id-pt-modify OBJECT IDENTIFIER ::={id-pt 3}

END

none(0),

signed (1)}

ANNEX B

(to Recommendation X.511)

Directory object identifiers

This Annex is part of the standard.

This Annex includes all of the ASN.1 object identifiers contained in this Recommendation in the form of the ASN.1 module "**DirectoryObjectIdentifiers**".

```
DirectoryObjectIdentifiers {joint-ISO-CCITT ds(5) modules(1)
                                        directoryObjectIdentifiers(9)}
DEFINITIONS  ::=
BEGIN
EXPORTS
      id-ot-directory, id-ot-dua, id-pt-read, id-pt-search, id-pt-modify;
IMPORTS
      id-ot, id-pt
  FROM UsefulDefinitions {joint-iso-ccitt ds(5) modules(1),
                                  usefulDefinitions(0)};
-- Objects --

id-ot-directoryOBJECT IDENTIFIER ::={id-ot 1}

id-ot-dua              OBJECT IDENTIFIER ::={id-ot 2}

-- Port Types --

id-pt-read     OBJECT IDENTIFIER ::={id-pt 1}
```

```
                    OPTIONAL }
EntryInformationSelection ::=  SET {
    attributeTypes
            CHOICE {
                    allAttributes [0] NULL,
                    select [1] SET OF AttributeType
                    -- empty set implies no attributes
                    -- are requested --}
            DEFAULT allAttributes NULL,
    infoTypes [2] INTEGER {
                    attributeTypesOnly (0),
                    attributeTypesAndValues (1) } DEFAULT
                        attributeTypesandValues }
EntryInformation  ::=  SEQUENCE {
    DistinguishedName,
    fromEntry BOOLEAN DEFAULT TRUE,
    SET OF CHOICE {
                        AttributeType,
                        Attribute} OPTIONAL }
Filter  ::=CHOICE {
    item      [0]   FilterItem,
    and             [1]SET OF Filter,
    or                [2] SET OF Filter,
    not             [3]Filter }
FilterItem::=  CHOICE {
    equality      [0]  AttributeValueAssertion,
    substrings[1]  SEQUENCE {
                    typeAttributeType,
                    stringsSEQUENCE OF CHOICE {
                        initial[0]AttributeValue,
                        any  [1]       AttributeValue,
                        final[2]       AttributeValue}},
    greaterOrEqual [2]AttributeValueAssertion,
                    lessOrEqual[3]AttributeValueAssertion,
                    present     [4]AttributeType,
    approximateMatch  [5]AttributeValueAssertion }
SecurityParameters ::=  SET {
    certification-Path  [0]  CertificationPath OPTIONAL,
    name              [1] DistinguishedName OPTIONAL,
    time              [2] UTCTime OPTIONAL,
    random          [3]BIT STRING OPTIONAL,
    target          [4]ProtectionRequest OPTIONAL }
```

```
                    problem  [0]  UpdateProblem }

UpdateProblem  ::=  INTEGER {
      namingViolation (1),

      objectClassViolation (2),

      notAllowedOnNonLeaf (3),

      notAllowedOnRDN (4),

      entryAlreadyExists (5),

      affectsMultipleDSAs (6),

      objectClassModificationProhibited (7)}

-- common arguments/results --

CommonArguments  ::=  SET {
      [30] ServiceControls DEFAULT {}

      [29] SecurityParameters DEFAULT {},

      requestor [28] DistinguishedName OPTIONAL,

      [27] OperationProgress DEFAULT notStarted,

      aliasedRDNs [26] INTEGER OPTIONAL,

      extensions [25] SET OF Extension OPTIONAL }

Extension  ::=SET {
      identifier[0]  INTEGER,

      critical          [1]  BOOLEAN DEFAULT FALSE,

      item                    [2]  ANY DEFINED BY identifier }

CommonResults ::=SET {
      [30]       SecurityParameters  OPTIONAL,

      performer [29] DistinguishedName OPTIONAL,

      aliasDereferenced  [28] BOOLEAN DEFAULT FALSE}

-- common data types --

ServiceControls  ::= SET {
      options [0]BIT STRING {

      preferChaining (0),

      chainingProhibited (1),

      localScope (2),

      dontUseCopy (3),

      dontDereferenceAliases(4)}

                        DEFAULT{},

      priority [1] INTEGER {
                        low (0),

                        medium (1),

                        high (2) } DEFAULT medium,

timeLimit       [2]   INTEGER OPTIONAL,

sizeLimit       [3]   INTEGER OPTIONAL,

scopeOfReferral [4]INTEGER {
```

```
        attributeOrValueAlreadyExists (6) }

NameError  ::=  ABSTRACT-ERROR
        PARAMETER SET {
                        problem [0]  NameProblem,
                        matched [1]  Name}
NameProblem  ::=INTEGER {
                        noSuchObject (1),
                        aliasProblem (2),
                        invalidAttributeSyntax (3),
                        aliasDereferencingProblem (4)}

Referral  ::=  ABSTRACT-ERROR  -- not literally an "error"
        PARAMETER SET {
                        candidate [0]  ContinuationReference}

SecurityError  ::=ABSTRACT-ERROR
        PARAMETER SET {
                problem [0] SecurityProblem }
SecurityProblem  ::= INTEGER {
        inappropriateAuthentication (1),
        invalidCredentials (2),
        insufficientAccessRights (3),
        invalidSignature (4),
        protectionRequired (5),
        noInformation (6) }

ServiceError  ::= ABSTRACT-ERROR
