingMatch MATCHING-RULE ::=      {
    SYNTAX                         OCTET STRING
    ID                             id-mr-octetStringOrderingMatch }
```

The rule compares octet strings from first octet to last octet, and from the most significant bit to the least significant bit within the octet. The first occurrence of a different bit determines the ordering of the strings. A zero bit precedes a one bit. If the strings are identical but contain different numbers of octets, the shorter string precedes the longer string.

6.2.7   Octet String Substrings Match

The *Octet String Substrings Match* rule determines whether a presented octet string is a substring of an attribute value of type **OCTET STRING**.

```
octetStringSubstringsMatch MATCHING-RULE       ::=      {
    SYNTAX                          OctetSubstringAssertion
    ID                              id-mr-octetStringSubstringsMatch }

OctetSubstringAssertion ::= SEQUENCE OF CHOICE {
    initial              [0]     OCTET STRING,
    any                  [1]     OCTET STRING,
    final                [2]     OCTET STRING }
    -- at most one initial and one final component
```

The rule returns TRUE if the attribute value contains the sequence of octets in the presented string, as described for **caseIgnoreSubstringsMatch**.

6.2.8   Telephone Number Match

The *Telephone Number Match* rule compares for equality a presented value with an attribute value of type **PrintableString** which is a telephone number.

```
telephoneNumberMatch MATCHING-RULE   ::=      {
    SYNTAX                          PrintableString
    ID                              id-mr-telephoneNumberMatch }
```

The rules for matching are identical to those for **caseIgnoreMatch**, except that all space and "-" characters are skipped during the comparison.

6.2.9   Telephone Number Substrings Match

The *Telephone Number Substrings Match* rule determines if a presented substring is a substring of an attribute value of type **PrintableString** which is a telephone number.

```
telephoneNumberSubstringsMatch MATCHING-RULE  ::=      {
    SYNTAX                          SubstringAssertion
    ID                              id-mr-telephoneNumberSubstringsMatch }
```

The rules for matching are identical to those for **caseExactSubstringsMatch**, except that all space and "-" characters are skipped during the comparison.

6.2.10   Presentation Address Match

The *Presentation Address Match* rule compares for equality a presented Presentation Address with an attribute value of type **PresentationAddress**.

```
presentationAddressMatch MATCHING-RULE ::=       {
    SYNTAX                          PresentationAddress
    ID                              id-mr-presentationAddressMatch }
```

The rule returns TRUE if and only if the selectors of the presented and stored presentation address are equal and the presented **nAddresses** are a subset of the stored ones.

6.2.11   Unique Member Match

The *Unique Member Match* rule compares for equality a presented Unique Member value with an attribute value of type **NameAndOptionalUID**.

```
uniqueMemberMatch MATCHING-RULE       ::=      {
    SYNTAX                          NameAndOptionalUID
    ID                              id-mr-uniqueMemberMatch }
```

The rule returns TRUE if and only if the **dn** components of the attribute value and the presented value match according to the **distinguishedNameMatch** rule, and the **uid** component is absent from the attribute value or matches the corresponding component from the presented value according to the **bitStringMatch** rule.

6.2.12   Protocol Information Match

The *Protocol Information Match* rule compares for equality presented values of **ProtocolInformation** with values of the same type.

```
protocolInformationMatch MATCHING-RULE ::=        {
    SYNTAX                              OCTET STRING
    ID                                  id-mr-protocolInformationMatch }
```

A value of the assertion syntax is derived from a value of the attribute syntax by using the **nAddress** component.

The value returns True if the presented value and the **nAddress** component of the stored value match according to the **octetStringMatch** rule.

6.3   Time matching rules

6.3.1   UTC Time Match

The *UTC Time Match* rule compares for equality a presented value with an attribute value of type **UTCTime**.

