INTERNATIONAL STANDARD

ISO/IEC 9594-5

Fourth edition
2001-12-15

Information technology — Open Systems Interconnection — The Directory: Protocol specifications

Technologies de l'information — Interconnexion de systèmes ouverts (OSI) — L'annuaire: Spécification du protocole

Reference number
ISO/IEC 9594-5:2001(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.ch
Web www.iso.ch

Published by ISO in 2002

Printed in Switzerland

CONTENTS

Page

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

The main task of the joint technical committee is to prepare International Standards. 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.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 9594 may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Users and implementors should note the existence of a “defect resolution” procedure in ISO/IEC JTC 1 to identify and correct errors in International Standards through the publication of Technical Corrigenda. Identical corrections are made to the corresponding ITU-T Recommendations through Corrigenda and may also be made in the form of Implementors' Guides. Details of Technical Corrigenda to International Standards are available on the ISO website; published Technical Corrigenda can be obtained via the ISO webstore or from the ISO and IEC national bodies. Corrigenda and Implementors' Guides to ITU-T Recommendations can be obtained from the ITU-T website.

ISO/IEC 9594-5 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems*, in collaboration with ITU-T. The identical text is published as ITU-T Rec. X.519.

This fourth edition of ISO/IEC 9594-5 constitutes a technical revision of the third edition (ISO/IEC 9594-5:1998), which is provisionally retained in order to support implementations based on the third edition. This edition also incorporates Corrigendum 1:2002.

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: Public-key and attribute certificate frameworks*

— *Part 9: Replication*

— *Part 10: Use of systems management for administration of the Directory*

Annexes A to H form a normative part of this part of ISO/IEC 9594. Annex I is for information only.

Introduction

This Recommendation | International Standard, together with the other Recommendations | 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 that 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 | International Standard specifies the application service elements and application contexts for two protocols – the Directory Access Protocol (DAP) and the Directory System Protocol (DSP). The DAP provides for access to the Directory to retrieve or modify Directory information. The DSP provides for the chaining of requests to retrieve or modify Directory information to other parts of the distributed Directory System where the information may be held.

In addition this Recommendation | International Standard specifies the application service elements and application contexts for the Directory Information Shadowing Protocol (DISP) and the Directory Operational Binding Management Protocol (DOP). The DISP provides for the shadowing of information held in one DSA to another DSA. The DOP provides for the establishment, modification and termination of bindings between pairs of DSAs for the administration of relationships between the DSAs (such as for shadowing or hierarchical relationships).

This fourth edition technically revises and enhances, but does not replace, the third edition of this Recommendation | International Standard. Implementations may still claim conformance to the third edition. However, at some point, the third edition will not be supported (i.e. reported defects will no longer be resolved). It is recommended that implementations conform to this fourth edition as soon as possible.

This fourth edition specifies version 1 and version 2 of the Directory protocols.

The first and second editions specified only version 1. Most of the services and protocols specified in this edition are designed to function under version 1. However some enhanced services and protocols, e.g. signed errors, will not function unless all Directory entities involved in the operation have negotiated version 2. Whichever version has been negotiated, differences between the services and between the protocols defined in the four editions, except for those specifically assigned to version 2, are accommodated using the rules of extensibility defined in this edition of ITU-T Rec. X.519 | ISO/IEC 9594-5.

This Directory Specification also specifies an alternative version of the DAP, DSP, DISP and DOP protocols, known as DAP-IP, DSP-IP, DISP-IP and DOP-IP respectively, which are mappings of the corresponding abstract services directly onto the TCP/IP protocol instead of onto an OSI stack. These alternative protocols allow support of Directory service elements without the implementation overhead of supporting a full OSI stack.

Annex A, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for the directory access protocol.

Annex B, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for the directory system protocol.

Annex C, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for the directory information shadowing protocol.

Annex D, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for the directory operational binding management protocol.

Annex E, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for the IDM protocol specification.

Annex F, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for the Directory IDM protocols.

Annex G, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module which contains all the ASN.1 object identifiers assigned in this Recommendation | International Standard.

Annex H, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module which contains all the ASN.1 object identifiers assigned to identify operational binding types in this series of Recommendations | International Standards.

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

INTERNATIONAL STANDARD

ITU-T RECOMMENDATION

Information technology – Open Systems Interconnection – The Directory: Protocol specifications

1 Scope

This Recommendation | International Standard specifies the Directory Access Protocol, the Directory System Protocol, the Directory Information Shadowing Protocol, and the Directory Operational Binding Management Protocol fulfilling the abstract services specified in ITU-T Rec. X.511 | ISO/IEC 9594-3, ITU-T Rec. X.518 | ISO/IEC 9594-4, and ITU-T Rec. X.525 | ISO/IEC 9594-9.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. 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 | International Standard are encouraged to investigate the possibility of applying the most recent edition 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 | International Standards

- ITU-T Recommendation X.200 (1994) | ISO/IEC 7498-1:1994, *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model.*
- ITU-T Recommendation X.213 (1995) | ISO/IEC 8348:1996, *Information technology – Open Systems Interconnection – Network service definition.*
- ITU-T Recommendation X.214 (1995) | ISO/IEC 8072:1996, *Information technology – Open Systems Interconnection – Transport service definition.*
- ITU-T Recommendation X.215 (1995) | ISO/IEC 8326:1996, *Information technology – Open Systems Interconnection – Session service definition.*
- ITU-T Recommendation X.216 (1994) | ISO/IEC 8822:1994, *Information technology – Open Systems Interconnection – Presentation service definition.*
- ITU-T Recommendation X.217 (1995) | ISO/IEC 8649:1996, *Information technology – Open Systems Interconnection – Service definition for the Association Control Service Element.*
- ITU-T Recommendation X.227 (1995) | ISO/IEC 8650-1:1996, *Information technology – Open Systems Interconnection – Connection-oriented protocol for the Association Control Service Element: Protocol specification.*
- ITU-T Recommendation X.500 (2001) | ISO/IEC 9594-1:2001, *Information technology – Open Systems Interconnection – The Directory: Overview of concepts, models and services.*
- ITU-T Recommendation X.501 (2001) | ISO/IEC 9594-2:2001, *Information technology – Open Systems Interconnection – The Directory: Models.*
- ITU-T Recommendation X.509 (2000) | ISO/IEC 9594-8:2001, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks.*
- ITU-T Recommendation X.511 (2001) | ISO/IEC 9594-3:2001, *Information technology – Open Systems Interconnection – The Directory: Abstract service definition.*

- ITU-T Recommendation X.518 (2001) | ISO/IEC 9594-4:2001, *Information technology – Open Systems Interconnection – The Directory: Procedures for distributed operation.*
- ITU-T Recommendation X.520 (2001) | ISO/IEC 9594-6:2001, *Information technology – Open Systems Interconnection – The Directory: Selected attribute types.*
- ITU-T Recommendation X.521 (2001) | ISO/IEC 9594-7:2001, *Information technology – Open Systems Interconnection – The Directory: Selected object classes.*
- ITU-T Recommendation X.525 (2001) | ISO/IEC 9594-9:2001, *Information technology – Open Systems Interconnection – The Directory: Replication.*
- ITU-T Recommendation X.530 (2001) | ISO/IEC 9594-10:1998, *Information technology – Open Systems Interconnection – The Directory: Use of systems management for administration of the Directory.*
- ITU-T Recommendation X.680 (1997) | ISO/IEC 8824-1:1998, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*
- ITU-T Recommendation X.681 (1997) | ISO/IEC 8824-2:1998, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification.*
- ITU-T Recommendation X.682 (1997) | ISO/IEC 8824-3:1998, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*
- ITU-T Recommendation X.683 (1997) | ISO/IEC 8824-4:1998, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications.*
- ITU-T Recommendation X.690 (1997) | ISO/IEC 8825-1:1998, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).*
- ITU-T Recommendation X.880 (1994) | ISO/IEC 13712-1:1995, *Information technology – Remote Operations: Concepts, model and notation plus Technical Corrigendum 1 (1995).*
- ITU-T Recommendation X.880 (1994)/Amd.1 (1995) | ISO/IEC 13712-1:1995/Amd.1:1996, *Information technology – Remote Operations: Concepts, model and notation – Amendment 1: Built-in operations.*
- ITU-T Recommendation X.881 (1994) | ISO/IEC 13712-2:1995, *Information technology – Remote Operations: OSI realizations – Remote Operations Service Element (ROSE) service definition.*
- ITU-T Recommendation X.881 (1994)/Amd.1 (1995) | ISO/IEC 13712-2:1995/Amd.1:1996, *Information technology – Remote Operations: OSI realizations – Remote Operations Service Element (ROSE) service definition – Amendment 1: Mapping to A-UNIT-DATA and built-in operations.*
- ITU-T Recommendation X.882 (1994) | ISO/IEC 13712-3:1995, *Information technology – Remote Operations: OSI realizations – Remote Operations Service Element (ROSE) protocol specification plus Technical Corrigendum 1 (1995).*
- ITU-T Recommendation X.882 (1994)/Amd.1 (1995) | ISO/IEC 13712-3:1995/Amd.1:1996, *Information technology – Remote Operations: OSI realizations – Remote Operations Service Element (ROSE) protocol specification – Amendment 1: Mapping to A-UNIT-DATA and built-in operations.*

2.2 Paired Recommendations | International Standards equivalent in technical content

- ITU-T Recommendation X.218 (1993) *Reliable Transfer: Model and Service Definition.*

  ISO/IEC 9066-1:1989, *Information processing systems – Text communication – Reliable Transfer – Part 1: Model and service definition.*

2.3 ISO/IEC Standards

- ISO/IEC 10646-1:2000, *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1 – Architecture and Basic Multilingual Plane.*

2.4 Other references

- ITU-T Recommendation E.164 (1997), *The international public telecommunication numbering plan*.
- ITU-T Recommendation X.121 (2000), *International numbering plan for public data networks*.
- IETF RFC 2025 (1996), *The Simple Public-Key GSS-API Mechanism (SPKM)*.

- IETF RFC 793 (1981), *Transmission control protocol – DARPA Internet Program – Protocol Specification.*
- IETF RFC 1277 (1991), *Encoding Network Addresses to support operation over non-OSI lower layers.*
- IETF RFC 1738 (1994), *Uniform Resource Locators (URL).*

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply:

3.1 OSI Reference Model Definitions

The following terms are defined in ITU-T Rec. X.200 | ISO/IEC 7498-1:

a) *abstract-syntax;*
b) *application-context*;
c) *application-entity*;
d) *application process*;
e) *application-protocol-control-information*;
f) *application-protocol-data-unit*;
g) *application-service-element*.

3.2 Remote Operations Definitions

The following terms are defined in ITU-T Rec. X.880 | ISO/IEC 13712-1:

a) *connection package*;
b) *contract, association contract*;
c) *error*;
d) *operation*;
e) *operation package*;
f) *ROS-object*.

3.3 Basic Directory Definitions

The following terms are defined in ITU-T Rec. X.501 | ISO/IEC 9594-2:

a) *the Directory*;
b) *(Directory) user*;
c) *Directory System Agent (DSA)*;
d) *Directory User Agent (DUA)*.

3.4 Distributed Operation Definitions

The following terms are defined in ITU-T Rec. X.518 | ISO/IEC 9594-4:

a) *chaining*;
b) *referral*.

3.5 Upper layer security definitions

The following terms are used as defined in ITU-T Rec. X.803 | ISO/IEC 10745:

a) *security association;*
b) *security transformation;*
c) *security exchange;*
d) *security exchange item.*

The following terms are used as defined in ITU-T Rec. X.830 | ISO/IEC 11586-1:

a) *protecting transfer syntax;*

b) *protection mapping.*

4 Abbreviations

For the purposes of this Recommendation | International Standard, the following abbreviations apply:

AC	Application Context
ACSE	Association Control Service Element
AE	Application Entity
APCI	Application Protocol Control Information
APDU	Application Protocol Data Unit
ASE	Application Service Element
DAP	Directory Access Protocol
DISP	Directory Information Shadowing Protocol
DOP	Directory Operational Binding Management Protocol
DSA	Directory System Agent
DSP	Directory System Protocol
DUA	Directory User Agent
GULS	Generic Upper Layers Security
ROS	Remote Operations Service
ROSE	Remote Operations Service Element
RTSE	Reliable Transfer Service Element

5 Conventions

With minor exceptions this Directory Specification has been prepared according to the "Rules for presentation of ITU-T | ISO/IEC common text" in the Guide for ITU-T and ISO/IEC JTC 1 Cooperation, October 1996.

The term "Directory Specification" (as in "this Directory Specification") shall be taken to mean ITU-T Rec. X.519 | ISO/IEC 9594-5. 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 first edition of the Directory Specifications, i.e. the 1988 edition of the series of CCITT X.500 Recommendations and the ISO/IEC 9594:1990 edition. This Directory Specification uses the term "1993 edition systems" to refer to systems conforming to the second (1993) edition of the Directory Specifications, i.e. the 1993 edition of the series of ITU-T X.500 Recommendations and the ISO/IEC 9594:1995 edition. This Directory Specification uses the term "1997 edition systems" to refer to systems conforming to the third edition of the Directory Specifications, i.e. the 1997 edition of the series of ITU-T X.500 Recommendations and the ISO/IEC 9594:1998 edition. This Directory Specification uses the term "4th edition systems" to refer to systems conforming to this fourth edition of the Directory Specifications, i.e. the 2001 editions of ITU-T X.500, X.501, X.511, X.518, X.519, X.520, X.521, X.525, and X.530, the 2000 edition of ITU-T X.509, and parts 1-0 of the ISO/IEC 9594:2001 edition.

This Directory Specification presents ASN.1 notation in the bold Helvetica typeface. When ASN.1 types and values are referenced in normal text, they are differentiated from normal text by presenting them in the bold Helvetica typeface. The names of procedures, typically referenced when specifying the semantics of processing, are differentiated from normal text by displaying them in bold Times. Access control permissions are presented in italicized Times.

If the items in a list are numbered (as opposed to using "–" or letters), then the items shall be considered steps in a procedure.

This Directory Specification defines directory operations using the Remote Operation notation defined in ITU-T Rec. X.880 | ISO/IEC 13712-1.

6 OSI protocol overview

6.1 Remote Operations – Specification and OSI Realization

ITU-T Rec. X.880 | ISO/IEC 13712-1 defines several information object classes that are useful in the specification of ROS-based application protocols such as the various Directory protocols defined in this Directory Specification. A number of these classes are used in this and subsequent clauses. The specification techniques provided in ITU- Rec. X.880 | ISO/IEC 13712-1 are used to define a generic protocol between objects. When realized as an OSI application layer protocol, the concepts of ITU-T Rec. X.880 | ISO/IEC 13712-1 are mapped to OSI concepts in ITU-T Rec. X.881 | ISO/IEC 13712-2 and ITU-T Rec. X.882 | ISO/IEC 13712-3.

The **ROS-OBJECT-CLASS** class is used to define a set of common capabilities of a set of ROS-objects in terms of the (association) contracts they support as initiators and/or responders. When realized using the communication services of OSI, a ROS-object maps to an application process and a contract to an application context. In these Directory Specifications the term abstract service is used to refer to a ROS association contract and OSI application layer protocol to refer to the realization of a contract between two open systems using OSI communication services.

The **OPERATION-PACKAGE** class is used to define a set of operations which may be invoked by a ROS-object assuming the role of "consumer", the operations which may be invoked by a ROS-object assuming the role of "supplier", and the operations which may be invoked by both ROS-objects. When using the communication services of OSI, an operation package is realized as an application service element (ASE).

The **CONNECTION-PACKAGE** class is used to define the bind and unbind operations used to establish and release an association. When realized using the communication services of OSI, a connection package is realized as the procedures that use the services of the Association Control Service Element.

The **CONTRACT** class is used to define an association contract in terms of a connection package and one or more operation packages. When specifying the contract, the packages in which the association initiator assumes the role of consumer, the association responder assumes the role of consumer, and either may assume the role of consumer are identified. When using the communication services of OSI, a contract is realized as an application context.