        PARAMETER SET {
                 problem [0] ServiceProblem }

ServiceProblem  ::=INTEGER {
        busy (1),
        unavailable (2),
        unwillingToPerform (3),
        chainingRequired (4),
        unableToProceed (5),
        invalidReference (6),
        timeLimitExceeded (7),
        administrativeLimitExceeded (8),
        loopDetected (9),
        unavailableCriticalExtension (10),
        outOfScope (11),
        ditError (12) }

UpdateError  ::=  ABSTRACT-ERROR
```

```
EntryModification  ::=  CHOICE {
        addAttribute  [0]Attribute,
        removeAttribute[1]AttributeType,
        addValues     [2]Attribute,
        removeValues[3]Attribute}
ModifyRDN ::=  ABSTRACT-OPERATION
        ARGUMENT   ModifyRDNArgument
        RESULT          ModifyRDNResult
        ERRORS {
                        NameError,
                        ServiceError, Referral, SecurityError,
                        UpdateError }
ModifyRDNArgument  ::=  OPTIONALLY-SIGNED SET {
        object          [0]  DistinguishedName,
        newRDN          [1]  RelativeDistinguishedName,
        deleteoldRDN[2]  BOOLEAN DEFAULT FALSE,
        COMPONENTS OF CommonArguments }
ModifyRDNResult  ::=  NULL

--  errors and parameters --
Abandoned  ::=  ABSTRACT-ERROR  -- not literally an "error"
AbandonFailed  ::=  ABSTRACT-ERROR
        PARAMETER SET {
                problem   [0]  AbandonProblem,
                operation  [1]  InvokeID}
AbandonProblem  ::=INTEGER {
                        noSuchOperation (1),
                        tooLate (2),
                        cannotAbandon (3)}
AttributeError  ::=ABSTRACT-ERROR
        PARAMETER SET {
                object[0]  Name,
                problems[1]  SET OF SEQUENCE {
                        problem[0]AttributeProblem,
                        type[1] AttributeType,
                        value[2]AttributeValue OPTIONAL }}
AttributeProblem  ::=
        INTEGER {
                 noSuchAttributeOrValue (1),
                 invalidAttributeSyntax (2),
                 undefinedAttributeType (3),
                 inappropriateMatching (4),
```

```
        DistinguishedName OPTIONAL,
        entries  [0]  SET OF EntryInformation,
        partialOutcomeQualifier
                [2] partialOutcomeQualifier OPTIONAL,
        COMPONENTS OF CommonResults },
        uncorrelatedSearchInfo [0] SET OF
                        SearchResult }
AddEntry  ::=  ABSTRACT-OPERATION
        ARGUMENT   AddEntryArgument
        RESULT          AddEntryResult
        ERRORS {
                        AttributeError, NameError,
                        ServiceError, Referral, SecurityError
                        UpdateError }
AddEntryArgument  ::=  OPTIONALLY-SIGNED SET {
                        object[0]  DistinguishedName,
                        entry[1]  SET OF Attribute,
                        COMPONENTS OF CommonArguments}
AddEntryResult ::=NULL
RemoveEntry ::=  ABSTRACT-OPERATION
        ARGUMENT   RemoveEntryArgument
        RESULT          RemoveEntryResult
        ERRORS {
                        NameError,
                        ServiceError, Referral, SecurityError,
                        UpdateError}
RemoveEntryArgument  ::=  OPTIONALLY-SIGNED SET {
                        object[0]  DistinguishedName,
                        COMPONENTS OF CommonArguments }
RemoveEntryResult  ::=  NULL
ModifyEntry  ::=  ABSTRACT-OPERATION
        ARGUMENT   ModifyEntryArgument
        RESULT          ModifyEntryResult
        ERRORS {
                        AttributeError, NameError,
                        ServiceError, Referral, SecurityError,
                        UpdateError}
ModifyEntryArgument  ::=  OPTIONALLY-SIGNED SET {
        object          [0]DistinguishedName,
        changes [1]   SEQUENCE OF EntryModification,
        COMPONENTS OF CommonArguments }
```

```
        COMPONENTS OF CommonArguments }
ListResult  ::=  OPTIONALLY-SIGNED CHOICE{
        listInfo SET {
        DistinguishedName OPTIONAL
        subordinates [1] SET OF SEQUENCE {
                        RelativeDistinguishedName,
                        aliasEntry[0]  BOOLEAN DEFAULT FALSE,
                        fromEntry[1]  BOOLEAN DEFAULT TRUE },
                        partialOutcomeQualifier [2] PartialOutcomeQualifier
                      OPTIONAL,
                        COMPONENTS OF CommonResults},
        uncorrelatedListInfo [0] SET OF
                        ListResult }
PartialOutcomeQualifier  ::=  SET {
        limitProblem[0]   LimitProblem OPTIONAL,
        unexplored[1]   SET OF
                        ContinuationReference OPTIONAL,
        unavailableCriticalExtensions [2] BOOLEAN DEFAULT FALSE }
LimitProblem  ::=  INTEGER {
        timeLimitExceeded(0),
        sizeLimitExceeded(1),
        administrativeLimitExceeded (2) }
Search  ::=  ABSTRACT-OPERATION
        ARGUMENT  SearchArgument
        RESULT          SearchResult
        ERRORS {
                        AttributeError, NameError,
                        ServiceError, Referral, Abandoned,