```
uTCTimeMatch MATCHING-RULE          ::=        {
    SYNTAX                              UTCTime
    ID                                  id-mr-uTCTimeMatch }
```

The rule returns TRUE if the attribute value represents the same time as the presented value. If a UTC time is specified with the seconds absent, the number of seconds is assumed to be zero.

6.3.2   UTC Time Ordering Match

The *UTC Time Ordering* rule compares the time ordering of a presented value with an attribute value of type **UTCTime**.

```
uTCTimeOrderingMatch MATCHING-RULE ::=        {
    SYNTAX                              UTCTime
    ID                                  id-mr-uTCTimeOrderingMatch }
```

The rule returns TRUE if the attribute value represents a time which is earlier than the presented time. If a UTC time is specified with the seconds absent, the number of seconds is assumed to be zero.

6.3.3   Generalized Time Match

The *Generalized Time Match* rule compares for equality a presented value with an attribute value of type **GeneralizedTime** [as per 34.3 b) or c) of CCITT Recommendation X.208 I ISO/IEC 8824].

```
generalizedTimeMatch MATCHING-RULE   ::=        {
    SYNTAX                              GeneralizedTime
                                        -- as per 34.3 b) or c) of CCITT Rec. X.208 I ISO/IEC 8824
    ID                                  id-mr-generalizedTimeMatch }
```

The rule returns TRUE if the attribute value represents the same time as the presented value. If a time is specified with the minutes or seconds absent, the number of minutes or seconds is assumed to be zero.

6.3.4   Generalized Time Ordering Match

The *Generalized Time Ordering* rule compares the time ordering of a presented value with an attribute value of type **GeneralizedTime** [as per 34.3 b) and c) of CCITT Recommendation X.208 I ISO/IEC 8824].

```
generalizedTimeOrderingMatch MATCHING-RULE    ::=        {
    SYNTAX                              GeneralizedTime
                                        -- as per 34.3 b) or c) of CCITT Rec. X.208 I ISO/IEC 8824
    ID                                  id-mr-generalizedTimeOrderingMatch }
```

The rule returns TRUE if the attribute value represents a time which is earlier than the presented time. If a time is specified with the minutes or seconds absent, the number of minutes or seconds is assumed to be zero.

6.4 First component matching rules

6.4.1 Integer First Component Match

The *Integer First Component Match* rule compares for equality a presented integer value with an attribute value of type **SEQUENCE** whose first component is mandatory and of type **INTEGER**.

```
integerFirstComponentMatch MATCHING-RULE     ::=     {
    SYNTAX                          INTEGER
    ID                              id-mr-integerFirstComponentMatch }
```

The rule returns TRUE if the attribute value has a first component whose value equals the presented integer.

A value of the assertion syntax is derived from a value of the attribute syntax by using the value of the first component of the SEQUENCE.

6.4.2 Object Identifier First Component Match

The *Object Identifier First Component Match* rule compares for equality a presented object identifier value with attribute values of type **SEQUENCE** whose first component is mandatory and of type **OBJECT IDENTIFIER**.

```
objectIdentifierFirstComponentMatch MATCHING-RULE     ::=     {
    SYNTAX                          OBJECT IDENTIFIER
    ID                              id-mr-objectIdentifierFirstComponentMatch }
```

The rule returns TRUE if the attribute value has a first component whose value matches the presented object identifier using the rules of **objectIdentifierMatch**.

A value of the assertion syntax is derived from a value of the attribute syntax by using the value of the first component of the SEQUENCE.

6.4.3 Directory String First Component Match

The *Directory String First Component Match* rule compares for equality a presented **DirectoryString** value with an attribute value of type **SEQUENCE** whose first component is mandatory and of type **DirectoryString**.

```
directoryStringFirstComponentMatch MATCHING-RULE     ::=     {
    SYNTAX                          DirectoryString {ub-directory-string-first-component-match}
    ID                              id-mr-directoryStringFirstComponentMatch }
```

The rule returns TRUE if the attribute value has a first component whose value matches the presented **DirectoryString** using the rules of **caseIgnoreMatch**.

A value of the assertion syntax is derived from a value of the attribute syntax by using the value of the first component of the SEQUENCE.

6.5 Word matching rules

6.5.1 Word Match

The *Word Match* rule compares a presented string with words in an attribute value of type **DirectoryString**.