The **APPLICATION-CONTEXT** class is used to define the static aspects of an application context. These include the contract that is realized via the application context, the OSI service that establishes and releases the association, the OSI service that provides information transfer for the interactions of the contract, and the abstract syntaxes used.

The **ABSTRACT-SYNTAX** class, which is built in to ASN.1, is used to define and assign an object identifier to an ASN.1 type whose values comprise an abstract syntax.

The OSI application layer protocols defined in the Directory Specifications, the DAP, DSP, DISP and DOP, are protocols to provide communication between a pair of application processes. In the OSI environment this is represented as communication between a pair of Application-Entities (AEs) using the presentation service. The function of an AE is provided by a set of Application-Service-Elements (ASEs). The interaction between AEs is described in terms of their use of the services provided by the ASEs. All the services provided by the Directory ASEs are contained in a single AE.

The Remote Operations Service Element (ROSE) supports the request/reply paradigm of the operation. The Directory ASEs provide the mapping function of the abstract-syntax notation of the directory operation packages onto the services provided by the ROSE.

The Association Control Service Element (ACSE) supports the establishment and release of an application-association between a pair of AEs. Associations between a DUA and a DSA may be established only by the DUA. Only the initiator of an established association can release it.

Optionally, the Reliable Transfer Service Element (RTSE) may be used to reliably transfer the Application Protocol Data Units (APDUs) of the DISP.

6.2 Directory ROS-Objects and Contracts

ITU-T Rec. X.511 | ISO/IEC 9594-3 defines the abstract service between a DUA and the Directory which provides an access point to support a user accessing Directory services.

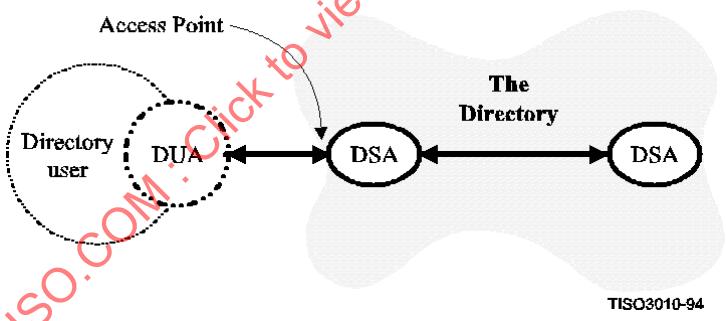
The **dua** class of ROS-object describes a DUA, being an instance of this class, as the initiator of the contract **dapContract**. This contract is referred to in these Directory Specifications as the Directory Abstract Service. It is specified as a ROS-based information object in 6.3.

```
dua ROS-OBJECT-CLASS ::= {
        INITIATES       { dapContract }
        ID              id-rosObject-dua }
```

The **directory** class of ROS-object describes the provider of the Directory Abstract Service. This provider is the responder of the **dapContract**.

```
directory ROS-OBJECT-CLASS ::= {
        RESPONDS        { dapContract }
        ID              id-rosObject-directory }
```

The Directory is further modelled, as depicted in Figure 1, as being represented to a DUA by a DSA which supports the particular access point concerned. ITU-T Rec. X.518 | ISO/IEC 9594-4 defines the interactions between a pair of DSAs within the Directory to support user requests which are chained.



The **directory** object is therefore manifested as a set interacting DSAs. Each DSA comprising the **directory** is an instance of the **dap-dsa** class. A **dap-dsa** object assumes the role of responder in the **dapContract**.

```
dap-dsa ROS-OBJECT-CLASS ::= {
        RESPONDS        { dapContract }
        ID              id-rosObject-dapDSA }
```

In addition to interacting with DUAs, DSAs interact with one another to achieve various objectives. In what follows, a number of contracts and ROS-objects expressing how DSAs participate in these contracts are defined. Any real DSA may instantiate one or more of these DSA ROS-objects.

The interactions between DSAs generally required to provide the Directory Abstract Service in the presence of a distributed DIB are defined as a **dspContract**. A DSA that participates in this contract is defined as a ROS-object of class **dsp-dsa**. The contract is referred to in these Directory Specifications as the DSA Abstract Service. It is specified as a ROS-based information object in 6.4.

```
dsp-dsa ROS-OBJECT-CLASS ::= {
        BOTH        { dspContract }
        ID              id-rosObject-dspDSA }
```

The Shadow Abstract Service specifies the shadowing of information between a shadow supplier and a shadow consumer DSA. This service is manifested in two forms and therefore is defined as two distinct contracts. They are specified as a ROS-based information objects in 6.5.

The **shadowConsumerContract** expresses the form of the service in which the shadow consumer, a ROS-object of class **initiating-consumer-dsa**, initiates the contract. A ROS-object of class **responding-supplier-dsa**, responds in this contract.

```
initiating-consumer-dsa  ROS-OBJECT-CLASS  ::= {
        INITIATES          { shadowConsumerContract }
        ID                 id-rosObject-initiatingConsumerDSA }

responding-supplier-dsa  ROS-OBJECT-CLASS  ::= {
        RESPONDS           { shadowConsumerContract }
        ID                 id-rosObject-respondingSupplierDSA }
```

The **shadowSupplierContract** expresses the form of the service in which the shadow supplier, a ROS-object of class **initiating-supplier-dsa**, initiates the contract. A ROS-object of class **responding-consumer-dsa**, responds in this contract.

```
initiating-supplier-dsa  ROS-OBJECT-CLASS  ::= {
        INITIATES          { shadowSupplierContract }
        ID                 id-rosObject-initiatingSupplierDSA }

responding-consumer-dsa  ROS-OBJECT-CLASS  ::= {
        RESPONDS           { shadowSupplierContract }
        ID                 id-rosObject-respondingConsumerDSA }
```

The interactions between two DSAs to manage a set of operational bindings are defined as a **dopContract**.

```
dop-dsa  ROS-OBJECT-CLASS  ::= {
        BOTH       { dopContract }
        ID                 id-rosObject-dopDSA }
```

A DSA that participates in this contract is defined as a ROS-object of class **dop-dsa**. This contract is specified as a ROS-based information object in 6.6.

6.3 DAP Contract and Packages

The **dapContract** is defined as an information object of class **CONTRACT**.

```
dapContract  CONTRACT  ::= {
        CONNECTION                 dapConnectionPackage
        INITIATOR CONSUMER OF      { readPackage | searchPackage | modifyPackage }
        ID                         id-contract-dap }
```

When a DUA and DSA from different open systems interact, this association contract may be realized as an OSI application layer protocol, referred to in these Directory Specifications as the Directory Access Protocol (DAP). The definition of this protocol in terms of an OSI application context is provided in 7.2.

The **dapContract** is composed of a connection package, **dapConnectionPackage**, and three operation packages, **readPackage**, **searchPackage** and **modifyPackage**.

The connection package, **dapConnectionPackage**, is defined as an information object of class **CONNECTION-PACKAGE**. The bind and unbind operations of this connection package, **directoryBind** and **directoryUnbind**, are defined in ITU-T Rec. X.511 | ISO/IEC 9594-3.

```
dapConnectionPackage  CONNECTION-PACKAGE  ::= {
        BIND       directoryBind
        UNBIND     directoryUnbind
        ID         id-package-dapConnection }
```

The operation packages, **readPackage**, **searchPackage** and **modifyPackage**, are defined as information objects of class **OPERATION-PACKAGE**. The operations of these operation packages are defined in ITU-T Rec. X.511 | ISO/IEC 9594-3.

```
readPackage  OPERATION-PACKAGE  ::=  {
        CONSUMER INVOKES        { read | compare | abandon }
        ID                      id-package-read }

searchPackage  OPERATION-PACKAGE  ::=  {
        CONSUMER INVOKES        { list | search }
        ID                      id-package-search }

modifyPackage  OPERATION-PACKAGE  ::=  {
        CONSUMER INVOKES        { addEntry | removeEntry | modifyEntry | modifyDN }
        ID                      id-package-modify }
```

NOTE – These packages, when realized as ASEs, are used for the construction of application contexts defined in this Directory Specification. They are not intended to allow for claims of conformance to individual, or other combinations of, ASEs.

Since the DUA is the initiator of the **dapContract**, it assumes the role of consumer of the operation packages of the contract. This means that only the DUA can invoke operations in this contract and its OSI realization.

6.4 DSP Contract and Packages

The **dspContract** is defined as an information object of class **CONTRACT**.

```
dspContract  CONTRACT  ::=  {
        CONNECTION              dspConnectionPackage
        OPERATIONS OF           { chainedReadPackage | chainedSearchPackage | chainedModifyPackage }
        ID                      id-contract-dsp }
```

When a pair of DSAs from different open systems interact, this association contract is realized as an OSI application layer protocol, referred to in these Directory Specifications as the Directory System Protocol (DSP). The definition of this protocol in terms of an OSI application context is provided in 7.2.

The **dspContract** is composed of a connection package, **dspConnectionPackage**, and three operation packages, **chainedReadPackage**, **chainedSearchPackage** and **chainedModifyPackage**.

The connection package, **dspConnectionPackage**, is defined as an information object of class **CONNECTION-PACKAGE**. It is identical to the connection package, **dapConnectionPackage**.

```
dspConnectionPackage  CONNECTION-PACKAGE  ::=  {
        BIND        dSABind
        UNBIND      dSAUnbind
        ID          id-package-dspConnection }
```

The operation packages, **chainedReadPackage**, **chainedSearchPackage** and **chainedModifyPackage**, are defined as information objects of class **OPERATION-PACKAGE**. The operations of these operation packages are defined in ITU-T Rec. X.518 | ISO/IEC 9594-4.

```
chainedReadPackage  OPERATION-PACKAGE  ::=  {
        OPERATIONS      { chainedRead | chainedCompare | chainedAbandon }
        ID              id-package-chainedRead }

chainedSearchPackage  OPERATION-PACKAGE  ::=  {
        OPERATIONS      { chainedList | chainedSearch }
        ID              id-package-chainedSearch }

chainedModifyPackage  OPERATION-PACKAGE  ::=  {
        OPERATIONS      { chainedAddEntry | chainedRemoveEntry |
                        chainedModifyEntry | chainedModifyDN }
        ID              id-package-chainedModify }
```

NOTE – These packages, when realized as ASEs, are used for the construction of application contexts defined in this Directory Specification. They are not intended to allow for claims of conformance to individual, or other combinations of, ASEs.

In the **dspContract** either DSA may assume the role of initiator and either the initiating or responding DSA may invoke the operations of the contract.

6.5 DISP Contracts and Packages

The **shadowConsumerContract** and **shadowSupplierContract** are defined as information objects of class **CONTRACT**.

```
shadowConsumerContract  CONTRACT  ::= {
        CONNECTION                  dispConnectionPackage
        INITIATOR CONSUMER OF       { shadowConsumerPackage }
        ID                          id-contract-shadowConsumer }
shadowSupplierContract  CONTRACT  ::= {
        CONNECTION                  dispConnectionPackage
        RESPONDER CONSUMER OF       { shadowSupplierPackage }
        ID                          id-contract-shadowSupplier }
```

NOTE – The term consumer and supplier are employed in the notation for the **CONTRACT** and **OPERATION-PACKAGE** classes are used to designate two roles. These roles correspond to the two terms shadow consumer and shadow supplier, respectively, used in ITU-T Rec. X.525 | ISO/IEC 9594-9.

The OSI realizations of the two forms of the Shadow Abstract Service, referred to collectively as the Directory Information Shadowing Protocol (DISP), are defined in terms of several OSI application contexts, provided in 7.2.

The **shadowConsumerContract** and **shadowSupplierContract** are composed of a common connection package, **dispConnectionPackage**, and one operation package, either **shadowConsumerPackage** in the first case or **shadowSupplierPackage** in the second.

The connection package, **dispConnectionPackage**, is defined as an information object of class **CONNECTION-PACKAGE**. It is identical to the connection package, **dapConnectionPackage**.

```
dispConnectionPackage  CONNECTION-PACKAGE  ::= {
        BIND                    dSAShadowBind
        UNBIND                  dSAShadowUnbind
        ID                      id-package-dispConnection }
```

The operation packages, **shadowConsumerPackage** and **shadowSupplierPackage**, are defined as information objects of class **OPERATION-PACKAGE**. The operations of these operation packages are defined in ITU-T Rec. X.525 | ISO/IEC 9594-9.

```
shadowConsumerPackage  OPERATION-PACKAGE  ::= {
        CONSUMER INVOKES    { requestShadowUpdate }
        SUPPLIER INVOKES    { updateShadow }
        ID                  id-package-shadowConsumer }

shadowSupplierPackage  OPERATION-PACKAGE  ::= {
        SUPPLIER INVOKES    { coordinateShadowUpdate |
                              updateShadow }
        ID                  id-package-shadowSupplier }
```

Since the shadow consumer is the initiator of the **shadowConsumerContract**, it assumes the role of consumer of the **shadowConsumerPackage**. This means that the shadow consumer invokes the **requestShadowUpdate** operation and that the shadow supplier invokes the **updateShadow** operation.

Since the shadow supplier is the initiator of the **shadowSupplierContract**, it assumes the role of supplier of the **shadowSupplierPackage**. This means that the shadow supplier invokes the operations of the contract.

6.6 DOP Contract and Packages

The **dopContract** is defined as an information object of class **CONTRACT**.

```
dopContract  CONTRACT  ::=  {
        CONNECTION              dopConnectionPackage
        OPERATIONS OF           { dopPackage }
        ID                      id-contract-dop }
```

When a pair of DSAs from different open systems interact, this association contract is realized as an OSI application layer protocol, referred to in these Directory Specifications as the Directory Operational Binding Management Protocol (DOP). The definition of this protocol in terms of an OSI application context is provided in 7.2.

The connection package, **dopConnectionPackage**, is defined as an information object of class **CONNECTION-PACKAGE**. It is identical to the connection package, **dapConnectionPackage**.

```
dopConnectionPackage  CONNECTION-PACKAGE  ::=  {
        BIND                dSAOperationalBindingManagementBind
        UNBIND              dSAOperationalBindingManagementUnbind
        ID                  id-package-dopConnection }
```

The operation package, **dopPackage**, is defined as information objects of class **OPERATION-PACKAGE**. The operations of these operation packages are defined in ITU-T Rec. X.501 | ISO/IEC 9594-2.

```
dopPackage  OPERATION-PACKAGE  ::=  {
        CONSUMER INVOKES    { establishOperationalBinding |
                              modifyOperationalBinding |
                              terminateOperationalBinding }
        ID                  id-package-operationalBindingManagement }
```

The DSA that may assume the role of initiator of the **dopContract**, depends on the DSA roles assigned for the operational binding(s) to be managed using the operations of this contract. Only the initiator may invoke the operations of the **dopContract**. More than one operational binding type may be managed with this contract only if the DSA roles for the distinct types are compatible (e.g. a DSA assumes Role A for each binding type).

6.7 Use of underlying services

The DAP, DSP, DOP and DISP protocols make use of underlying services as described below.

6.7.1 Use of ROSE services

The Remote Operations Service Element (ROSE) is defined ITU-T Rec. X.881 | ISO/IEC 13712-2.

The ROSE supports the request/reply paradigm of remote operations.

The Directory ASEs are users of the RO-INVOKE, RO-RESULT, RO-ERROR, RO-REJECT-U and RO-REJECT-P services of the ROSE.

The remote operations of the DAP and the DSP are asynchronous. Note that, as the DUA is a consumer of the DAP, it may choose to operate in a synchronous manner.

The remote operations of the DISP shall be supported as synchronous operations and may optionally be supported as asynchronous operations.

The remote operations of the DOP are asynchronous.

6.7.2 Use of RTSE services

The Reliable Transfer Service Element (RTSE) is defined in ITU-T Rec. X.218 | ISO/IEC 9066-1.

The RTSE provides for the reliable transfer of Application Protocol Data Units (APDUs). The RTSE ensures that each APDU is completely transferred exactly once, or that the sender is warned of an exception. The RTSE recovers from communication and end-system failure and minimizes the amount of retransmission needed for recovery.

Alternative application contexts with and without RTSE are defined to support the DISP.

The RTSE is used in normal mode. The use of the normal mode of the RTSE implies the use of the normal mode of the ACSE and the normal mode of the Presentation Service.