                        SecurityError }
SearchArgument ::=OPTIONALLY-SIGNED SET {
        baseObject[0]  Name,
        subset          [1]  INTEGER {
                        baseObject(0),
                        oneLevel(1),
                        wholeSubtree(2)} DEFAULT baseObject,
        filter          [2]  Filter DEFAULT and {},
        searchAliases[3]  BOOLEAN DEFAULT TRUE,
        selection[4]  EntryInformationSelection DEFAULT {},
        COMPONENTS OF CommonArguments }
SearchResult  ::=OPTIONALLY-SIGNED
        CHOICE {
```

```
                    AttributeError, NameError,
                    ServiceError, Referral, Abandoned,
                    SecurityError }
ReadArgument ::= OPTIONALLY-SIGNED SET {
    object          [0] Name,
    selection[1] EntryInformationSelection
                                    DEFAULT {},
    COMPONENTS OF CommonArguments }
ReadResult ::= OPTIONALLY-SIGNED SET {
    entry    [0] EntryInformation,
    COMPONENTS OF CommonResults }
Compare ::= ABSTRACT-OPERATION
    ARGUMENT  CompareArgument
    RESULT        CompareResult
    ERRORS {
                    AttributeError, NameError,
                    ServiceError, Referral, Abandoned,
                    SecurityError }
CompareArgument ::= OPTIONALLY-SIGNED SET {
    object          [0] Name,
    purported[1] AttributeValueAssertion,
    COMPONENTS OF CommonArguments }
CompareResult ::= OPTIONALLY-SIGNED SET {
    DistinguishedName OPTIONAL,
    matched [0] BOOLEAN,
    fromEntry [1] BOOLEAN DEFAULT TRUE,
    COMPONENTS OF CommonResults }
Abandon ::= ABSTRACT-OPERATION
    ARGUMENT AbandonArgument
    RESULT       AbandonResult
    ERRORS {AbandonFailed}
AbandonArgument ::= SEQUENCE {
    InvokeID [0] InvokeID}
AbandonResult ::=NULL
List ::= ABSTRACT-OPERATION
    ARGUMENT  ListArgument
    RESULT        ListResult
    ERRORS {
                    AttributeError, NameError,
                    ServiceError, Referral, Abandoned,
                    SecurityError }
ListArgument ::= OPTIONALLY-SIGNED SET {
```

```
        TO { readPort, searchPort, modifyPort }

        BIND

        ARGUMENT   DirectoryBindArgument

        RESULT          DirectoryBindResult

        BIND-ERRORDirectoryBindError

DirectoryBindArgument  ::=  SET {
        credentials[0]  Credentials  OPTIONAL,

        versions        [1]  Versions DEFAULT v1988}

Credentials ::=CHOICE {
        simple    [0]  SimpleCredentials,

        strong    [1]  StrongCredentials,

        externalProcedure [2] EXTERNAL }

SimpleCredentials ::=  SEQUENCE {
        name              [0] DistinguishedName,

        validity [1] SET {

                time1 [0]  UTCTime OPTIONAL,

                time2 [1]  UTCTime OPTIONAL,

                random1 [2]  BIT STRING OPTIONAL,

                random2 [3]  BIT STRING OPTIONAL }

                                OPTIONAL,

                password   [2]  OCTET STRING OPTIONAL }

StrongCredentials ::= SET {
        certification-path[0]  CertificationPath OPTIONAL,

        bind-token           [1]  Token }

Token  ::=  SIGNED SEQUENCE {
        algorithm[0]  AlgorithmIdentifier

        name      [1]   DistinguishedName,

        time      [2]   UTCTime,

        random    [3]  BIT STRING }

Versions ::=  BIT STRING (v1988(0)}

DirectoryBindResult  ::=  DirectoryBindArgument
DirectoryBindError  ::=  SET {

        versions [0] Versions DEFAULT v1988,

        CHOICE {

                serviceError[1]ServiceProblem,

                securityError[2]SecurityProblem }}

DirectoryUnbind ::=  ABSTRACT-UNBIND
        FROM {readPort, searchPort, modifyPort }

-- operations, arguments, and results --

Read  ::=  ABSTRACT-OPERATION
        ARGUMENT   ReadArgument

        RESULT          ReadResult
```

```
PROTECTED, AlgorithmIdentifier
                FROM  AuthenticationFramework authenticationFramework
InvokeID,
                FROM  Remote-Operations-Notation {joint-iso-ccitt
                                                        remoteOperations(4) notation(0)};

-- macro for representing optional signing --
OPTIONALLY-SIGNED MACRO ::=
BEGIN
      TYPE NOTATION    ::=  type (Type)
      VALUE NOTATION ::=  value (VALUE CHOICE { Type, SIGNED Type})
END

-- objects and ports --
directory
      OBJECT
                      PORTS {readPort [S].
                             searchPort [S],
                             modifyPort [S]}
::=  id-ot-directory
dua                 OBJECT
                      PORTS { readPort [C],
                              searchPort [C]
                              modifyPort [C]}
::=  id-ot-dua
readPort
      PORT
                      CONSUMER INVOKES {
                             Read, Compare, Abandon}
::=  id-pt-read
searchPort
      PORT
                      CONSUMER INVOKES {
                             List, Search }
::=  id-pt-search
modifyPort
      PORT
                      CONSUMER INVOKES {
                             AddEntry, RemoveEntry,
                             ModifyEntry, ModifyRDN}
::=  id-pt-modify

-- bind and unbind --
```