```
wordMatch MATCHING-RULE     ::=     {
    SYNTAX                          DirectoryString {ub-match}
    ID                              id-mr-wordMatch }
```

The rule returns TRUE if a presented word matches any word in the attribute value. Individual word matching is as for the **caseIgnoreMatch** matching rule. The precise definition of a "word" is a local matter.

6.5.2 Keyword Match

The *Keyword Match* rule compares a presented string with keywords in an attribute value of type **DirectoryString**.

```
keywordMatch MATCHING-RULE ::=     {
    SYNTAX                          DirectoryString {ub-match}
    ID                              id-mr-keywordMatch }
```

The rule returns TRUE if a presented value matches any *keyword* in the attribute value. The identification of keywords in an attribute value and of the exactness of match are both local matters.

Annex A

Selected attribute types in ASN.1

(This annex forms an integral part of this Recommendation I International Standard)

This annex includes all of the ASN.1 type and value definitions contained in this Directory Specification in the form of the ASN.1 module **SelectedAttributeTypes**.

```
SelectedAttributeTypes {joint-iso-ccitt ds(5) module(1) selectedAttributeTypes(5) 2}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --

-- The types and values defined in this module are exported for use in the other ASN.1 modules contained
-- within the Directory Specifications, and for the use of other applications which will use them to access
-- Directory services. Other applications may use them for their own purposes, but this will not constrain
-- extensions and modifications needed to maintain or improve the Directory service.

IMPORTS
        informationFramework, upperBounds, id-at, id-mr
                FROM  UsefulDefinitions {joint-iso-ccitt ds(5) module(1) usefulDefinitions(0) 2 }

        ATTRIBUTE, MATCHING-RULE, AttributeType, OBJECT-CLASS, DistinguishedName,
        objectIdentifierMatch, distinguishedNameMatch
                FROM InformationFramework informationFramework

        G3FacsimileNonBasicParameters, TeletexNonBasicParameters
                FROM MTSAbstractService{joint-iso-ccitt mhs-motis(6) mts(3) modules(0)
                        mts-abstract-service(1)}

        ub-answerback, ub-name, ub-common-name, ub-surname, ub-serial-number, ub-locality-name,
        ub-state-name, ub-street-address, ub-organization-name, ub-organizational-unit-name, ub-title,
        ub-description, ub-business-category, ub-postal-line, ub-postal-string, ub-postal-code,
        ub-post-office-box, ub-physical-office-name, ub-telex-number, ub-country-code,
        ub-teletex-terminal-id, ub-telephone-number, ub-x121-address, ub-international-isdn-number,
        ub-destination-indicator, ub-user-password, ub-match, ub-knowledge-information, ub-name,
        ub-directory-string-first-component-match
                FROM UpperBounds upperBounds     ;

-- Directory string type --

DirectoryString { INTEGER : maxSize } ::= CHOICE {
        teletexString           TeletexString (SIZE (1..maxSize)),
        printableString         PrintableString (SIZE (1..maxSize)),
        universalString         UniversalString (SIZE (1..maxSize)) }

-- Attribute types --

knowledgeInformation ATTRIBUTE          ::=     {
        WITH SYNTAX             DirectoryString { ub-knowledge-information }
        EQUALITY MATCHING RULE  caseIgnoreMatch
        ID                      id-at-knowledgeInformation }

name ATTRIBUTE ::=      {
        WITH SYNTAX             DirectoryString { ub-name }
        EQUALITY MATCHING RULE  caseIgnoreMatch
        SUBSTRINGS MATCHING RULE caseIgnoreSubstringsMatch
        ID                      id-at-name }

commonName ATTRIBUTE ::=        {
        SUBTYPE OF      name
        WITH SYNTAX             DirectoryString {ub-common-name}
        ID                      id-at-commonName }
```