If the RTSE is included in an application context, the RO-BIND service maps onto the RT-OPEN service of the RTSE and the RO-UNBIND service maps onto the RT-CLOSE service of the RTSE. The basic ROSE services are the sole user of the RT-TRANSFER, RT-TURN-PLEASE, RT-TURN-GIVE, RT-P-ABORT and RT-U-ABORT services of the RTSE.

6.7.3 Use of ACSE services

The Association Control Service Element (ACSE) is defined in ITU-T Rec. X.217 | ISO/IEC 8649.

The ACSE provides for the control (establishment, release, abort) of application-associations between AEs.

If the RTSE is included in an application context, the RTSE is the sole user of the A-ASSOCIATE, A-RELEASE, A-ABORT and A-P-ABORT services of the ACSE.

If the RTSE is not included in an application context, the RO-BIND and RO-UNBIND services are the sole users of the A-ASSOCIATE and A-RELEASE services of the ACSE. The application-process is the user of the A-ABORT and A-P-ABORT services of the ACSE.

The receipt of an A-ABORT or A-P-ABORT on an association supporting the DAP terminates all request processing. Except for certain conditions described in ITU-T Rec. X.518 | ISO/IEC 9594-4, this is also true for the DSP. It is a Directory user responsibility to confirm if requested modifications to the DIB occurred.

The receipt of an A-ABORT or A-P-ABORT on an association supporting the DISP is described in ITU-T Rec. X.525 | ISO/IEC 9594-9.

The receipt of an A-ABORT or A-P-ABORT on an association supporting the DOP is described in ITU-T Rec. X.518 | ISO/IEC 9594-4.

6.7.4 Use of the presentation service

The presentation service is defined in ITU-T Rec. X.216 | ISO/IEC 8822.

The Presentation Layer coordinates the representation (syntax) of the Application Layer semantics that are to be exchanged.

In normal mode, a different presentation-context is used for each abstract-syntax included in the application-context.

The ACSE is the sole user of the P-CONNECT, P-RELEASE, P-U-ABORT and P-P-ABORT services of the presentation service.

If the RTSE is included in an application context, the RTSE is the sole user of the P-ACTIVITY-START, P-ACTIVITY-END, P-ACTIVITY-INTERRUPT, P-ACTIVITY-DISCARD, P-ACTIVITY-RESUME, P-DATA, P-MINOR-SYNCHRONIZE, P-U-EXCEPTION-REPORT, P-P-EXCEPTION-REPORT, P-TOKEN-PLEASE and P-CONTROL-GIVE services of the presentation service.

If the RTSE is not included in an application context, the ROSE is the sole user of the P-DATA service of the presentation service.

Presentation default context, context restoration, and context management are not used.

6.7.5 Use of Lower Layer Services

(This subclause applies to ITU-T Rec. X.519 only and not to ISO/IEC 9594-5.)

The session-service is defined in ITU-T Rec. X.215 | ISO/IEC 8326. The Session Layer structures the dialogue of the flow of information between the end-systems.

If the RTSE is included in an application context, the Kernel, Half-duplex, Exceptions, Minor-synchronize and Activity Management functional units of the Session Service are used by the Presentation Layer.

If the RTSE is not included in the application context, the Kernel and Duplex functional units of the Session Service are used by the Presentation Layer.

The transport-service is defined in ITU-T Rec. X.214 | ISO/IEC 8072. The Transport Layer provides for the end-to-end transparent transfer of data over the underlying network connection.

The choice of the class of transport-service used by the Session Layer depends on the requirements for multiplexing and error recovery. Support for Transport Class 0 (non-multiplexing) is mandatory. Transport Expedited Service is not used.

Support for other classes is optional. A multiplexing class may be used to multiplex the DAP or DSP and other protocols over the same network connection. An error recovery class may be chosen over a network connection with an unacceptable residual error rate.

An underlying network supporting the OSI network-service defined in ITU-T Rec. X.213 | ISO/IEC 8348 is assumed.

A network-address is as defined in ITU-T Recs. X.121, E.164, or ITU-T Rec. X.213 | ISO/IEC 8348 (OSI NSAP-address).

7 Directory protocol OSI abstract syntax

7.1 Abstract syntaxes

Two abstract syntaxes used in the Directory protocols are specified elsewhere. The abstract-syntax of ACSE, **acse-abstract-syntax**, is needed to establish the associations. The abstract-syntax of RTSE, **rtse-abstract-syntax**, is optionally needed for the DISP.

The ASN.1 type from which the values of the abstract syntaxes are derived is specified using the parameterized types **ROS {}**, **Bind {}**, and **Unbind {}** which are defined in ITU-T Rec. X.880 | ISO/IEC 13712-1.

These abstract syntaxes and those specified below shall (as a minimum) be encoded according to the Basic ASN.1 encoding rules.

NOTE – The abstract syntaxes defined in this clause that import from module **DirectoryShadowAbstractService** will use a mixture of implicit and explicit tags.

7.1.1 DAP Abstract Syntax

The Directory ASEs that realize the operation packages specified in 6.3 share a single abstract syntax, **directoryAccessAbstractSyntax**. This is specified as an information object of the class **ABSTRACT-SYNTAX**.

```
directoryAccessAbstractSyntax  ABSTRACT-SYNTAX  ::=  {
        DAP-PDUs
        IDENTIFIED BY    id-as-directoryAccessAS }

DAP-PDUs  ::=  CHOICE  {
        basicRos    ROS { { DAP-InvokeIDSet }, { DAP-Invokable }, { DAP-Returnable } },
        bind        Bind { directoryBind },
        unbind      Unbind { directoryUnbind } }

DAP-InvokeIDSet  ::=  InvokeId (ALL EXCEPT absent:NULL)

DAP-Invokable  OPERATION  ::=    { read | compare | abandon |
                                   list | search |
                                   addEtry | removeEntry | modifyEntry | modifyDN }

DAP-Returnable  OPERATION  ::=   { read | compare | abandon |
                                   list | search |
                                   addEntry | removeEntry | modifyEntry | modifyDN }
```

7.1.2 DSP Abstract Syntax

The Directory ASEs that realize the operation packages specified in 6.4 share a single abstract syntax, **directorySystemAbstractSyntax**. This is specified as an information object of the class **ABSTRACT-SYNTAX**.

```
directorySystemAbstractSyntax  ABSTRACT-SYNTAX  ::=  {
        DSP-PDUs
        IDENTIFIED BY    id-as-directorySystemAS }
```

```
DSP-PDUs ::= CHOICE {
        basicRos    ROS { {DSP-InvokeIDSet }, { DSP-Invokable }, { DSP-Returnable } },
        bind        Bind { dSABind },
        unbind      Unbind { dSAUnbind } }

DSP-InvokeIDSet ::= InvokeId (ALL EXCEPT absent:NULL)

DSP-Invokable OPERATION ::=     { chainedRead | chainedCompare | chainedAbandon |
                                  chainedList | chainedSearch |
                                  chainedAddEntry | chainedRemoveEntry | chainedModifyEntry |
                                  chainedModifyDN }

DSP-Returnable OPERATION ::=    { chainedRead | chainedCompare | chainedAbandon |
                                  chainedList | chainedSearch |
                                  chainedAddEntry | chainedRemoveEntry | chainedModifyEntry |
                                  chainedModifyDN }
```

7.1.3 DISP Abstract Syntax

The Directory ASEs that realize the operation packages specified in 6.5 employ either the abstract syntax **directoryShadowAbstractSyntax** or **directoryReliableShadowAbstractSyntax**, depending on whether RTSE is not or is used in the application context. These two abstract syntaxes are specified as information objects of the class **ABSTRACT-SYNTAX**.

```
directoryShadowAbstractSyntax ABSTRACT-SYNTAX ::= {
        DISP-PDUs
        IDENTIFIED BY    id-as-directoryShadowAS }

directoryReliableShadowAbstractSyntax ABSTRACT-SYNTAX ::= {
        Reliable-DISP-PDUs
        IDENTIFIED BY    id-as-directoryReliableShadowAS }
```

In addition, the following abstract syntax is used in the contexts employing RTSE. It is comprised of the abstract syntax of RTSE itself and the abstract syntax of **Bind { dSAShadowBind }**, and **Unbind { dSAShadowUnbind }**.

```
reliableShadowBindingAbstractSyntax    ABSTRACT-SYNTAX ::= {
        ReliableShadowBinding-PDUs
        IDENTIFIED BY    id-as-reliableShadowBindingAS }
```

The ASN.1 types from which the values of the abstract syntaxes are derived are specified using the **ROS {}**, **Bind {}**, and **Unbind {}** parameterized types.

```
DISP-PDUs ::= CHOICE {
        basicROS    ROS { { DISP-InvokeIDSet }, { DISP-Invokable }, {DISP-Returnable } },
        bind        Bind { dSAShadowBind },
        unbind      Unbind { dSAShadowUnbind } }

Reliable-DISP-PDUs ::= ROS { { DISP-InvokeIDSet }, { DISP-Invokable },
                {DISP-Returnable } }

ReliableShadowBinding-PDUs ::= CHOICE {
        rTS         [0] RTSE-apdus,
        bind        Bind { dSAShadowBind },
        unbind      Unbind { dSAShadowUnbind } }

DISP-InvokeIDSet ::= InvokeId (ALL EXCEPT absent:NULL)

DISP-Invokable OPERATION ::=    { requestShadowUpdate | updateShadow |
                                  coordinateShadowUpdate }

DISP-Returnable OPERATION ::=   { requestShadowUpdate | updateShadow |
                                  coordinateShadowUpdate }
```

7.1.4 DOP Abstract Syntax

The Directory ASE that realizes the operation package specified in 6.6 employs the abstract syntax, **directory-OperationalBindingManagementAbstractSyntax**. This is specified as an information object of the class **ABSTRACT-SYNTAX**.

```
directoryOperationalBindingManagementAbstractSyntax  ABSTRACT-SYNTAX  ::=  {
        DOP-PDUs
        IDENTIFIED BY    id-as-directoryOperationalBindingManagementAS }

DOP-PDUs  ::=  CHOICE  {
        basicRos    ROS { { DOP-InvokeIDSet }, { DOP-Invokable }, { DOP-Returnable } },
        bind        Bind { directoryBind },
        unbind      Unbind { directoryUnbind } }

DOP-InvokeIDSet  ::=  InvokeId (ALL EXCEPT absent:NULL)

DOP-Invokable  OPERATION  ::=    { establishOperationalBinding |
                                   modifyOperationalBinding |
                                   terminateOperationalBinding }

DOP-Returnable  OPERATION  ::=   { establishOperationalBinding |
                                   modifyOperationalBinding |
                                   terminateOperationalBinding }
```

7.2 Directory application contexts

7.2.1 Directory Access Application Context

The **dapContract** is realized as the **directoryAccessAC**. This application context is specified as an information object of the class **APPLICATION-CONTEXT**.

```
directoryAccessAC  APPLICATION-CONTEXT  ::=  {
        CONTRACT                        dapContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directoryAccessAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-directoryAccessAC }
```

7.2.2 Directory System Application Context

The **dspContract** is realized as the **directorySystemAC**. This application context is specified as an information object of the class **APPLICATION-CONTEXT**.

```
directorySystemAC  APPLICATION-CONTEXT  ::=  {
        CONTRACT                        dspContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directorySystemAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-directorySystemAC }
```

7.2.3 Directory Shadow Application Contexts

If a DSA supports the DISP, that DSA shall support at least one of the shadow supplier roles or shadow consumer roles and at least one of the **shadowSupplierInitiatedACs** or the **shadowConsumerInitiatedAC**. If a DSA supports the **shadowSupplierInitiatedAC** for a particular role, it may also optionally support the **reliableShadowSupplierInitiated-AC** for the same role. If a DSA supports the **shadowConsumerInitiatedAC** for a particular role, it may also optionally support the **reliableShadowConsumerInitiatedAC** for the same role.

7.2.3.1 Shadow Supplier Initiated Contexts

The **shadowSupplierContract** may be realized as the **shadowSupplierInitiatedAC**. This application context is specified as an information object of the class **APPLICATION-CONTEXT**.

```
shadowSupplierInitiatedAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowSupplierContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directoryShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-shadowSupplierInitiatedAC }
```

This application context requires that only synchronous operations be employed.

The **shadowSupplierInitiatedAsynchronousAC** variant of this application context permits the use of asynchronous operations.

```
shadowSupplierInitiatedAsynchronousAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowSupplierContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directoryShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-shadowSupplierInitiatedAsynchronousAC }
```

The **shadowSupplierContract** may optionally be realized as the **reliableShadowSupplierInitiatedAC**. This application context is specified as an information object of the class **APPLICATION-CONTEXT**.

```
reliableShadowSupplierInitiatedAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowSupplierContract
        ESTABLISHED BY                  association-by-RTSE
        INFORMATION TRANSFER BY         transfer-by-RTSE
        ABSTRACT SYNTAXES               { acse-abstract-syntax |
                                        reliableShadowBindingAbstractSyntax |
                                        directoryReliableShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-reliableShadowSupplierInitiatedAC }
```

7.2.3.2 Shadow Consumer Initiated Contexts

The **shadowConsumerContract** may be realized as the **shadowConsumerInitiatedAC**. This application context is specified as an information object of the class **APPLICATION-CONTEXT**.

```
shadowConsumerInitiatedAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowConsumerContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directoryShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-shadowConsumerInitiatedAC }
```

This application context requires that only synchronous operations be employed.

The **shadowConsumerInitiatedAsynchronousAC** variant of this application context permits the use of asynchronous operations.

```
shadowConsumereInitiatedAsynchronousAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowConsumerContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directoryShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-shadowConsumerInitiatedAsynchronousAC }
```

The **shadowConsumerContract** may optionally be realized as the **reliableShadowConsumerInitiatedAC**. This application context is specified as an information object of the class **APPLICATION-CONTEXT**.

```
reliableShadowConsumerInitiatedAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowConsumerContract
        ESTABLISHED BY                  association-by-RTSE
        INFORMATION TRANSFER BY         transfer-by-RTSE
        ABSTRACT SYNTAXES               { acse-abstract-syntax |
                                        reliableShadowBindingAbstractSyntax |
                                        directoryReliableShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-reliableShadowConsumerInitiatedAC }
```

7.2.4 Directory Operational Binding Management Application Context

The **dopContract** is realized as the **directoryOperationalBindingManagementAC**. This application context is specified as an information object of the class **APPLICATION-CONTEXT**.

```
directoryOperationalBindingManagementAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        dopContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax |
                                        directoryOperationalBindingManagementAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-directoryOperationalBindingManagementAC }
```

7.3 Operation Codes

7.3.1 Operation Codes for DAP and DSP Packages

The following operation codes are used by the operation packages of the DAP and the DSP:

```
id-opcode-read              Code    ::=   local : 1
id-opcode-compare           Code    ::=   local : 2
id-opcode-abandon           Code    ::=   local : 3
id-opcode-list              Code    ::=   local : 4
id-opcode-search            Code    ::=   local : 5
id-opcode-addEntry          Code    ::=   local : 6
id-opcode-removeEntry       Code    ::=   local : 7
id-opcode-modifyEntry       Code    ::=   local : 8
id-opcode-modifyDN          Code    ::=   local : 9
```

7.3.2 Operation Codes for DISP Packages

The following operation codes are used by the operation packages of the DISP.

```
id-opcode-requestShadowUpdate       Code    ::=   local : 1
id-opcode-updateShadow              Code    ::=   local : 2
id-opcode-coordinateShadowUpdate    Code    ::=   local : 3
```

7.3.3 Operation Codes for DOP Packages

The following operation codes are used by the operation package of the DOP.

```
id-op-establishOperationalBinding     Code    ::=   local : 100
id-op-modifyOperationalBinding        Code    ::=   local : 102
id-op-terminateOperationalBinding     Code    ::=   local : 101
```

7.4 Error Codes

7.4.1 Error Codes for DAP and DSP Packages

The following error codes are used by the operation packages of the DAP and the DSP. The code **id-errcode-referral** is only used in the DAP. The code **id-opcode-dsaReferral** is only used in the DSP.

```
id-errcode-attributeError       Code    ::=    local : 1
id-errcode-nameError            Code    ::=    local : 2
id-errcode-serviceErro          Code    ::=    local : 3
id-errcode-referral             Code    ::=    local : 4
id-errcode-abandoned            Code    ::=    local : 5
id-errcode-securityError        Code    ::=    local : 6
id-errcode-abandonFailed        Code    ::=    local : 7
id-errcode-updateError          Code    ::=    local : 8
id-errcode-dsaReferral          Code    ::=    local : 9
```

7.4.2 Error Codes for DISP Packages

The following error code is used by the operation packages of the DISP.

```
id-errcode-shadowError          Code    ::=    local : 1
```

7.4.3 Error Codes for DOP Packages

The following error codes are used by the operation package of the DOP.

```
id-err-operationalBindingError  Code    ::=    local : 100
```

[7.5.2.2 Pointer to rules of extensibility: The text referred to by clause 7 of the 4th edition of ITU-T Rec. X.509 | ISO/IEC 9594-8 has been moved to 12.2.2 in this Directory Specification.]