This Annex is part of the standard.

This Annex includes all of the ASN.1 type, value and macro definitions contained in this Recommendation in the form of the ASN.1 module **DirectoryAbstractService**.

```
    DirectoryAbstractService {joint-ISO-CCITT ds(5) modules(1)  directoryAbstractService(2)}
DEFINITIONS ::=
BEGIN
EXPORTS
    directory, readPort, searchPort, modifyPort,
    DirectoryBind, DirectoryBindArgument,
    DirectoryUnbind,
    Read, ReadArgument, ReadResult,
    Abandon, AbandonArgument, AbandonResult,
    Compare, CompareArgument, CompareResult,
    List, ListArgument, ListResult,
    Search, SearchArgument, SearchResult,
    AddEntry, AddEntryArgument, AddEntryResult,
    RemoveEntry, RemoveEntryArgument, RemoveEntryResult,
    ModifyEntry, ModifyEntryArgument, ModifyEntryResult,
    ModifyRDN, ModifyRDNArgument, ModifyRDNResult,
    Abandoned, AbandonFailed, AttributeError, NameError,
    Referral, SecurityError, ServiceError, UpdateError,
    SecurityParameters;
IMPORTS
    informationFramework, authenticationFramework,
                distributedOperations, directoryObjectIdentifiers
        FROM  UsefulDefinitions {joint-iso-ccitt ds(5) modules(1)
                                usefulDefinitions(0)}
OBJECT, PORT, ABSTRACT-BIND, ABSTRACT-UNBIND,
ABSTRACT-OPERATION, ABSTRACT-ERROR
        FROM AbstractServiceNotation {joint-iso-ccitt mhs-motis(6)
                                                asdc(2) modules(0) notation(1) }
Attribute, AttributeType, AttributeValue, AttributeValueAssertion,
DistinguishedName, Name, RelativeDistinguishedName
        FROM  InformationFramework InformationFramework
id-ot-directory, id-ot-dua, id-pt-read, id-pt-search, id-pt-modify
        FROM  DirectoryObjectIdentifiers  directoryObjectIdentifiers
ContinuationReference, OperationProgress
        FROM  DistributedOperations distributedOperations
```