```
surname ATTRIBUTE          ::=            {
    SUBTYPE OF              name
    WITH SYNTAX                            DirectoryString {ub-name}
    ID                                     id-at-surname }

givenName ATTRIBUTE        ::=    {
    SUBTYPE OF              name
    WITH SYNTAX                            DirectoryString {ub-name}
    ID                                     id-at-givenName }

initials ATTRIBUTE ::=            {
    SUBTYPE OF              name
    WITH SYNTAX                            DirectoryString {ub-name}
    ID                                     id-at-initials }

generationQualifier ATTRIBUTE   ::=             {
    SUBTYPE OF              name
    WITH SYNTAX                            DirectoryString {ub-name}
    ID                                     id-at-generationQualifier}

uniqueIdentifier ATTRIBUTE      ::=      {
    WITH SYNTAX                            UniqueIdentifier
    EQUALITY MATCHING RULE                 bitStringMatch
    ID                                     id-at-uniqueIdentifier }

UniqueIdentifier    ::=       BIT STRING

dnQualifier ATTRIBUTE     ::=     {
    WITH SYNTAX                            PrintableString
    EQUALITY MATCHING RULE                 caseIgnoreMatch
    ORDERING MATCHING RULE                 caseIgnoreOrderingMatch
    SUBSTRINGS MATCHING RULE   caseIgnoreSubstringsMatch
    ID                                     id-at-dnQualifier }

serialNumber ATTRIBUTE    ::=     {
    WITH SYNTAX                            PrintableString (SIZE (1..ub-serial-number))
    EQUALITY MATCHING RULE                 caseIgnoreMatch
    SUBSTRINGS MATCHING RULE   caseIgnoreSubstringsMatch
    ID                                     id-at-serialNumber }

countryName ATTRIBUTE  ::=       {
    SUBTYPE OF             name
    WITH SYNTAX                            PrintableString (SIZE (2)) -- IS 3166 codes only
    SINGLE VALUE                           TRUE
    ID                                     id-at-countryName }

localityName ATTRIBUTE  ::=      {
    SUBTYPE OF             name
    WITH SYNTAX                            DirectoryString {ub-locality-name}
    ID                                     id-at-localityName }

collectiveLocalityName ATTRIBUTE        ::=      {
    SUBTYPE OF             localityName
    COLLECTIVE            TRUE
    ID                                     id-at-collectiveLocalityName }

stateOrProvinceName ATTRIBUTE           ::=      {
    SUBTYPE OF             name
    WITH SYNTAX                            DirectoryString {ub-state-name}
    ID                                     id-at-stateOrProvinceName }

collectiveStateOrProvinceName ATTRIBUTE         ::=      {
    SUBTYPE OF             stateOrProvinceName
    COLLECTIVE            TRUE
    ID                                     id-at-collectiveStateOrProvinceName }

streetAddress ATTRIBUTE ::=      {
    WITH SYNTAX                            DirectoryString {ub-street-address}
    EQUALITY MATCHING RULE                 caseIgnoreMatch
    SUBSTRINGS MATCHING RULE   caseIgnoreSubstringsMatch
    ID                                     id-at-streetAddress }
```

```
collectiveStreetAddress ATTRIBUTE      ::=      {
      SUBTYPE OF              streetAddress
      COLLECTIVE              TRUE
      ID                      id-at-collectiveStreetAddress }

houseIdentifier ATTRIBUTE       ::=     {
      WITH SYNTAX             DirectoryString {ub-name}
      EQUALITY MATCHING RULE  caseIgnoreMatch
      SUBSTRINGS MATCHING RULE caseIgnoreSubstringsMatch
      ID                      id-at-houseIdentifier }

organizationName ATTRIBUTE      ::=     {
      SUBTYPE OF              name
      WITH SYNTAX             DirectoryString {ub-organization-name}
      ID                      id-at-organizationName }