8 Directory protocol mapping onto OSI services

This clause defines the mapping of the DAP, DSP, DOP and DISP onto the used services.

The mapping onto used services of the DAP, DSP and DOP, as well as for the DISP application contexts that omit the RTSE is defined in 8.1. The mapping onto used services for the DISP application contexts that use the RTSE is defined in 8.2.

8.1 Application contexts omitting RTSE

This subclause defines the mapping onto used services of the DAP, DSP and DOP application contexts, as well as the DISP application contexts that do not include the RTSE.

8.1.1 Mapping onto ACSE

This subclause defines the mapping of the (**DirectoryBind**, **DSABind**, **DSAShadowBind** or **DSADOPBind**) and (**DirectoryUnbind**, **DSAUnbind**, **DSAShadowUnbind** or **DSADOPUnbind**) services onto the services of the ACSE. The ACSE is defined in ITU-T Rec. X.217 | ISO/IEC 8649.

8.1.1.1 Bind onto A-ASSOCIATE

The **DirectoryBind**, **DSABind**, **DSAShadowBind** or **DSADOPBind** service is mapped onto the A-ASSOCIATE service of the ACSE. The use of the parameters of the A-ASSOCIATE service is qualified in the following subclauses.

8.1.1.1.1 Mode

This parameter shall be supplied by the initiator of the association in the A-ASSOCIATE request primitive, and shall have the value 'normal mode'.

8.1.1.1.2 Application context name

The initiator of the association shall propose one of the following application contexts:

a) For the DAP, the **directoryAccessAC**;

b) For the DSP, the **directorySystemAC**;

c) For the DOP, the **directoryOperationalBindingManagementAC**;

d) For the DISP, one of **shadowSupplierInitiatedAC**, **shadowConsumerInitiatedAC**, **shadowSupplierInitiatedAsynchronousAC**, or **shadowConsumerInitiatedAsynchronousAC**.

8.1.1.1.3 User Information

The mapping of the **DirectoryBind** or **DSABind** onto the User Information parameters of the A-ASSOCIATE request primitive is defined in ITU-T Rec. X.880 | ISO/IEC 13712-1.

8.1.1.1.4 Presentation Context Definition List

The initiator of the association shall supply the Presentation Context Definition List in the A-ASSOCIATE request primitive which shall contain the ACSE abstract-syntax (**id-as-acse**) and either the DAP abstract syntax (**id-as-directoryAccessAS**), the DSP abstract syntax (**id-as-directorySystemAS**), the DOP abstract syntax (**id-as-directoryOperationalBinding ManagementAS**), or the DISP abstract syntax (**id-as-directoryShadowAS**).

8.1.1.1.5 Quality of Service

This parameter shall be supplied by the initiator of the association in the A-ASSOCIATE request primitive, and by the responder of the association in the A-ASSOCIATE response primitive. The parameters 'Extended Control' and 'Optimized Dialogue Transfer' shall be set to "feature not desired". The remaining parameters shall be such that default values are used.

8.1.1.1.6 Session Requirements

This parameter shall be set by the initiator of the association in the A-ASSOCIATE request primitive, and by the responder of the association in the A-ASSOCIATE response primitive. The parameter shall be set to specify the following functional units:

a) Kernel;

b) Duplex.

8.1.1.1.7 Application Entity Title and Presentation Address

These parameters shall be supplied by the initiator and the responder of the association (Application Entity Title is optionally supplied).

For a DUA establishing an association for an initial request, these parameters are obtained from locally held information.

For a DUA (or DSA) establishing an association with a DSA to which it has been referred, these parameters are obtained from the **AccessPoint** value of a **Continuation Reference**.

For a DSA establishing an association, this parameter is obtained from its knowledge information, i.e. an external reference.

8.1.1.2 Unbind onto A-RELEASE

The **DirectoryUnbind**, **DSAUnbind**, **DSAShadowUnbind** or **DSADOPUnbind** is mapped onto the A-RELEASE service of the ACSE. The use of the parameters of the A-RELEASE service is qualified in the following subclause.

8.1.1.2.1 Result

This parameter shall have the value 'affirmative'.

8.1.1.3 Use of A-ABORT and A-P-ABORT Services

The application-process is the user of the A-ABORT and A-P-ABORT services of the ACSE.

8.1.2 Mapping onto ROSE

The Directory ASE services are mapped onto the RO-INVOKE, RO-RESULT, RO-ERROR, RO-REJECT-U and RO-REJECT-P services of the ROSE. The mapping of the abstract-syntax notation of the Directory ASEs onto the ROSE services is as defined in ITU-T Rec. X.880 | ISO/IEC 13712-1.

8.2 Application contexts including RTSE

This subclause defines the mapping onto used services for the DISP application contexts that include the RTSE. Support for this mapping is conditional on a claim of conformance to these application contexts. The RTSE is defined in ITU-T Rec. X.218 | ISO/IEC 9066-1.

8.2.1 Mapping onto RT-OPEN and RT-CLOSE

This subclause defines the mapping of the **DSAShadowBind** and **DSAShadowUnbind** services onto the RT-OPEN and RT-CLOSE services of the RTSE.

8.2.1.1 DSAShadowBind onto RT-OPEN

The **DSAShadowBind** is mapped onto the RT-OPEN service of the RTSE. The use of the parameters of the RT-OPEN service is qualified in the following subclauses.

8.2.1.1.1 Mode

This parameter shall be supplied by the initiator of the association in the RT-OPEN request primitive, and shall have the value "normal mode".

8.2.1.1.2 Application context name

The initiator of the association shall propose either the **reliableShadowSupplierInitiatedAC** application context or the **reliableShadowConsumerInitiatedAC** application context in the RT-OPEN request primitive.

8.2.1.1.3 User-data

The mapping of the bind-operation onto the user-data parameter of the RT-OPEN request primitive is defined in ITU-T Rec. X.880 | ISO/IEC 13712-1.

8.2.1.1.4 Presentation Context Definition List

The initiator of the association shall supply the Presentation Context Definition List in the **RT-OPEN** request primitive which shall contain the ACSE abstract-syntax (**id-as-acse**) and the DISP abstract-syntax that includes the RTSE (**id-as-directoryReliableShadowAS**).

8.2.1.1.5 Initial turn

This parameter shall be supplied by the initiator of the association in the RT-OPEN request primitive, and shall have the value "association-initiator".

8.2.1.1.6 Application Entity Title and Presentation Address

These parameters shall be supplied by the initiator and the responder of the association in the RT-OPEN request primitive (Application Entity Title is optionally supplied).

8.2.1.2 DSAShadowUnbind onto RT-CLOSE

The **DSAShadowUnbind** is mapped onto the RT-CLOSE service of the RTSE.

8.2.2 Mapping onto ROSE

The **shadowSupplierASE** and the **shadowConsumerASE** services are mapped onto the RO-INVOKE, RO-RESULT, RO-ERROR, RO-REJECT-U and RO-REJECT-P services of the ROSE. The mapping of the abstract-syntax notation of these DISP ASEs onto the ROSE services is as defined in ITU-T Rec. X.880 | ISO/IEC 13712-1.

ROSE is the user of the RT-TRANSFER, RT-TURN-PLEASE, RT-TURN-GIVE, RT-P-ABORT and RT-U-ABORT services of the RTSE. The use of the RTSE services by the ROSE is defined in ITU-T Rec. X.882 | ISO/IEC 13712-3.

8.2.2.1 Managing the turn

ITU-T Rec. X.881 | ISO/IEC 13712-2 defines the use by the ROSE of the RT-TURN-PLEASE and RT-TURN-GIVE services of the RTSE to manage the turn.

The values of the priority parameter of the RT-TURN-PLEASE service used by the ROSE to request the turn are as follows:

- *Priority zero* is the highest priority, and is reserved for the action of releasing the association by the initiator.
- *Priority one* is used by the ROSE to provide the RO-REJECT-U and RO-ERROR services of the ROSE.
- *Priority two* is used by the ROSE to provide the RO-RESULT service of the ROSE.
- *Priority three* is used by the ROSE to provide the RO-INVOKE service of the ROSE.

9 IDM protocol

This clause defines the Internet Directly Mapped Protocol (IDM), a mapping of request-response service elements directly onto the Internet TCP/IP protocol, bypassing the ROSE, ACSE, Presentation, Session and Transport layers of the OSI model. The protocol is deliberately minimal and is designed for simplicity of implementation. It is connection-oriented and is fully asynchronous.

The protocol makes use of a number of protocol data units to transfer bind, request, response and error messages.

9.1 IDM-PDUs

The messages of the Internet Directly Mapped protocol are conveyed over a TCP/IP connection as protocol data units called IDM-PDUs. The ASN.1 definition for an IDM-PDU follows.

```
IDM-PDU {IDM-PROTOCOL:protocol} ::= CHOICE {
        bind            [0]  Bind{ {protocol} },
        bindResult      [1]  BindResult{ {protocol} },
        bindError       [2]  BindError{ {protocol} },
        request         [3]  Request{ {protocol.&Operations} },
        result          [4]  Result{ {protocol.&Operations} },
        error           [5]  Error{ {protocol.&Operations} },
        reject          [6]  Reject,
        unbind          [7]  Unbind,
        abort           [8]  Abort }

Bind {IDM-PROTOCOL:Protocols} ::= SEQUENCE {
        protocolID                IDM-PROTOCOL.&id ({Protocols}),
        callingAETitle      [0]   GeneralName     OPTIONAL,
        calledAETitle       [1]   GeneralName     OPTIONAL,
        argument            [2]   IDM-PROTOCOL.&bind-operation.&ArgumentType
                                      ({Protocols} {@protocolID}) }

BindResult {IDM-PROTOCOL:Protocols} ::= SEQUENCE {
        protocolID                IDM-PROTOCOL.&id ({Protocols}),
        respondingAETitle   [0]   GeneralName     OPTIONAL,
        result              [1]   IDM-PROTOCOL.&bind-operation.&ResultType
                                      ({Protocols} {@protocolID}) }

BindError {IDM-PROTOCOL:Protocols} ::= SEQUENCE {
        protocolID                IDM-PROTOCOL.&id ({Protocols}),
        errcode                   IDM-PROTOCOL.&bind-operation.&Errors.&errorCode
                                      ({Protocols} {@protocolID}),
        respondingAETitle   [0]   GeneralName     OPTIONAL,
        aETitleError              ENUMERATED {
                                      callingAETitleNotAccepted (0),
                                      calledAETitleNotRecognized (1) } OPTIONAL,
        error               [1]   IDM-PROTOCOL.&bind-operation.&Errors.&ParameterType
                                      ({Protocols} {@protocolID, @errcode}) }

Request {OPERATION:Operations} ::= SEQUENCE {
        invokeID          INTEGER,
        opcode            OPERATION.&operationCode ({Operations}),
        argument          OPERATION.&ArgumentType ({Operations} {@opcode}) }
```

```
Result {OPERATION:Operations} ::=  SEQUENCE {
        invokeID            INTEGER,
        opcode              OPERATION.&operationCode ({Operations}),
        result              OPERATION.&ResultType ({Operations} {@opcode}) }

Error {OPERATION:Operations}  ::=  SEQUENCE {
        invokeID            INTEGER,
        errcode             OPERATION.&Errors.&errorCode ({Operations}),
        error               OPERATION.&Errors.&ParameterType
                                ({Operations} {@errcode}) }

Reject  ::=  SEQUENCE {
        invokeID    INTEGER,
        reason      ENUMERATED {
                        mistypedPDU                 (0),
                        duplicateInvokeIDRequest    (1),
                        unsupportedOperationRequest (2),
                        unknownOperationRequest     (3),
                        mistypedArgumentRequest     (4),
                        resourceLimitationRequest   (5),
                        unknownInvokeIDResult       (6),
                        mistypedResultRequest       (7),
                        unknownInvokeIDError        (8),
                        unknownError                (9),
                        mistypedParameterError      (10) } }

Unbind  ::=  NULL

Abort  ::=  ENUMERATED {
                    mistypedPDU         (0),
                    unboundRequest      (1),
                    invalidPDU          (2),
                    resourceLimitation  (3),
                    connectionFailed    (4),
                    invalidProtocol     (5),
                    reasonNotSpecified  (6) }
```

A **bind** PDU is sent to request a binding between the sender and the responder. **protocolID** identifies the **IDM-PROTOCOL** protocol to be used (see 9.4). **argument** is a value for the **ARGUMENT** field of the **BIND-OPERATION** of the identified protocol. **callingAETitle** is the name of the local application entity sending the **bind** PDU. **calledAETitle** is the name of the remote application entity to which the **bind** PDU is being sent.

A **bindResult** PDU is returned in response to a successful bind request. **protocolID** is the same value sent in the corresponding **bind** PDU. **result** is a value for the **RESULT** field of the **BIND-OPERATION** of the identified protocol. **respondingAETitle** is the name of the remote application entity which sent the **bindResult**.

A **bindError** PDU is returned in response to an unsuccessful bind request. **protocolID** is the same value sent in the corresponding **bind** PDU. **errcode** is the code for one of the errors listed against the **ERRORS** field of the **BIND-OPERATION** of the identified protocol. **error** is a value for the **PARAMETER** field of the **ERROR** identified by **errcode**. **respondingAETitle** is the name of the remote application entity which sent the **bindError**. **aETitleError** is set to **callingAETitleNotAccepted** if a **bind** PDU is received and the supplied **callingAETitle** is not acceptable to the called system. **aETitleError** is set to **calledAETitleNotRecognized** if a **bind** PDU is received and the remote application entity knows the application entity which is binding, but does not accept the **calledAETitle** sent in the **bind** PDU as its own name.

A **request** PDU is sent to request an operation. **invokeID** identifies a particular request and its associated responses, and is a positive integer chosen to be different to the value sent in any previous request over that TCP/IP connection. **opcode** is the code for one of the operations listed against the **OPERATIONS** field of the chosen protocol. **argument** is a value for the **ARGUMENT** field of the **OPERATION** identified by **opcode**.

A **result** PDU is returned in response to a successful operation request. **invokeID** and **opcode** are the same values as sent in the request PDU to which this PDU is a reply. **result** is a value for the **RESULT** field of the **OPERATION** identified by **opcode**.

An **error** PDU is returned in response to an unsuccessful operation request. **invokeID** has the same value as sent in the request PDU to which this PDU is a reply. **errcode** is the code for one of the errors listed against the **ERRORS** field of the operation in the request PDU. **error** is a value for the **PARAMETER** field of the **ERROR** identified by **errcode**.

A **reject** PDU is returned in response to a protocol error detected in a received **request**, **result** or **error** PDU from which an invoke ID can be recovered. **invokeID** is the invoke ID of the received PDU that was in error. **reason** is an integer code for the error, as described in 9.5.

An **unbind** PDU is sent to close a binding in an orderly manner, as described in 9.3. It has no parameters.