l)**ditError**: The Directory is unable to accomplish the request due to a DIT consistency problem.

12.9Update Error

12.9.1

An **UpdateError** reports problems related to attempts to add, delete, or modify information in the DIB.

```
UpdateError  ::=  ABSTRACT-ERROR
                  PARAMETER SET {

                        problem   [0]   UpdateProblem }

UpdateProblem      ::=   INTEGER {
                  namingViolation (1),

                  objectClassViolation (2),

                  notAllowedOnNonLeaf (3),

                  notAllowedOnRDN (4),

                  entryAlreadyExists (5),

                  affectsMultipleDSAs (6),

                  objectClassModificationProhibited (7) }
```

12.9.2

The error has a single **problem** parameter, which reports the particular **problem** encountered. The following problems may be indicated:

a)**namingViolation**: The attempted addition or modification would violate the structure rules of the DIT as defined in the Directory schema and Recommendation X.501. That is, it would place an entry as the subordinate of an alias entry, or in a region of the DIT not permitted to a member of its object class or would define an RDN for an entry to include a forbidden attribute type;

b) **objectClassViolation**: The attempted update would produce an entry inconsistent with the definition provided by its object class or with the definitions of Recommendation X.501 as they pertain to object classes;

c) **notAllowedOnNonLeaf**: The attempted operation is only allowed on leaf entries of the DIT;

d) **notAllowedOnRDN**: The attempted operation would affect the RDN (e.g. removal of an attribute which is a part of the RDN);

e) **entryAlreadyExists**: An attempted AddEntry operation names an entry which already exists;

f) **affectsMultipleDSAs**: An attempted update would need to operate on multiple DSAs, which is not permitted;

g) **objectClassModificationProhibited**: An operation attempted to modify the object class attribute.

Note - The **UpdateError** is not used to report problems with attribute types, values or constraint violations encountered in an **AddEntry**, **RemoveEntry**, **ModifyEntry** or **ModifyRDN** operation. Such problems are reported via an **AttributeError**.

ANNEX A

(to Recommendation X.511)

Abstract service in ASN.1