collectiveOrganizationName ATTRIBUTE ::=      {
      SUBTYPE OF              organizationName
      COLLECTIVE              TRUE
      ID                      id-at-collectiveOrganizationName }

organizationalUnitName ATTRIBUTE       ::=     {
      SUBTYPE OF              name
      WITH SYNTAX             DirectoryString {ub-organizational-unit-name}
      ID                      id-at-organizationalUnitName }

collectiveOrganizationalUnitName ATTRIBUTE    ::=      {
      SUBTYPE OF              organizationalUnitName
      COLLECTIVE              TRUE
      ID                      id-at-collectiveOrganizationalUnitName }

title ATTRIBUTE   ::=                   {
      SUBTYPE OF              name
      WITH SYNTAX             DirectoryString {ub-title}
      ID                      id-at-title }

description ATTRIBUTE     ::=           {
      WITH SYNTAX             DirectoryString {ub-description}
      EQUALITY MATCHING RULE  caseIgnoreMatch
      SUBSTRINGS MATCHING RULE caseIgnoreSubstringsMatch
      ID                      id-at-description }

searchGuide ATTRIBUTE   ::=       {
      WITH SYNTAX             Guide
      ID                      id-at-searchGuide }

Guide ::=    SET {
      objectClass            [0]   OBJECT-CLASS.&id OPTIONAL,
      criteria               [1]   Criteria }

Criteria     ::=      CHOICE {
      type                   [0]   CriteriaItem,
      and                    [1]   SET OF Criteria,
      or                     [2]   SET OF Criteria,
      not                    [3]   Criteria}

CriteriaItem ::= CHOICE {
      equality               [0]   AttributeType,
      substrings             [1]   AttributeType,
      greaterOrEqual         [2]   AttributeType,
      lessOrEqual            [3]   AttributeType,
      approximateMatch       [4]   AttributeType}

enhancedSearchGuide ATTRIBUTE          ::=             {
      WITH SYNTAX            EnhancedGuide
      ID                     id-at-enhancedSearchGuide }

EnhancedGuide       ::=        SEQUENCE {
      objectClass           [0]   OBJECT-CLASS.&id,
      criteria              [1]   Criteria,
      subset                [2]   INTEGER
          { baseObject (0), oneLevel (1), wholeSubtree (2) } DEFAULT oneLevel }
```

```
businessCategory ATTRIBUTE        ::=      {
    WITH SYNTAX                   DirectoryString {ub-business-category}
    EQUALITY MATCHING RULE        caseIgnoreMatch
    SUBSTRINGS MATCHING RULE      caseIgnoreSubstringsMatch
    ID                            id-at-businessCategory }

postalAddress ATTRIBUTE ::=      {
    WITH SYNTAX                   PostalAddress
    EQUALITY MATCHING RULE        caseIgnoreListMatch
    SUBSTRINGS MATCHING RULE      caseIgnoreListSubstringsMatch
    ID                            id-at-postalAddress }

PostalAddress          ::=        SEQUENCE SIZE(1..ub-postal-line) OF  DirectoryString {ub-postal-string}

collectivePostalAddress ATTRIBUTE        ::=      {
    SUBTYPE OF                    ostalAddress
    COLLECTIVE                    TRUE
    ID                            id-at-collectivePostalAddress }

postalCode ATTRIBUTE       ::=            {
    WITH SYNTAX                   DirectoryString {ub-postal-code}
    EQUALITY MATCHING RULE        caseIgnoreMatch
    SUBSTRINGS MATCHING RULE      caseIgnoreSubstringsMatch
    ID                            id-at-postalCode }

collectivePostalCode ATTRIBUTE  ::=     {
    SUBTYPE OF                    postalCode
    COLLECTIVE                    TRUE
    ID                            id-at-collectivePostalCode }