9.2 Use of OPERATION and ERROR classes

The IDM protocol makes use of the **OPERATION** and **ERROR** information object classes of ROSE, so that modules that define an Abstract Service using these information object classes can be used in the protocol without extensive rewriting of the module. So that the IDM protocol can be implemented without reference to ROSE, the following definitions of **OPERATION** and **ERROR** are provided to be used in place of the ROSE definitions. The definitions are equivalent and compatible with the ROSE definitions but are self-contained and considerably simpler since other ROSE functionality is not required.

```
OPERATION ::= CLASS {
        &ArgumentType   OPTIONAL,
        &ResultType     OPTIONAL,
        &Errors         ERROR OPTIONAL,
        &operationCode  Code UNIQUE OPTIONAL }
WITH SYNTAX {
        ARGUMENT        &ArgumentType
        RESULT          &ResultType
        ERRORS          &Errors
        CODE            &operationCode }

ERROR ::= CLASS {
        &ParameterType,
        &errorCode      Code UNIQUE OPTIONAL }
WITH SYNTAX {
        PARAMETER       &ParameterType
        CODE            &errorCode }

Code ::= CHOICE {
        local       INTEGER,
        global      OBJECT IDENTIFIER }
```

9.3 Sequencing requirements

9.3.1 Binding

The initiator of the TCP/IP connection shall send the **bind** PDU to the responder. The responder shall reply by sending either a **bindResponse** or a **bindError** PDU. Once the **bindResponse** PDU has been received an *association* is said to be in place between the initiator and the responder.

The initiator shall send a **bind** PDU before sending **request** PDUs. It may send **request** PDUs after sending the **bind** PDU but before receiving a **bindResponse** or **bindError**. The responder shall process and reply to a received **bind** PDU before processing and replying to received **request** PDUs.

If the protocol permits the responder to initiate requests, the responder may initiate such requests as soon as it has sent a **bindResponse** PDU. The initiator shall process the **bindResponse** before replying to received **request** PDUs.

If a **bindError** is received, the initiator may choose whether to attempt another bind by sending a new bind PDU or whether to close the TCP/IP connection.

If both application entities use the **AETitle** information of the **bind** PDU, a **bindError** PDU with **aETitleError** set to **callingAETitleNotAccepted** or **calledAETitleNotRecognized** can be received as a response to a **bind** PDU.

9.3.2 Unbinding

When the DAP protocol is being used, only the initiator of the bind shall send an **unbind** PDU. For any other protocol, either the initiator or responder may send an **unbind** PDU. An **unbind** is destructive in that the results of any outstanding operations are lost (undefined). To avoid loss of data the initiator should only unbind when all requests have been responded to.

Either the initiator or responder may close the underlying TCP/IP connection at any time. Any outstanding requests are lost.

9.3.3 Requests and responses

A **request** PDU may be sent at any time after sending a **bind** PDU or **bindResult** PDU, and requests the recipient of the PDU to perform the indicated operation. The recipient of the **request** PDU shall reply with a **result**, **error**, or **reject** PDU.

Requests are asynchronous and the order of the responses is not guaranteed to be the same as that of the requests.

The receiver of a response shall use the invoke ID as the primary indicator of the request to which the response belongs, and shall reject the response if the invoke ID is in error.

9.3.4 Rejects

The **reject** PDU shall be used to indicate that a problem was encountered in processing a **request**, **result**, or **error** PDU.

If any other protocol error occurs or if the invoke ID cannot be determined, the connection shall be closed.

9.4 Protocols

Protocols for use within the IDM protocol are defined through use of the **IDM-PROTOCOL** information object class, defined as follows:

```
IDM-PROTOCOL ::= CLASS {
        &bind-operation         OPERATION,
        &Operations             OPERATION,
        &id                     OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        BIND-OPERATION          &bind-operation
        OPERATIONS              &Operations
        ID                      &id }
```

Each instance of an **IDM-PROTOCOL** class defines the bind operation and request/response operations for use within the IDM protocol. The **bindOperation** field defines the operation to be used for binding; the **ARGUMENT** field of this operation is used with the **bind** PDU that signals the protocol, the **RESULT** field is used with the **bindResult** PDU, and one of the errors given in the **ERRORS** field of this operation is used with the **bindError** PDU. The **Operations** field defines the operations that may be used within the **request**, **result** and **error** PDUs of the IDM protocol. The **id** field is the protocol identifier. It also implicitly determines the application context for a bind operation. As a consequence, a separate **IDM-PROTOCOL** is defined for each required application context.

9.5 Reject reasons

A **reject** PDU is returned in response to various error conditions. The error conditions and the reason code with which they are signalled are described below:

A **mistypedPDU** reason is returned if the PDU is invalidly constructed.

A **duplicateInvokeIDRequest** reason is returned if a **request** PDU is received and the **invokeID** has previously been used since the connection was established.

An **unsupportedOperationRequest** reason is returned if a **request** PDU is received and the requested operation is not supported.

An **unknownOperationRequest** reason is returned if a **request** PDU is received and the requested operation is unknown.

A **mistypedArgumentRequest** reason is returned if a **request** PDU is received and the **argument** is invalidly constructed.

A **resourceLimitationRequest** reason is returned if a **request** PDU is received and no operations can be performed because of resource limitations.

An **unknownInvokeIDResult** reason is returned if a **result** PDU is received and the **invokeID** does not match that of an operation to which a response is expected.

A **mistypedResultRequest** reason is returned if a **result** PDU is received and the **result** is invalidly constructed, or the **opcode** does not match that of the corresponding **request** PDU.

An **unknownInvokeIDError** reason is returned if an **error** PDU is received and the **invokeID** does not match that of an operation to which a response is expected.

An **unknownError** reason is returned if an **error** PDU is received and the indicated **error** does not belong to the indicated protocol or is not permitted as a response to the operation.

A **mistypedParameterError** reason is returned if an **error** PDU is received and the **parameter** is invalidly constructed, or the **opcode** does not match that of the corresponding **request** PDU.

9.6 Abort reasons

An **Abort** PDU is returned in response to various error conditions which are not covered by the **Reject** nor the **BindError** PDUs. The error conditions and the reason code with which they are signalled are described below:

A **mistypedPDU** reason is returned if the PDU received has an invalid construction.

An **unboundRequest** reason is returned if a **request** PDU request is received before an association has been established.

A **invalidPDU** reason is returned if a DSA gets a PDU which is no IDM-PDU.

A **resourceLimitation** reason is returned if a **Bind** PDU is received and no operations can be performed because of resource limitations, e.g. maximum number of connections exceeded.

A **connectionFailed** reason is returned if the DSA was not able to create the TCP/IP connection in order to send a **Bind** PDU.

A **invalidProtocol** reason is returned if a **resultBind**, a **BindResult** or a **BindError** PDU is received and the **protocolID** is unknown or not supported.

A **reasonNotSpecified** reason is returned if the initiator or the responder wants to close the association for any other reason.

NOTE – An abort may be generated by the underlying service of the initiator, resulting in protocol that will not flow across the connection, e.g. returning an abort with **unboundRequest** would be initiated by the underlying service as opposed to the target system which cannot be reached.

9.7 Mapping onto TCP/IP

Each IDM PDU is encoded using the ASN.1 Basic Encoding Rules without restriction. The binary data resulting from the encoding is then partitioned and placed in one or more segments to be sent over the TCP/IP connection. Each segment has a *header* and carries the next *fragment* or portion of the encoded data. The division of an IDM PDU into fragments and the size of any fragment are at the choice of the sender and carry no significance. All fragments of an IDM PDU shall be sent before another IDM PDU is sent.

The format for a segment (header plus fragment of an IDM PDU) is as follows:

version (1 octet)	final (1 octet)	length (4 octets)	data (*length* octets)

`version` indicates the version of the IDM PDU and its mapping onto TCP/IP. The version described in this Directory Specification shall be indicated with the value 1. All packets on a connection shall have the same value of `version`.

NOTE – How the communicating parties negotiate the version number is for further study.

`final` indicates whether `data` holds a non-final IDM PDU fragment (value 0), or the whole value or final fragment (value 1).

`length` is the length of `data` field in octets. It is sent in 'network octet order' with more significant octets preceding less significant octets. The minimum value of length is 1. For performance reasons, it is recommended that the whole IDM PDU be contained in one segment if the length can be expressed in the 4 octets of the length field; IDM fragmentation should only be used if the length of the IDM PDU cannot be expressed in 4 octets.

data holds the next fragment of the IDM PDU being conveyed, or the whole IDM PDU if the whole value is conveyed in one fragment.

9.8 Addressing

An IDM-style communications endpoint is defined by its IP address and its port number, and can be written in the notation of IETF RFC 1738 as:

```
idm://host:port
```

This clause defines an equivalent OSI network address format for such an endpoint, in order to allow the IDM protocol to be used with service definitions that make reference to OSI presentation addresses (such as the Directory service definitions). A presentation address for a system that supports IDM access is structured just as for OSI access except that the P-, S-, and T-selectors are ignored if present and the network address is of the form specified below. Systems that support both OSI and IDM stacks can have a single OSI presentation address containing OSI and IDM network addresses.

The OSI network address format for an IDM endpoint follows that described in IETF RFC 1277. Expressed as an octet string, it consists of 29 binary coded decimal digits and one fill digit, as follows:

- The AFI (first 2 digits) is '54' (F.69 format, decimal, leading zero significant).
- The IDI (next 8 digits) is '00728722'.
- The DSP (next 20 digits) is constructed as follows:
  - The first 2 digits form the DSP prefix and have the value '10' for IDM

    NOTE 1 – The values 01, 02, 03 and 06 are already assigned in IETF RFC 1277. 03 is the value for the RFC 1006 stack.
  - The next 12 digits are the 4-component dotted decimal IP address, with 3 digits per component.
  - The next 5 digits are the port number

    NOTE 2 – The port number is optional in IETF RFC 1277 but mandatory for IDM.
  - The last digit is a final single hex 'F' to pack out the DSP to a full octet.

A DSA able to communicate over two different stacks (such as IDM over TCP/IP or OSI over TCP/IP using RFC 1006) would have two network addresses in its presentation address. For example, if the DSA uses the port 1200 for the IDM stack and the default port 102 for the OSI stack, then the DSA's **myAccessPoint** would hold a presentation address containing:

- a network address 1 for IDM with the following encoding, (containing as own IP address the loopback address 127.0.0.1 and the port 1200):

  '540072872210127000000000101200F'H
- a network address 2 for OSI over RFC 1006 with the following encoding, (containing as own IP address again the loopback address 127.0.0.1; the port 102, which is the default for RFC 1006, is no more explicitly in the encoding, which is allowed because the DSP prefix is in this case 03 for RFC 1006 and no more 10 as for IDM):

  '54007287220312700000000 1'H

10 Directory protocol mapping onto the IDM protocol

This clause gives definitions for mapping the Directory protocols onto the IDM protocol. The complete **DirectoryIDMProtocols** module is given in Annex F. The components are repeated in this clause for clarity.

10.1 DAP-IP Protocol

The DAP-IP protocol **dap-ip** (Directory Access Protocol over TCP/IP) is used to invoke operations of the **DirectoryAbstractService** abstract service. It is defined as:

```
DAP-IDM-PDUs  ::=  IDM-PDU (dap-ip)

dap-ip IDM-PROTOCOL ::= {
        BIND-OPERATION      directoryBind
        OPERATIONS          { read | compare | abandon | list | search
                                | addEntry | removeEntry | modifyEntry | modifyDN }
        ID                  id-idm-dap }
```

The operation and error codes for this protocol are the same as those given in 7.3.1 and 7.4.1.

Only DUAs shall initiate connections using this protocol. Only the initiator of a connection shall request operations of the protocol.

10.2 DSP-IP Protocol

The DSP-IP protocol **dsp-ip** (Directory System Protocol over TCP/IP) is used to invoke operations of the **DistributedOperations** abstract service. It is defined as:

```
DSP-IDM-PDUs  ::=  IDM-PDU (dsp-ip)

dsp-ip IDM-PROTOCOL  ::=  {
        BIND-OPERATION      directoryBind
        OPERATIONS          { chainedRead | chainedCompare | chainedAbandon
                                | chainedList | chainedSearch
                                | chainedAddEntry | chainedRemoveEntry
                                | chainedModifyEntry | chainedModifyDN }
        ID                  id-idm-dsp }
```

The operation and error codes for this protocol are the same as those given in 7.3.1 and 7.4.1.

DSAs may use this protocol, and both the initiator and the acceptor of a connection may request operations of the protocol.

10.3 DISP-IP Protocol

The DISP-IP protocol **disp-ip** (Directory Information Shadowing Protocol over TCP/IP) is used to invoke operations of the **DirectoryShadowAbstractService** abstract service. It is defined as:

```
DISP-IDM-PDUs  ::=  IDM-PDU (disp-ip)

disp-ip IDM-PROTOCOL  ::=  {
        BIND-OPERATION      directoryBind
        OPERATIONS          { requestShadowUpdate
                                | updateShadow
                                | coordinateShadowUpdate }
        ID                  id-idm-disp }
```

The operation and error codes for this protocol are the same as those given in 7.3.2 and 7.4.2.

DSAs may use this protocol, and both the initiator and the acceptor of a connection may request operations of the protocol.

10.4 DOP-IP Protocol

The DOP-IP protocol **dop-ip** (Directory Operational Binding Protocol over TCP/IP) is used to invoke operations of the **OperationalBindingManagement** abstract service. It is defined as:

```
DOP-IDM-PDUs  ::=  IDM-PDU (dop-ip)

dop-ip IDM-PROTOCOL  ::= {
        BIND-OPERATION      directoryBind
        OPERATIONS          { establishOperationalBinding
                                | modifyOperationalBinding
                                | terminateOperationalBinding}
        ID                  id-idm-dop }
```

The operation and error codes for this protocol are the same as those given in 7.3.3 and 7.4.3.

DSAs may use this protocol, and both the initiator and the acceptor of a connection may request operations of the protocol.

11 Protocol stack coexistence

Subclause 9.8 defined an OSI network address format for an IDM communications endpoint. This clause recommends an approach for coexistence between DSAs supporting different protocol stacks, such as OSI, IDM and LDAP. In order to allow referrals to contain LDAP access points, this clause also specifies an OSI network address format for an LDAP communications endpoint.

11.1 Coexistence between OSI and IDM stacks

During migration to the use of IDM, DSAs may be deployed that support either only the OSI upper layers protocol stack (i.e. ACSE, Presentation, and Session) or only the IDM protocol stack. New DSA products may choose only to implement the IDM stack. Older DSA products may or may not choose to add support of IDM to their existing implementations. Interoperability between such DSAs will be accomplished through the use of referrals.

If a chaining DSA needs to forward a request to a target DSA and if the two DSAs do not support a protocol stack in common, then the chaining DSA shall return instead a referral. That referral will be returned through each DSA that chained the request. If any one of these DSAs supports the target DSA's protocol stack, it may choose to send the request directly to the target DSA identified in the referral.

If none of the chaining DSAs support the target DSA's protocol stack, the referral shall be returned to the DUA. That DUA may be able to send the request directly to the target DSA.

If deploying within a domain a mixture of DSA products some of which support only one protocol stack, it is recommended that either:

a) DSAs holding knowledge of DSAs that support only one protocol stack should support that protocol stack; or

b) the DSA to which the DUA binds should support both protocol stacks.

11.2 Coexistence in the presence of LDAP

DSAs supporting either the OSI upper layer protocol stack or the IDM protocol stack may also choose to support LDAP. Interoperability between such DSAs may be accomplished through the use of chaining or referrals. Interoperability between such DSAs and DUAs may be accomplished through the use of LDAP or DAP.

In order for a DSA to be able to provide useful referrals for DUAs supporting only LDAP, it is necessary to represent the LDAP access point of a potential target DSA in an OSI presentation address. Subclause 11.3 defines an NSAP format for LDAP. A DSA getting a referral containing an NSAP of this type can convert it to an LDAP referral and send it back to the connected LDAP client.

11.3 Defining an NSAP format for LDAP

This clause defines an OSI network address format for an LDAP communication endpoint, in order to allow this NSAP to be used with service definitions that refer to OSI presentation addresses (such as the Directory service definitions). A presentation address for a system that supports LDAP access is structured just as for OSI access except that the P-, S-,

and T-selectors are ignored if present and the network address is of the form specified below. Systems that simultaneously support OSI, IDM and LDAP stacks can have a single OSI presentation address containing OSI, IDM and LDAP network addresses.

The OSI network address format for an LDAP endpoint follows that described in IETF RFC 1277. Expressed as an octet string, it consists of 29 binary coded decimal digits and one fill digit, as follows:

– The AFI (first 2 digits) is '54' (F.69 format, decimal, leading zero significant).

– The IDI (next 8 digits) is '00728722'.

– The DSP (next 20 digits) is constructed as follows:

  • The first 2 digits form the DSP prefix and have the value '11' for LDAP.

    NOTE 1 – The values 01, 02, 03, and 06 are already assigned in IETF RFC 1277. 03 is the value for the RFC 1006 stack. The value '10' is the value for the IDM stack.

  • The next 12 digits are the 4-component dotted decimal IP address, with 3 digits per component.

  • The next 5 digits are the port number.

    NOTE 2 – The port number is optional in IETF RFC 1277 but mandatory for LDAP.

  • The last digit is a final single hex 'F' to pack out the DSP to a full octet.

A DSA able to communicate over three different stacks (such as IDM over TCP/IP or OSI over TCP/IP using RFC 1006 or LDAP) would have three network addresses in its presentation address. For example, if the DSA uses port 1200 for the IDM stack, the default port 102 for the OSI stack and port 389 for LDAP, then the DSA's **myAccessPoint** would hold a presentation address containing:

– a network address 1 for IDM with the following encoding, (containing as own IP address the loopback address 127.0.0.1 and the port 1200):

  '54007287221012700000000101200F'H

– a network address 2 for OSI over RFC 1006 with the following encoding, (containing as own IP address again the loopback address 127.0.0.1; the port 102 ,which is the default for RFC 1006, is not explicitly included in the encoding, which is allowed because the DSP prefix is in this case 03 for RFC 1006 as opposed to 10 as for IDM):

  '54007287220312700000000001'H

– a network address 3 for LDAP with the following encoding, (containing as own IP address the loopback address 127.0.0.1 and the port 389):

  '54007287221127000000000100389F'H

12 Versions and the rules for extensibility

This clause describes version negotiation rules and rules for extensibility for the OSI-mapped protocols defined in clause 7, and the IDM-mapped protocols defined in clause 10.

The Directory may be distributed and more than two Directory Application Entities may interoperate to service a request. The Directory AEs may be implemented conforming to different editions of the Directory specification of the Directory service which may or may not be represented by different protocol version numbers. The version number is negotiated to the highest common version number between two directly binding Directory AEs.

NOTE 1 – There are currently two versions of each Directory protocol. The 1988 edition and the 1993 edition are of version 1. Most features added in post-1993 editions are also available in version 1. However, some enhanced services and protocols, e.g. signed errors, require that version 2 has been negotiated among all involved parties.

A DUA may issue a request as specified in the latest edition of the Directory specification to which the DUA was implemented. Using the rules of extensibility defined below, that request shall be forwarded to the appropriate DSA that will respond to that request, regardless of the edition of the intervening DSAs. The responding DSA shall function as defined below.

NOTE 2 – An intermediate DSA only chaining the request may choose to examine selected elements of the Directory APDU that is needed to perform its function, e.g. name resolution.

12.1 DUA to DSA

12.1.1 Version negotiation

When accepting an association, i.e. binding, utilizing the DAP, the version negotiated shall only affect the point-to-point aspects of the protocol exchanged between the DUA and the DSA to which it is connected. Subsequent requests or responses on the association shall not be constrained by the version negotiated.

NOTE – There are no point-to-point aspects of the DAP that are currently indicated by different protocol versions.

12.1.2 Request and response processing

The DUA may initiate requests using the highest edition of the specification of that request it supports. If one or more elements of the request are critical, it shall indicate the extension number(s) in the **criticalExtensions** parameter.

NOTE 1 – If a value defined by an extension is encoded in a **CHOICE**, **ENUMERATED**, or **INTEGER** (used as **ENUMERATED**) type and if that type is essential for proper operation in a DSA implemented according to an earlier edition of the Specification, it is recommended that the extension be marked critical.

When processing a request from a DUA, a DSA shall follow the rules defined in 12.2.2.

When processing a response, a DUA shall:

a) ignore all unknown bit name assignments within a bit string; and

b) ignore all unknown named numbers in an **ENUMERATED** type or **INTEGER** type that is being used in the enumerated style, provided the number occurs as an optional element of a **SET** or **SEQUENCE**; and

c) ignore all unknown elements in **SET**s, at the end of **SEQUENCE**s, or in **CHOICE**s where the **CHOICE** is itself an optional element of a **SET** or **SEQUENCE**.

NOTE 2 – Implementations may as a local option ignore certain additional elements in a Directory PDU. In particular, some unknown named numbers and unknown **CHOICE**s in mandatory elements of **SET**s and **SEQUENCE**s can be ignored without invalidating the operation. The identification of such elements is for further study.

d) not consider the receipt of unknown attribute types and attribute values as a protocol violation; and

e) optionally report the unknown attribute types and attribute values to the user.

12.1.3 Extensibility rules for error handling

When processing a known error type with unknown indicated problems and parameters, a DUA shall:

a) not consider the receipt of unknown indicated problems and parameters as a protocol violation (i.e. it shall not issue a RO-U-REJECT or an IDM-REJECT, as appropriate, or abort the application association); and

b) optionally report the additional error information to the user.

When processing an unknown error type, a DUA shall:

a) not consider the receipt of unknown error type as a protocol violation (i.e. it shall not issue a RO-U-REJECT or an IDM-REJECT, as appropriate, or abort the application association); and

b) optionally report the error to the user.

12.2 DSA to DSA

12.2.1 Version negotiation

When establishing or accepting an association, i.e. binding, utilizing the DSP, the version negotiated shall only effect the point-to-point aspects of the protocol exchanged between the DSAs. Subsequent requests or responses on the association shall not be constrained by the version negotiated.

NOTE 1 – There are no point-to-point aspects of the DSP that are currently indicated by different protocol versions.

When establishing or accepting an association, i.e. binding, utilizing the DISP, the version negotiated shall define all aspects of the protocol exchanged between the DSAs. Subsequent requests or responses on the association shall be constrained by the version negotiated.

NOTE 2 – There is currently only one version of the DISP protocol.

When establishing or accepting an association, i.e. binding, utilizing the DOP, the version negotiated shall define all aspects of the protocol exchanged between the DSAs. Subsequent requests or responses on the association shall be constrained by the version negotiated.

NOTE 3 – There is currently only one version of the DOP protocol.

12.2.2 Rules of extensibility for operation processing

If any DSA performing an operation (after name resolution is completed) detects an element of **criticalExtensions** whose semantic is unknown, it shall return an **unavailableCriticalExtension** indication as a **serviceError** or in a **PartialOutcomeQualifier**.

NOTE 1 – If a criticalExtensions string with one or more zero values is received, this indicates either that the extensions corresponding to the values are not present in the operation or are not critical. The presence of a zero value in a criticalExtensions string shall not be inferred as either the presence or absence of the corresponding extension in the APDU.

Otherwise, when processing a Directory PDU a DSA shall:

a) ignore all unknown bit name assignments within a bit string; and

b) ignore all unknown named numbers in an **ENUMERATED** type or **INTEGER** type that is being used in the enumerated style, provided the number occurs as an optional element of a **SET** or **SEQUENCE**; and

c) ignore all unknown elements in **SET**s, at the end of **SEQUENCE**s, or in **CHOICE**s where the **CHOICE** is itself an optional element of a **SET** or **SEQUENCE**.

NOTE 2 – Implementations may, as a local option, ignore certain additional elements in a Directory PDU. In particular, some unknown named numbers and unknown **CHOICE**s in mandatory elements of **SET**s and **SEQUENCE**s can be ignored without invalidating the operation. The identification of such elements is for further study.

12.2.3 Rules of extensibility for chaining

If the PDU is a request, the DSA shall forward the request containing the unknown types and values to any additional DSAs determined by the name resolution process.

If the PDU is a response, the DSA shall process the unknown types and values as it would process known types and values (see clause on results merging in the Directory Specification on Distributed Operations) and forward to the initiating DSA or DUA.

12.2.4 Rules of extensibility for error handling

When processing an known error type with unknown indicated problems and parameters, a DSA:

a) shall not consider the receipt of unknown indicated problems and parameters as a protocol violation (i.e. it shall not issue a RO-U-REJECT, or an IDM-REJECT, as appropriate, or abort the application association); and

b) may attempt to recover as appropriate to its understanding of just the error type, or may just return the error (and its unknown indicated problems and parameters) to the next appropriate DSA or DUA.

When processing an unknown error type, a DSA which is only involved in chaining the request shall:

a) not consider the unknown error type as a protocol violation (i.e. it shall not issue a RO-U-REJECT or an IDM-REJECT, as appropriate, or abort the application association); and

b) not attempt to correct or recover from the error and its indicated problems and parameters; and

c) return the unknown error type to the next appropriate DSA or DUA.

When processing an unknown error, a DSA which is correlating multiple responses shall:

a) not consider the unknown error type as a protocol violation (i.e. it shall not issue a RO-U-REJECT or an IDM-REJECT, as appropriate, or abort the application association); and

b) not attempt to correct or recover from the error and its indicated problems and parameters; and

c) put the unknown error in **PartialOutcomeQualifier**; and

d) continue correlating results as usual.

12.3 Rules of extensibility for object classes

Optional user attributes may be added to an existing object class without assigning a new object identifier.

A DSA not supporting an object class extension may reject any operation that attempts to create or modify an entry resulting in an extension attribute to be present in the entry.

12.4 Rules of extensibility for user attribute types

A user attribute type definition may be extended in such a way that its matching characteristics are not changed. This may include:

– adding values to **ENUMERATED** and **INTEGER** types that is being used in the enumerated style;

– adding bits to a bitstring.

A DSA is not required to handle an attribute value that includes such extensions.

A DUA shall not consider the receipt of an extended attribute value as an error.

13 Conformance

This clause defines the requirements for conformance to this Directory Specification.

13.1 Conformance by DUAs

A DUA implementation claiming conformance to this Directory Specification shall satisfy the requirements specified in 13.1.1 through 13.1.3.

13.1.1 Statement requirements

The following shall be stated:

a) the operations of the **directoryAccessAC** application-context and/or **dap-ip** protocol that the DUA is capable of invoking for which conformance is claimed;

b) the bind security level(s) for which conformance is claimed (none, simple, strong – and if simple, then whether without password, with password or with protected-password); and whether the DUA can generate signed arguments or validate signed results;

c) the extensions listed in the table of 7.3.1 of ITU-T Rec. X.511 | ISO/IEC 9594-3, that the DUA is capable of initiating for which conformance is claimed;

d) whether conformance is claimed to Rule-based Access Control; and

e) if conformance is claimed for strong authentication, signed operations, or protected operations, identification of the Certificate and CRL extensions for which conformance is claimed.

13.1.2 Static requirements

A DUA shall:

a) have the capability of supporting the **directoryAccessAC** application-context as defined by its abstract syntax in clause 7; and/or or the **dap-ip** protocol defined in clause 10;

b) conform to the extensions for which conformance was claimed in 13.1.1 c);

c) if conformance is claimed to Rule-based Access Control, have the capability of supporting security labels as identified in 19.4 of ITU-T Rec. X.501 | ISO/IEC 9594-2; and

d) conform to clauses 8 and 15 of ITU-T Rec. X.509 | ISO/IEC 9594-8 for the Certificate and CRL extensions for which conformance was claimed in 13.1.1 e).

13.1.3 Dynamic Requirements

A DUA shall:

a) shall conform to the mapping onto the used service defined in clause 8 or clause 10 or both; and

b) shall conform to the rules of extensibility procedures defined in 12.1.

13.2 Conformance by DSAs

A DSA implementation claiming conformance to this Directory Specification shall satisfy the requirements specified in 13.2.1 through 13.2.3.

13.2.1 Statement requirements

The following shall be stated:

a) The application-contexts and IDM protocols for which conformance is claimed: **directoryAccessAC**, **directorySystemAC**, **directoryOperationalBindingManagementAC**, **dap-ip**, **dsp-ip**, **dop-ip**, or a combination of these. A DSA that claims conformance to the **directoryOperationalBinding ManagementAC** or to the **dop-ip** in support of hierarchical operational bindings shall also support the **directorySystemAC** or **dsp-ip**. If a DSA is such that knowledge of it has been disseminated, causing knowledge references to the DSA to be held in other DSAs outside of its own DMD, then it shall claim conformance to the **directorySystemAC** or **dsp-ip**.

NOTE 1 – An application context shall not be divided except as stated herein; in particular, conformance shall not be claimed to particular operations.

b) The operational binding types for which conformance is claimed: **shadowOperationalBindingID**, **specificHierarchicalBindingID**, **non-specificHierarchicalBindingID**, or a combination of these. A DSA that claims conformance to the **shadowOperationalBindingID** shall support one or more of the application contexts for shadow suppliers and/or shadow consumers indicated in 13.3 and 13.4.

c) Whether or not the DSA is capable of acting as a first level DSA, as defined in ITU-T Rec. X.518 | ISO/IEC 9594-4.

d) If conformance is claimed to the application-context specified by **directorySystemAC** and/or associated with the **dap-ip** protocol, whether or not the chained mode of operation is supported, as defined in ITU-T Rec. X.518 | ISO/IEC 9594-4.

e) If conformance is claimed to the application-context specified by **directoryAccessAC** and/or associated with the **dap-ip** protocol, the bind security level(s) for which conformance is claimed (none, simple, strong – and if simple, then whether without password, with password, or with protected password); whether the DSA can perform originator authentication as defined in 22.1 of ITU-T Rec. X.518 | ISO/IEC 9594-4 and if so, whether identity-based or signature-based; and whether the DSA can perform result authentication as defined in 22.2 of ITU-T Rec. X.518 | ISO/IEC 9594-4.

f) If conformance is claimed to the application-context specified by **directorySystemAC** and/or associated with the **dsp-ip** protocol, the bind security level(s) for which conformance is claimed (none, simple, strong – and if simple, then whether without password, with password, or with protected password); whether the DSA can perform originator authentication as defined in 22.1 of ITU-T Rec. X.518 | ISO/IEC 9594-4 and if so, whether identity-based or signature-based; and whether the DSA can perform result authentication as defined in 22.2 of ITU-T Rec. X.518 | ISO/IEC 9594-4.

g) The selected attribute types defined in ITU-T Rec. X.520 | ISO/IEC 9594-6, and any other attribute types, for which conformance is claimed and whether for attributes based on the syntax **DirectoryString**, conformance is claimed for the **UniversalString**, **BMPString**, or **UTF8String** choices.

h) The selected object classes defined in ITU-T Rec. X.521 | ISO/IEC 9594-7, and any other object classes, for which conformance is claimed.

i) The extensions listed in the table of 7.3.1 of ITU-T Rec. X.511 | ISO/IEC 9594-3, that the DSA is capable of responding to for which conformance is claimed.

j) Whether conformance is claimed for collective attributes as defined in 8.9 of ITU-T Rec. X.501 | ISO/IEC 9594-2 and 7.6, 7.8.2 and 9.2.2 of ITU-T Rec. X.511 | ISO/IEC 9594-3.

k) Whether conformance is claimed for hierarchical attributes as defined in 7.6, 7.8.2 and 9.2.2 of ITU-T Rec. X.511 | ISO/IEC 9594-3.

l) The operational attribute types defined in ITU-T Rec. X.501 | ISO/IEC 9594-2 and any other operational attribute types for which conformance is claimed.

m) Whether conformance is claimed for return of alias names as described in 7.7.1 of ITU-T Rec. X.511 | ISO/IEC 9594-3.

n) Whether conformance is claimed for indicating that returned entry information is complete, as described in 7.7.6 of ITU-T Rec. X.511 | ISO/IEC 9594-3.

o) Whether conformance is claimed for modifying the object class attribute to add and/or remove values identifying auxiliary object classes, as described in 11.3.2 of ITU-T Rec. X.511 | ISO/IEC 9594-3.

p) Whether conformance is claimed to Basic Access Control.

q) Whether conformance is claimed to Simplified Access Control.

r) Whether the DSA is capable of administering the subschema for its portion of the DIT, as defined in ITU-T Rec. X.501 | ISO/IEC 9594-2.