postOfficeBox ATTRIBUTE ::=      {
    WITH SYNTAX                   DirectoryString {ub-post-office-box}
    EQUALITY MATCHING RULE        caseIgnoreMatch
    SUBSTRINGS MATCHING RULE      caseIgnoreSubstringsMatch
    ID                            id-at-postOfficeBox }

collectivePostOfficeBox ATTRIBUTE        ::=      {
    SUBTYPE OF                    postOfficeBox
    COLLECTIVE                    TRUE
    ID                            id-at-collectivePostOfficeBox }

physicalDeliveryOfficeName ATTRIBUTE ::=      {
    WITH SYNTAX                   DirectoryString {ub-physical-office-name}
    EQUALITY MATCHING RULE        caseIgnoreMatch
    SUBSTRINGS MATCHING RULE      caseIgnoreSubstringsMatch
    ID                            id-at-physicalDeliveryOfficeName }

collectivePhysicalDeliveryOfficeName ATTRIBUTE        ::=      {
    SUBTYPE OF                    physicalDeliveryOfficeName
    COLLECTIVE                    TRUE
    ID                            id-at-collectivePhysicalDeliveryOfficeName }

telephoneNumber ATTRIBUTE        ::=      {
    WITH SYNTAX                   PrintableString (SIZE (1..ub-telephone-number))
    EQUALITY MATCHING RULE        telephoneNumberMatch
    SUBSTRINGS MATCHING RULE      telephoneNumberSubstringsMatch
    ID                            id-at-telephoneNumber }

collectiveTelephoneNumber ATTRIBUTE         ::=      {
    SUBTYPE OF                    telephoneNumber
    COLLECTIVE                    TRUE
    ID                            id-at-collectiveTelephoneNumber }

telexNumber ATTRIBUTE   ::=      {
    WITH SYNTAX                   TelexNumber
    ID                            id-at-telexNumber }

TelexNumber ::= SEQUENCE {
    telexNumber             PrintableString (SIZE(1..ub-telex-number)),
    countryCode             PrintableString (SIZE(1..ub-country-code)),
    answerback              PrintableString (SIZE(1..ub-answerback))}
```

```
collectiveTelexNumber  ATTRIBUTE        ::=     {
        SUBTYPE OF              telexNumber
        COLLECTIVE              TRUE
        ID                              id-at-collectiveTelexNumber }

teletexTerminalIdentifier ATTRIBUTE     ::=     {
        WITH SYNTAX             TeletexTerminalIdentifier
        ID                              id-at-teletexTerminalIdentifier }

TeletexTerminalIdentifier ::= SEQUENCE {
        teletexTerminal         PrintableString (SIZE(1..ub-teletex-terminal-id)),
        parameters              TeletexNonBasicParameters OPTIONAL}

collectiveTeletexTerminalIdentifier  ATTRIBUTE ::=      {
        SUBTYPE OF              teletexTerminalIdentifier
        COLLECTIVE              TRUE
        ID                              id-at-collectiveTeletexTerminalIdentifier }

facsimileTelephoneNumber ATTRIBUTE      ::=     {
        WITH SYNTAX             FacsimileTelephoneNumber
        ID                              id-at-facsimileTelephoneNumber }

FacsimileTelephoneNumber        ::=     SEQUENCE {
        telephoneNumber         PrintableString (SIZE(1.. ub-telephone-number)),
        parameters              G3FacsimileNonBasicParameters  OPTIONAL}

collectiveFacsimileTelephoneNumber  ATTRIBUTE       ::=     {
        SUBTYPE OF              facsimileTelephoneNumber
        COLLECTIVE              TRUE
        ID                              id-at-collectiveFacsimileTelephoneNumber }

x121Address ATTRIBUTE   ::=     {
        WITH SYNTAX             NumericString (SIZE (1..ub-x121-address))
        EQUALITY MATCHING RULE  numericStringMatch
        SUBSTRINGS MATCHING RULE  numericStringSubstringsMatch
        ID                              id-at-x121Address }