NOTE 2 – The capability to administer a subschema shall not be divided; specifically, the capability to administer particular subschema definitions shall not be claimed.

s) The selected name bindings defined in ITU-T Rec. X.521 | ISO/IEC 9594-7 and any other name bindings, for which conformance is claimed.

t) Whether the DSA is capable of administering collective attributes, as defined in ITU-T Rec. X.501 | ISO/IEC 9594-2.

u) The selected context types defined in ITU-T Rec. X.520 | ISO/IEC 9594-6, and any other context types, for which conformance is claimed.

v) Whether conformance is claimed for contexts as defined in 8.7, 8.8 and 11.8 of ITU-T Rec. X.501 | ISO/IEC 9594-2, and 7.3 and 7.6 of ITU-T Rec. X.511 | ISO/IEC 9594-3.

w) Whether conformance is claimed for the use of contexts in RDNs, as defined in 8.5 and 9.3 of ITU-T Rec. X.501 | ISO/IEC 9594-2, 7.7 of ITU-T Rec. X.511 | ISO/IEC 9594-3, and ITU-T Rec. X.518 | ISO/IEC 9594-4.

x) Whether conformance is claimed for the management of the DSA Information Tree, as defined in 7.13 of ITU-T Rec. X.511 | ISO/IEC 9594-3.

y) Whether conformance is claimed for the use of systems management for administration of the Directory, as defined in ITU-T Rec. X.530 | ISO/IEC 9594-10.

z) The selected managed objects and management attribute types defined in ITU-T Rec. X.530 | ISO/IEC 9594-10, and any other managed objects and attributes, for which conformance is claimed.

aa) Whether conformance is claimed to Rule-based Access Control.

NOTE 3 – The support of security labels requires the following minimal support of contexts: Context lists as per 8.7 of ITU-T Rec. X.501 | ISO/IEC 9594-2 and **returnContexts** per 7.6 of ITU-T Rec. X.511 | ISO/IEC 9594-3.

bb) Whether conformance is claimed to integrity of Directory operations.

cc) Whether conformance is claimed to integrity and confidentiality of Directory operations.

dd) Whether conformance is claimed that the DSA can hold and provide access to encrypted and digitally signed information.

ee) If conformance is claimed for strong authentication, signed operations, or protected operations, identification of the Certificate and CRL extensions for which conformance is claimed.

13.2.2 Static requirements

A DSA shall:

a) have the capability of supporting the application-contexts whose abstract syntaxes are defined in clause 7, and the IDM protocols defined in clause 10, for which conformance is claimed;

b) have the capability of supporting the information framework defined by its abstract syntax in ITU-T Rec. X.501 | ISO/IEC 9594-2;

c) conform to the minimal knowledge requirements defined in ITU-T Rec. X.518 | ISO/IEC 9594-4;

d) if conformance is claimed as a first-level DSA, conform to the requirements support of the root context, as defined in ITU-T Rec. X.518 | ISO/IEC 9594-4;

e) have the capability of supporting the attribute types for which conformance is claimed; as defined by their abstract syntaxes;

f) have the capability of supporting the object classes for which conformance is claimed, as defined by their abstract syntaxes;

g) conform to the extensions for which conformance was claimed in 13.2.1 i);

h) if the capability to administer subschema as defined in ITU-T Rec. X.501 | ISO/IEC 9594-2 is claimed, the DSA shall be able to do this administration;

i) if conformance is claimed for collective attributes, have the capability of performing the related procedures defined in 7.6, 7.8.2 and 9.2.2 of ITU-T Rec. X.511 | ISO/IEC 9594-3;

j) if conformance is claimed for hierarchical attributes, have the capability of performing the related procedures defined in 7.6, 7.8.2 and 9.2.2 of ITU-T Rec. X.511 | ISO/IEC 9594-3;

k) have the capability of supporting the operational attribute types for which conformance is claimed;

l) if conformance is claimed to Basic Access Control, have the capability of holding ACI items that conform to the definitions of Basic Access Control;

m) if conformance is claimed to Simplified Access Control, have the capability of holding ACI items that conform to the definitions of Simplified Access Control;

n) have the capability of supporting the context types for which conformance is claimed, as defined by their abstract syntaxes;

o) if conformance is claimed for contexts, have the capability of performing the related procedures defined in ITU-T Rec. X.511 | ISO/IEC 9594-3;

p) if conformance is claimed for the use of contexts in RDNs, have the capability of performing the related procedures as defined in 9.3 of ITU-T Rec. X.501 | ISO/IEC 9594-2, 7.7 of ITU-T Rec. X.511 | ISO/IEC 9594-3, and ITU-T Rec. X.518 | ISO/IEC 9594-4;

q) if conformance is claimed for the management of the DSA Information Tree, have the capability of performing the related procedures as defined in 7.5 and 7.13 of ITU-T Rec. X.511 | ISO/IEC 9594-3;

r) if conformance is claimed for the support of the families of entries feature, have the capabilities as defined in 7.3.2, 7.6.4 and 7.8.3 of ITU-T Rec. X.511 | ISO/IEC 9594-3;

s) if conformance is claimed to the search relaxation feature, have the capabilities as defined in 13.6.2 of ITU-T Rec. X.501 | ISO/IEC 9594-2 and in 10.2.2 of ITU-T Rec. X.511 | ISO/IEC 9594-3. In particular an implementation shall specify:

- whether it supports the inclusion of the **RelaxationPolicy** construct in a search request;
- whether it supports mapping-based matching, matching rule substitution, or both; and
- if is supports mapping-based matching, what mappings are supported.

t) if conformance is claimed to the hierarchical group feature, have the capabilities as defined in 7.5 of ITU-T Rec. X.511 | ISO/IEC 9594-3;

in addition, the implementation shall declare:

- what hierarchy options are supported.

u) if conformance is claimed to basic administration of services, have the capabilities as defined in clause 16 of ITU-T Rec. X.501 | ISO/IEC 9594-2, and the basic checking procedures as defined in clause 13 of ITU-T Rec. X.511 | ISO/IEC 9594-3. This support includes:

- support for entry count;
- support of the service controls options **entryCount** and **performExactly**;
- support of the **notification** extension defined in 7.4 of ITU-T Rec. X.511 | ISO/IEC 9594-3.

in addition, the implementation shall declare whether it supports:

- service specific administrative points different from autonomous administrative points;

- the context feature within search-rules;
- the families of entries facility within search-rules, which also requires general conformance to that feature;
- the search relaxation feature within search-rules detailed as above in s), which also requires that the implementation claims general conformance to the search relaxation feature;
- hierarchical groups within search-rules.

v) if conformance is claimed for the use of systems management for administration of the Directory, have the capability of performing the related procedures as defined in ITU-T Rec. X.530 | ISO/IEC 9594-10 for the managed objects for which conformance is claimed;

w) if conformance is claimed to Rule-Based Access Control, have the capability of holding ACI items that conform to the definition of Rule-Based Access Control;

x) if conformance is claimed to integrity of Directory operations, be capable of signing all directory operations supported;

y) if conformance is claimed to integrity and confidentiality of Directory operations, be capable of signing and encrypting all directory operations supported;

z) if conformance is claimed to integrity of directory information in storage be capable to supporting the **attributeValueIntegrityInfoContext** to protect directory information;

aa) if conformance is claimed to cryptographic protection of Directory information in storage be able to encrypt attributes for which conformance is claimed; and

bb) conform to clause 8 of ITU-T Rec.X.509 | ISO/IEC 9594-8 for the Certificate and CRL extensions for which conformance was claimed in 13.2.1 ee).

13.2.3 Dynamic requirements

A DSA shall:

a) if claiming conformance to any application-contexts defined in 7.2.2, 7.2.3 and 7.2.4, conform to the mapping onto used OSI services defined in clause 8;

b) conform to the procedures for distributed operation of the Directory related to referrals, as defined in ITU-T Rec. X.518 | ISO/IEC 9594-4;

c) if conformance is claimed to the application-context specified by **directoryAccessAC** and/or associated with the **dap-ip** protocol, conform to the procedures of ITU-T Rec. X.518 | ISO/IEC 9594-4 as they relate to the referral mode of the DAP;

d) if conformance is claimed to the application-context specified by **directorySystemAC** and/or associated with the **dsp-ip** protocol, conform to the referral mode of interaction, as defined in ITU-T Rec. X.518 | ISO/IEC 9594-4;

e) if conformance is claimed to the chained mode of interaction, conform to the chained mode of interaction, as defined in ITU-T Rec. X.518 | ISO/IEC 9594-4;

NOTE – Only in this case is it necessary for a DSA to be capable of invoking operations of the **directorySystemAC** and/or **dsp-ip**.

f) conform to the rules of extensibility procedures defined in 12.2;

g) if conformance is claimed to Basic Access Control, have the capability of protecting information within the DSA in accordance with the procedures of Basic Access Control;

h) if conformance is claimed to Simplified Access Control, have the capability of protecting information within the DSA in accordance with the procedures of Simplified Access Control;

i) if conformance is claimed for the **shadowOperationalBindingID**, conform to the procedures of ITU-T Rec. X.525 | ISO/IEC 9594-9 and ITU-T Rec. X.501 | ISO/IEC 9594-2 as they relate to the DOP;

j) if conformance is claimed for the **specificHierarchicalBindingID**, conform to the procedures of ITU-T Rec. X.518 | ISO/IEC 9594-4 and ITU-T X.501 | ISO/IEC 9594-2 as they relate to specific hierarchical operational bindings;

k) if conformance is claimed for the **non-specificHierarchicalBindingID**, conform to the procedures of ITU-T Rec. X.518 | ISO/IEC 9594-4 and ITU-T Rec. X.501 | ISO/IEC 9594-2 as they relate to non-specific hierarchical operational bindings;

l) if conformance is claimed for the use of contexts in RDNs, conform to name resolution involving contexts as defined in 9.4 of ITU-T Rec. X.501 | ISO/IEC 9594-2, and 10.3, 10.4, 10.6, 10.9, 10.10 and 15.5.4 of ITU-T Rec. X.518 | ISO/IEC 9594-4;

m) if conformance is claimed to Rule-based Access Control, have the capability of protecting information within the DSA in accordance with the procedures of Rule-based Access Control;

n) if conformance is claimed to basic administration of services, have the capability of handling the search-rules as specified in 19.3.2 of ITU-T Rec. X.518 | ISO/IEC 9594-4.

13.3 Conformance by a shadow supplier

A DSA implementation claiming conformance to this Directory Specification in the role of shadow supplier shall satisfy the requirements specified in 13.3.1 through 13.3.3.

13.3.1 Statement requirements

The following shall be stated:

a) The application context(s) for which conformance is claimed as a shadow supplier: **shadowSupplierInitiatedAC**, **shadowConsumerInitiatedAC**, **shadowSupplierInitiatedAsynchronousAC**, **shadowConsumerInitiatedAsynchronousAC**, **reliableShadowSupplierInitiatedAC**, **reliableShadowConsumerInitiatedAC**, and **disp-ip**.

A DSA implementation claiming conformance as a shadow supplier and not supporting **disp-ip** shall, at a minimum, support either the **shadowSupplierInitiatedAC** or the **shadowConsumerInitiatedAC**. If the DSA supports the **shadowSupplierInitiatedAC**, it may optionally support one or both of the **shadowSupplierInitiatedAsynchronousAC** or **reliableShadowSupplierInitiatedAC**. If the DSA supports the **shadowConsumerInitiatedAC**, it may optionally support one or both of the **shadowConsumerInitiatedAsynchronousAC** or **reliableShadowConsumerInitiatedAC**. If claiming conformance to **disp-ip**, it shall be stated whether the implementation is capable of invoking the **requestShadowUpdate** operation, responding to a **coordinateShadowUpdate**, or both.

b) The security-level(s) for which conformance is claimed (none, simple, strong).

c) To which degree the **UnitOfReplication** is supported. Specifically, which (if any) of the following optional features are supported:

- entry filtering on **objectClass**;
- selection/Exclusion of attributes via **AttributeSelection**;
- the inclusion of subordinate knowledge in the replicated area;
- the inclusion of extended knowledge in addition to subordinate knowledge;
- selection/Exclusion of attribute values based on contexts.

13.3.2 Static requirements

A DSA shall:

a) have the capability of supporting the application-contexts whose abstract syntaxes are defined in clause 7, and the IDM protocols defined in clause 10, for which conformance is claimed;

b) provide support for **modifyTimestamp** and **createTimestamp** operational attributes.

13.3.3 Dynamic requirements

A DSA shall:

a) if claiming conformance to any application-contexts defined in 7.2.3, conform to the mapping onto used OSI services defined in clause 8;

b) conform to the procedures of ITU-T Rec. X.525 | ISO/IEC 9594-9 as they relate to the DISP.

13.4 Conformance by a shadow consumer

A DSA implementation claiming conformance to this Directory Specification as a shadow consumer shall satisfy the requirements specified in 13.4.1 through 13.4.3.

13.4.1 Statement requirements

The following shall be stated:

a) The application context(s) for which conformance is claimed as a shadow consumer: **shadowSupplierInitiatedAC**, **shadowConsumerInitiatedAC**, **shadowSupplierInitiatedAsynchronousAC**, **shadowConsumerInitiatedAsynchronousAC**, **reliableShadowSupplierInitiatedAC**, **reliableShadowConsumerInitiatedAC**, and **disp-ip**.

A DSA implementation claiming conformance as a shadow consumer and not supporting **disp-ip** shall, at a minimum, support either the **shadowSupplierInitiatedAC** or the **shadowConsumerInitiatedAC**. If the DSA supports the **shadowSupplierInitiatedAC**, it may optionally support one or both of the **shadowSupplierInitiatedAsynchronousAC** or **reliableShadowSupplierInitiatedAC**. If the DSA supports the **shadowConsumerInitiatedAC** it may optionally support one or both of the **shadowConsumerInitiatedAsynchronousAC** or **reliableShadowConsumerInitiatedAC**. If claiming conformance to **disp-ip**, it shall be stated whether the implementation is capable of responding to the **requestShadowUpdate** operation, requesting a **coordinateShadowUpdate**, or both.

b) the security-level(s) for which conformance is claimed (none, simple, strong);

c) whether the DSA can act as a secondary shadow supplier (i.e. participate in secondary shadowing as an intermediate DSA);

d) whether the DSA supports shadowing of overlapping units of replication.

13.4.2 Static requirements

A DSA shall:

a) have the capability of supporting the application-contexts whose abstract syntaxes are defined in clause 7, and the IDM protocols defined in clause 10, for which conformance is claimed;

b) provide support for **modifyTimestamp** and **createTimestamp** operational attributes if overlapping units of replication is supported;

c) provide support for the **copyShallDo** service control.

13.4.3 Dynamic requirements

A DSA shall:

a) if claiming conformance to any application-contexts, conform to the mapping onto used OSI services defined in clause 8;

b) conform to the procedures of ITU-T Rec. X.525 | ISO/IEC 9594-9 as they relate to the DISP.