internationalISDNNumber ATTRIBUTE       ::=     {
        WITH SYNTAX             NumericString (SIZE (1..ub-international-isdn-number))
        EQUALITY MATCHING RULE  numericStringMatch
        SUBSTRINGS MATCHING RULE  numericStringSubstringsMatch
        ID                              id-at-internationalISDNNumber }

collectiveInternationalISDNNumber ATTRIBUTE ::=        {
        SUBTYPE OF              internationalISDNNumber
        COLLECTIVE              TRUE
        ID                              id-at-collectiveInternationalISDNNumber }

registeredAddress ATTRIBUTE     ::=     {
        SUBTYPE OF              postalAddress
        WITH SYNTAX             PostalAddress
        ID                              id-at-registeredAddress }

destinationIndicator ATTRIBUTE  ::=     {
        WITH SYNTAX             PrintableString (SIZE (1..ub-destination-indicator))
                                -- alphabetical characters only
        EQUALITY MATCHING RULE  caseIgnoreMatch
        SUBSTRINGS MATCHING RULE  caseIgnoreSubstringsMatch
        ID                              id-at-destinationIndicator }

preferredDeliveryMethod ATTRIBUTE       ::=     {
        WITH SYNTAX             SEQUENCE OF INTEGER {
                any-delivery-method             (0),
                mhs-delivery                    (1),
                physical-delivery               (2),
                telex-delivery                  (3),
                teletex-delivery                (4),
                g3-facsimile-delivery   (5),
                g4-facsimile-delivery   (6),
                ia5-terminal-delivery           (7),
                videotex-delivery               (8),
                telephone-delivery              (9) }
```

```
       SINGLE VALUE                    TRUE
       ID                              id-at-preferredDeliveryMethod }

presentationAddress ATTRIBUTE  ::=     {
       WITH SYNTAX                     PresentationAddress
       EQUALITY MATCHING RULE          presentationAddressMatch
       SINGLE VALUE                    TRUE
       ID                              id-at-presentationAddress }

PresentationAddress ::= SEQUENCE {
       pSelector          [0]     OCTET STRING OPTIONAL,
       sSelector          [1]     OCTET STRING OPTIONAL,
       tSelector          [2]     OCTET STRING OPTIONAL,
       nAddresses         [3]     SET SIZE (1..MAX) OF  OCTET STRING}

supportedApplicationContext ATTRIBUTE        ::=     {
       WITH SYNTAX                     OBJECT IDENTIFIER
       EQUALITY MATCHING RULE          objectIdentifierMatch
       ID                              id-at-supportedApplicationContext }

protocolInformation ATTRIBUTE  ::=     {
       WITH SYNTAX                     ProtocolInformation
       EQUALITY MATCHING RULE          protocolInformationMatch
       ID                              id-at-protocolInformation }

ProtocolInformation           ::=      SEQUENCE {
       nAddress                        OCTET STRING,
       profiles                        SET OF OBJECT IDENTIFIER }

distinguishedName ATTRIBUTE    ::=     {
       WITH SYNTAX                     DistinguishedName
       EQUALITY MATCHING RULE          distinguishedNameMatch
       ID                              id-at-distinguishedName }

member ATTRIBUTE          ::=     {
       SUBTYPE OF                 distinguishedName
       ID                              id-at-member }

uniqueMember ATTRIBUTE         ::=          {
       WITH SYNTAX                     NameAndOptionalUID
       EQUALITY MATCHING RULE          uniqueMemberMatch
       ID                              id-at-uniqueMember }

NameAndOptionalUID            ::=      SEQUENCE {
       dn                              DistinguishedName,
       uid                             UniqueIdentifier OPTIONAL }

owner ATTRIBUTE               ::=     {
       SUBTYPE OF                 distinguishedName
       ID                              id-at-owner }

roleOccupant ATTRIBUTE  ::=     {
       SUBTYPE OF                 distinguishedName
       ID                              id-at-roleOccupant }

seeAlso ATTRIBUTE             ::=     {
       SUBTYPE OF                 distinguishedName
       ID                              id-at-seeAlso }

-- Matching rules --

caseIgnoreMatch MATCHING-RULE          ::=     {
       SYNTAX                     DirectoryString {ub-match}
       ID                         id-mr-caseIgnoreMatch }

caseIgnoreOrderingMatch MATCHING-RULE      ::=     {
       SYNTAX                     DirectoryString {ub-match}
       ID                         id-mr-caseIgnoreOrderingMatch }

caseIgnoreSubstringsMatch MATCHING-RULE ::=     {
       SYNTAX                     SubstringAssertion
       ID                         id-mr-caseIgnoreSubstringsMatch }
```

```
SubstringAssertion ::= SEQUENCE OF CHOICE {
        initial         [0]     DirectoryString {ub-match},
        any                     [1]     DirectoryString {ub-match},
        final                   [2]     DirectoryString {ub-match} }
        -- at most one initial and one final component

caseExactMatch MATCHING-RULE         ::=     {
        SYNTAX                  DirectoryString {ub-match}
        ID                      id-mr-caseExactMatch }

caseExactOrderingMatch MATCHING-RULE         ::=     {
        SYNTAX                  DirectoryString {ub-match}
        ID                      id-mr-caseExactOrderingMatch }

caseExactSubstringsMatch MATCHING-RULE         ::=     {
        SYNTAX                  SubstringAssertion -- only the PrintableString choice
        ID                      id-mr-caseExactSubstringsMatch }

numericStringMatch MATCHING-RULE ::=     {
        SYNTAX                  NumericString
        ID                      id-mr-numericStringMatch }

numericStringOrderingMatch MATCHING-RULE ::=     {
        SYNTAX                  NumericString
        ID                      id-mr-numericStringOrderingMatch }

numericStringSubstringsMatch MATCHING-RULE         ::=     {
        SYNTAX                  SubstringAssertion
        ID                      id-mr-numericStringSubstringsMatch }

caseIgnoreListMatch MATCHING-RULE  ::=     {
        SYNTAX                  SEQUENCE OF DirectoryString {ub-match}
        ID                      id-mr-caseIgnoreListMatch }

caseIgnoreListSubstringsMatch MATCHING-RULE         ::=     {
        SYNTAX                  SubstringAssertion
        ID                      id-mr-caseIgnoreListSubstringsMatch }

booleanMatch MATCHING-RULE  ::=     {
        SYNTAX                  BOOLEAN
        ID                      id-mr-booleanMatch }

integerMatch MATCHING-RULE  ::=     {
        SYNTAX                  INTEGER
        ID                      id-mr-integerMatch }

integerOrderingMatch MATCHING-RULE ::=     {
        SYNTAX                  INTEGER
        ID                      id-mr-integerOrderingMatch }

bitStringMatch MATCHING-RULE ::=     {
        SYNTAX                  BIT STRING
        ID                      id-mr-bitStringMatch }

octetStringMatch MATCHING-RULE         ::=     {
        SYNTAX                  OCTET STRING
        ID                      id-mr-octetStringMatch }

octetStringOrderingMatch MATCHING-RULE         ::=     {
        SYNTAX                  OCTET STRING
        ID                      id-mr-octetStringOrderingMatch }

octetStringSubstringsMatch MATCHING-RULE         ::=     {
        SYNTAX                  OctetSubstringAssertion
        ID                      id-mr-octetStringSubstringsMatch }

OctetSubstringAssertion ::= SEQUENCE OF CHOICE {
        initial         [0]     OCTET STRING,
        any                     [1]     OCTET STRING,
        final                   [2]     OCTET STRING }
        -- at most one initial and one final component

telephoneNumberMatch MATCHING-RULE         ::=     {
        SYNTAX                  PrintableString
        ID                      id-mr-telephoneNumberMatch }
```