Annex A

DAP in ASN.1

(This annex forms an integral part of this Recommendation | 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, **"DirectoryAccessProtocol"**.

```
DirectoryAccessProtocol {joint-iso-itu-t ds(5) module(1) dap(11) 4}
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

-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

        directoryAbstractService, protocolObjectIdentifiers
                FROM  UsefulDefinitions  {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4}

-- from ITU-T Rec. X.511 | ISO/IEC 9594-3

        abandon, addEntry, compare, directoryBind, directoryUnbind, list, modifyDN,
        modifyEntry, read, removeEntry, search
                FROM DirectoryAbstractService directoryAbstractService

-- from ITU-T Rec. X.519 | ISO/IEC 9594-5

        id-ac-directoryAccessAC, id-as-directoryAccessAS, id-contract-dap,
        id-package-dapConnection, id-package-modify, id-package-read,
        id-package-search, id-rosObject-dapDSA, id-rosObject-directory, id-rosObject-dua
                FROM ProtocolObjectIdentifiers protocolObjectIdentifiers

-- from ITU-T Rec. X.880 | ISO/IEC 13712-1

        Code, CONNECTION-PACKAGE, CONTRACT, OPERATION, OPERATION-PACKAGE,
        ROS-OBJECT-CLASS
                FROM Remote-Operations-Information-Objects
                        {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}

        Bind{}, InvokeId, ROS{}, Unbind{}
                FROM Remote-Operations-Generic-ROS-PDUs
                        {joint-iso-itu-t remote-operations(4) generic-ROS-PDUs(6) version1(0)}

-- from ITU-T Rec. X.881 | ISO/IEC 13712-2

        APPLICATION-CONTEXT
                FROM Remote-Operations-Information-Objects-extensions {joint-iso-itu-t
                        remote-operations(4) informationObjects-extensions(8) version1(0)}
```

```
-- from ITU-T Rec. X.882 | ISO/IEC 13712-3

        acse, pData
                FROM Remote-Operations-Realizations
                        {joint-iso-itu-t remote-operations(4) realizations(9) version1(0)}

        acse-abstract-syntax
                FROM Remote-Operations-Abstract-Syntaxes {joint-iso-itu-t remote-operations(4)
                        remote-operations-abstract-syntaxes(12) version1(0)}        ;

-- application contexts --

directoryAccessAC  APPLICATION-CONTEXT  ::=  {
        CONTRACT                        dapContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directoryAccessAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-directoryAccessAC }

-- ROS objects --

dua  ROS-OBJECT-CLASS  ::=  {
        INITIATES       { dapContract }
        ID              id-rosObject-dua }

directory  ROS-OBJECT-CLASS  ::=  {
        RESPONDS        { dapContract }
        ID              id-rosObject-directory }

dap-dsa  ROS-OBJECT-CLASS  ::=  {
        RESPONDS        { dapContract }
        ID              id-rosObject-dapDSA }

-- contracts --

dapContract  CONTRACT  ::=  {
        CONNECTION                  dapConnectionPackage
        INITIATOR CONSUMER OF       { readPackage | searchPackage | modifyPackage }
        ID                          id-contract-dap }

-- connection package --

dapConnectionPackage  CONNECTION-PACKAGE  ::=  {
        BIND        directoryBind
        UNBIND      directoryUnbind
        ID          id-package-dapConnection }

--  read package --

readPackage  OPERATION-PACKAGE  ::=  {
        CONSUMER INVOKES            { read | compare | abandon }
        ID                          id-package-read }

--  search package --

searchPackage  OPERATION-PACKAGE  ::=  {
        CONSUMER INVOKES            { list | search }
        ID                          id-package-search }

--  modify Package  --

modifyPackage  OPERATION-PACKAGE  ::=  {
        CONSUMER INVOKES            { addEntry | removeEntry | modifyEntry | modifyDN }
        ID                          id-package-modify }
```

```
-- abstract syntaxes --

directoryAccessAbstractSyntax  ABSTRACT-SYNTAX  ::=  {
        DAP-PDUs
        IDENTIFIED BY    id-as-directoryAccessAS }

DAP-PDUs  ::=  CHOICE  {
        basicRos     ROS { { DAP-InvokeIDSet }, { DAP-Invokable }, { DAP-Returnable } },
        bind         Bind { directoryBind },
        unbind       Unbind { directoryUnbind } }

DAP-InvokeIDSet  ::=  InvokeId (ALL EXCEPT absent:NULL)

DAP-Invokable  OPERATION  ::=    { read | compare | abandon |
                                 list | search |
                                 addEntry | removeEntry | modifyEntry | modifyDN }

DAP-Returnable  OPERATION  ::=   { read | compare | abandon |
                                 list | search |
                                 addEntry | removeEntry | modifyEntry | modifyDN }

--  remote operation codes --

id-opcode-read              Code        ::=    local : 1
id-opcode-compare           Code        ::=    local : 2
id-opcode-abandon           Code        ::=    local : 3
id-opcode-list              Code        ::=    local : 4
id-opcode-search            Code        ::=    local : 5
id-opcode-addEntry          Code        ::=    local : 6
id-opcode-removeEntry       Code        ::=    local : 7
id-opcode-modifyEntry       Code        ::=    local : 8
id-opcode-modifyDN          Code        ::=    local : 9

-- remote error codes  --

id-errcode-attributeError               Code        ::=    local : 1
id-errcode-nameError                    Code        ::=    local : 2
id-errcode-serviceError                 Code        ::=    local : 3
id-errcode-referral                     Code        ::=    local : 4
id-errcode-abandoned                    Code        ::=    local : 5
id-errcode-securityError                Code        ::=    local : 6
id-errcode-abandonFailed                Code        ::=    local : 7
id-errcode-updateError                  Code        ::=    local : 8

-- remote error code for DSP --

id-errcode-dsaReferral                  Code        ::=    local : 9

END  -- DirectoryAccessProtocol
```

Annex B

DSP in ASN.1

(This annex forms an integral part of this Recommendation | 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, "**DirectorySystemProtocol**".

```
DirectorySystemProtocol {joint-iso-itu-t ds(5) module(1) dsp(12) 4}
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

-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

        distributedOperations, protocolObjectIdentifiers
                FROM UsefulDefinitions  {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4}

-- from ITU-T Rec. X.518 | ISO/IEC 9594-4

        chainedAbandon, chainedAddEntry, chainedCompare, chainedList, chainedModifyDN,
        chainedModifyEntry, chainedRead, chainedRemoveEntry, chainedSearch,
        dSABind, dSAUnbind
                FROM DistributedOperations distributedOperations

-- from ITU-T Rec. X.519 | ISO/IEC 9594-5

        id-ac-directorySystemAC, id-as-directorySystemAS,
        id-contract-dsp, id-package-chainedModify, id-package-chainedRead,
        id-package-chainedSearch, id-package-dspConnection, id-rosObject-dspDSA
                FROM ProtocolObjectIdentifiers protocolObjectIdentifiers

-- from ITU-T Rec. X.880 | ISO/IEC 13712-1

        Code, CONNECTION-PACKAGE, CONTRACT, OPERATION, OPERATION-PACKAGE,
        ROS-OBJECT-CLASS
                FROM Remote-Operations-Information-Objects
                        {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}

        Bind{}, InvokeId, ROS{}, Unbind{}
                FROM Remote-Operations-Generic-ROS-PDUs
                        {joint-iso-itu-t remote-operations(4) generic-ROS-PDUs(6) version1(0)}

-- from ITU-T Rec. X.881 | ISO/IEC 13712-2

        APPLICATION-CONTEXT
                FROM Remote-Operations-Information-Objects-extensions {joint-iso-itu-t
                        remote-operations(4) informationObjects-extensions(8) version1(0)}
```

```
-- from ITU-T Rec. X.882 | ISO/IEC 13712-3

        acse, pData
                FROM Remote-Operations-Realizations
                        {joint-iso-itu-t remote-operations(4) realizations(9) version1(0)}

        acse-abstract-syntax
                FROM Remote-Operations-Abstract-Syntaxes {joint-iso-itu-t remote-operations(4)
                        remote-operations-abstract-syntaxes(12) version1(0)}        ;

-- application contexts --

directorySystemAC  APPLICATION-CONTEXT  ::=  {
        CONTRACT                        dspContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directorySystemAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-directorySystemAC }

-- ROS objects --

dsp-dsa  ROS-OBJECT-CLASS  ::=  {
        BOTH        { dspContract }
        ID                  id-rosObject-dspDSA }

-- contracts --

dspContract  CONTRACT  ::=  {
        CONNECTION          dspConnectionPackage
        OPERATIONS OF       { chainedReadPackage | chainedSearchPackage | chainedModifyPackage }
        ID                  id-contract-dsp }

-- connection package --

dspConnectionPackage  CONNECTION-PACKAGE  ::=  {
        BIND        dSABind
        UNBIND      dSAUnbind
        ID          id-package-dspConnection }

--  chained read package --

chainedReadPackage  OPERATION-PACKAGE  ::=  {
        OPERATIONS      { chainedRead | chainedCompare | chainedAbandon }
        ID              id-package-chainedRead }

--  chained search package --

chainedSearchPackage  OPERATION-PACKAGE  ::=  {
        OPERATIONS      { chainedList | chainedSearch }
        ID              id-package-chainedSearch }

--  chained modify package  --

chainedModifyPackage  OPERATION-PACKAGE  ::=  {
        OPERATIONS      { chainedAddEntry | chainedRemoveEntry |
                        chainedModifyEntry | chainedModifyDN }
        ID              id-package-chainedModify }
```

```
-- abstract syntaxes --

directorySystemAbstractSyntax  ABSTRACT-SYNTAX  ::=  {
        DSP-PDUs
        IDENTIFIED BY    id-as-directorySystemAS }

DSP-PDUs  ::=  CHOICE  {
        basicRos    ROS { {DSP-InvokeIDSet }, { DSP-Invokable }, { DSP-Returnable } },
        bind        Bind { dSABind },
        unbind      Unbind { dSAUnbind } }

DSP-InvokeIDSet  ::=  InvokeId (ALL EXCEPT absent:NULL)

DSP-Invokable  OPERATION  ::=    { chainedRead | chainedCompare | chainedAbandon |
                                 chainedList | chainedSearch |
                                 chainedAddEntry | chainedRemoveEntry | chainedModifyEntry |
                                 chainedModifyDN }

DSP-Returnable  OPERATION  ::=   { chainedRead | chainedCompare | chainedAbandon |
                                 chainedList | chainedSearch |
                                 chainedAddEntry | chainedRemoveEntry | chainedModifyEntry |
                                 chainedModifyDN }

END  -- DirectorySystemProtocol
```

Annex C

DISP in ASN.1

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

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

```
DirectoryInformationShadowProtocol {joint-iso-itu-t ds(5) module(1) disp(16) 4}
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

-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

        directoryShadowAbstractService, protocolObjectIdentifiers
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4}

-- from ITU-T Rec. X.519 | ISO/IEC 9594-5

        id-ac-shadowConsumerInitiatedAC, id-ac-shadowConsumerInitiatedAsynchronousAC,
        id-ac-shadowSupplierInitiatedAC, id-ac-shadowSupplierInitiatedAsynchronousAC,
        id-ac-reliableShadowConsumerInitiatedAC, id-ac-reliableShadowSupplierInitiatedAC,
        id-as-directoryReliableShadowAS, id-as-directoryShadowAS, id-as-reliableShadowBindingAS,
        id-contract-shadowConsumer, id-contract-shadowSupplier, id-package-dispConnection,
        id-package-shadowConsumer, id-package-shadowSupplier, id-rosObject-initiatingConsumerDSA,
        id-rosObject-initiatingSupplierDSA, id-rosObject-respondingSupplierDSA,
        id-rosObject-respondingConsumerDSA
                FROM ProtocolObjectIdentifiers protocolObjectIdentifiers

-- from ITU-T Rec. X.525 | ISO/IEC 9594-9

        coordinateShadowUpdate, dSAShadowBind, dSAShadowUnbind, requestShadowUpdate,
        updateShadow
                FROM DirectoryShadowAbstractService directoryShadowAbstractService

-- from ITU-T Rec. X.880 | ISO/IEC 13712-1

        Code, CONNECTION-PACKAGE, CONTRACT, OPERATION,
        OPERATION-PACKAGE, ROS-OBJECT-CLASS
                FROM Remote-Operations-Information-Objects
                        {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}

        Bind{}, InvokeId, ROS{}, Unbind{}
                FROM Remote-Operations-Generic-ROS-PDUs
                        {joint-iso-itu-t remote-operations(4) generic-ROS-PDUs(6) version1(0)}
```

```
-- from ITU-T Rec. X.881 | ISO/IEC 13712-2

        APPLICATION-CONTEXT
                FROM Remote-Operations-Information-Objects-extensions {joint-iso-itu-t
                        remote-operations(4) informationObjects-extensions(8) version1(0)}


-- from ITU-T Rec. X.882 | ISO/IEC 13712-3

        acse, association-by-RTSE, pData, transfer-by-RTSE
                FROM Remote-Operations-Realizations
                        {joint-iso-itu-t remote-operations(4) realizations(9) version1(0)}

        acse-abstract-syntax
                FROM Remote-Operations-Abstract-Syntaxes {joint-iso-itu-t remote-operations(4)
                        remote-operations-abstract-syntaxes(12) version1(0)}


-- from ITU-T Rec. X.218 | ISO/IEC 9066-1

        RTSE-apdus
                FROM Reliable-Transfer-APDUs {joint-iso-itu-t reliable-transfer(3) apdus(0)}   ;

-- application contexts --

shadowSupplierInitiatedAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowSupplierContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directoryShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-shadowSupplierInitiatedAC }

shadowSupplierInitiatedAsynchronousAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowSupplierContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               {acse-abstract-syntax |
                                        directoryShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-shadowSupplierInitiatedAsynchronousAC }

shadowConsumerInitiatedAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowConsumerContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               { acse-abstract-syntax | directoryShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-shadowConsumerInitiatedAC }

shadowConsumerInitiatedAsynchronousAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowConsumerContract
        ESTABLISHED BY                  acse
        INFORMATION TRANSFER BY         pData
        ABSTRACT SYNTAXES               {acse-abstract-syntax |
                                        directoryShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-shadowConsumerInitiatedAsynchronousAC }

reliableShadowSupplierInitiatedAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowSupplierContract
        ESTABLISHED BY                  association-by-RTSE
        INFORMATION TRANSFER BY         transfer-by-RTSE
        ABSTRACT SYNTAXES               { acse-abstract-syntax |
                                        reliableShadowBindingAbstractSyntax |
                                        directoryReliableShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-reliableShadowSupplierInitiatedAC }
```

```
reliableShadowConsumerInitiatedAC  APPLICATION-CONTEXT  ::= {
        CONTRACT                        shadowConsumerContract
        ESTABLISHED BY                  association-by-RTSE
        INFORMATION TRANSFER BY         transfer-by-RTSE
        ABSTRACT SYNTAXES               { acse-abstract-syntax |
                                          reliableShadowBindingAbstractSyntax |
                                          directoryReliableShadowAbstractSyntax }
        APPLICATION CONTEXT NAME        id-ac-reliableShadowConsumerInitiatedAC }

-- ROS objects --

initiating-consumer-dsa  ROS-OBJECT-CLASS  ::= {
        INITIATES       { shadowConsumerContract }
        ID              id-rosObject-initiatingConsumerDSA }

responding-supplier-dsa  ROS-OBJECT-CLASS  ::= {
        RESPONDS        { shadowConsumerContract }
        ID              id-rosObject-respondingSupplierDSA }

initiating-supplier-dsa  ROS-OBJECT-CLASS  ::= {
        INITIATES       { shadowSupplierContract }
        ID              id-rosObject-initiatingSupplierDSA }

responding-consumer-dsa  ROS-OBJECT-CLASS  ::= {
        RESPONDS        { shadowSupplierContract }
        ID              id-rosObject-respondingConsumerDSA }

-- contracts --

shadowConsumerContract  CONTRACT  ::= {
        CONNECTION                      dispConnectionPackage
        INITIATOR CONSUMER OF           { shadowConsumerPackage }
        ID                              id-contract-shadowConsumer }

shadowSupplierContract  CONTRACT  ::= {
        CONNECTION                      dispConnectionPackage
        RESPONDER CONSUMER OF           { shadowSupplierPackage }
        ID                              id-contract-shadowSupplier }

-- connection package --

dispConnectionPackage  CONNECTION-PACKAGE  ::= {
        BIND                    dSAShadowBind
        UNBIND                  dSAShadowUnbind
        ID                      id-package-dispConnection }

-- packages --

shadowConsumerPackage  OPERATION-PACKAGE  ::= {
        CONSUMER INVOKES  { requestShadowUpdate }
        SUPPLIER INVOKES  { updateShadow }
        ID                id-package-shadowConsumer }

shadowSupplierPackage  OPERATION-PACKAGE  ::= {
        SUPPLIER INVOKES        { coordinateShadowUpdate |
                                  updateShadow }
        ID                      id-package-shadowSupplier }

-- abstract syntaxes --

directoryShadowAbstractSyntax  ABSTRACT-SYNTAX  ::= {
        DISP-PDUs
        IDENTIFIED BY   id-as-directoryShadowAS }
```