The drawings contained in this Recommendation have been done in AUTOCAD

Recommendation X.518

THE DIRECTORY - PROCEDURES FOR DISTRIBUTED OPERATION [1)]

(Melbourne, 1988)

CONTENTS

SECTION 1 - *Introduction*

0 *Introduction*

1 *Scope and field of application*

2 *References*

3 *Definitions*

4 *Abbreviations*

5 *Notation*

SECTION 2 - *Overview*

6 *Overview*

SECTION 3 - *Distributed directory models*

7 *Distributed directory system model*

8 DSA interactions model

8.1 Chaining

8.2 Multicasting

8.3 Referral

8.4 Mode determination

9 *Directory distribution*

10 *Knowledge*

10.1 Minimal knowledge references

10.2 Root context

10.3 Knowledge references

10.4 Knowledge administration

SECTION 4 - *DSA abstract service*

11 *Overview of DSA abstract service*

12 *Information types*

12.1 Introduction

12.2 Information types defined elsewhere

12.3 Chaining arguments

---

1) Recommendation X.518 and ISO 9594-4, Information Processing Systems - Open Systems Interconnection - The Directory - Procedures for Distributed Operation, were developed in close collaboration and are technically aligned.

12.4 Chaining results
12.5 Operation progress
12.6 Trace information
12.7 Reference type
12.8 Access point
12.9 Continuation reference
13 *Abstract-bind and abstract-unbind*
13.1 DSA bind
13.2 Directory unbind
14 *Chained abstract-operations*
15 *Chained abstract-errors*
15.1 Introduction
15.2 DSA referral
SECTION 5 - *Distributed operations procedures*
16 *Introduction*
16.1 Scope and limits
16.2 Conceptual model
16.3 Individual and cooperative operation of DSAs
17 *Distributed directory behaviour*
17.1 Cooperative fulfillment of operations
17.2 Phases of operation processing
17.3 Managing distributed operations
17.4 Other considerations for distributed operation
17.5 Authentication of distributed operations
18 *DSA behaviour*
18.1 Introduction
18.2 Overview of the DSA behaviour
18.3 Specific operations
18.4 Operation dispatcher
18.5 Looping
18.6 Name resolution procedure
18.7 Object evaluation procedures
18.8 Result merging procedure
18.9 Procedures for distributed authentication
Annex A - ASN.1 for distributed operations
Annex B - Modelling of knowledge
Annex C - Distributed use of authentication
Annex D - Distributed directory object identifiers

0 Introduction

0.1 This document, together with the others of the series, has been produced to facilitate the interconnection of information processing systems to provide directory services. The set of all such systems, together with the directory information which they hold, can be viewed as an integrated whole, called the Directory. The information held by the Directory, collectively known as the Directory Information Base (DIB), is typically used to facilitate communication between, with or about objects such as OSI application entities, people, terminals, and distribution lists.

0.2 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.

0.3 This Recommendation specifies the procedures by which the distributed components of the Directory interwork in order to provide a consistent service to its users.

1 Scope and field of application

1.1 This Recommendation specifies the behaviour of DSAs taking part in the distributed Directory application. The allowed behaviour has been designed so as to ensure a consistent service given a wide distribution of the DIB across many DSAs.

1.2 The Directory is not intended to be a general purpose database system, although it may be built on such systems. It is assumed that there is a considerably higher frequency of queries than of updates.

2 References

Recommendation X.200 - Open Systems Interconnection - Basic Reference Model

Recommendation X.208 -Open Systems Interconnection - Specification of Abstract Syntax Notation (ASN.1)

Recommendation X.500 - The Directory - Overview of Concepts, Models and Services

Recommendation X.501 - The Directory - Models

Recommendation X.511 - The Directory - Abstract Service Definition

Recommendation X.519 - The Directory - Protocol Specifications

Recommendation X.520 - The Directory - Selected Attribute Types

Recommendation X.521 - The Directory - Selected Object Classes

Recommendation X.407 - Message Handling Systems - Abstract Service Definition Conventions

3 Definitions

The definitions contained in this paragraph make use of the abbreviations defined in 4.

3.1 *OSI Reference Model Definitions*

This Recommendation makes use of the following term defined in X.200:

a) application entity title.

3.2 *Basic Directory Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.500:

a) (the) Directory;

b) Directory Information Base.

3.3 *Directory Model Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.501:

a) access point;

b) alias;

c) distinguished name;

d) Directory Information Tree;

e) Directory System Agent;

f) Directory User Agent;

g) relative distinguished name.

3.4 *Abstract Service Definition Conventions*

This Recommendation makes use of the following terms defined in X.407:

a) abstract error;

b) abstract operation;

c) result.

3.5 *Distributed Operation Definitions*

This Recommendation makes use of the following terms, as defined here:

a) chaining: a mode of interaction optionally used by a DSA which cannot perform an operation itself. The DSA chains by invoking an operation of another DSA and then relaying the outcome to the original requestor;

b) context prefix: the sequence of RDNs leading from the Root of the DIT to the initial vertex of a naming context, corresponds to the distinguished name of that vertex;

c) cross reference: a knowledge reference containing information about the DSA that holds an entry. This is used for optimisation. The entry need have no superior or subordinate relationship;

d) DIB fragment: the portion of the DIB that is held by one DSA, comprising one or more naming contexts;

e) distributed name resolution: the process by which name resolution is performed in more than one DSA;

f) internal reference: a knowledge reference containing an internal pointer to an entry held in the same DSA;

g) knowledge information: the information which a particular DSA has about the entries it holds and how to locate other entries in the directory;

h) knowledge reference: knowledge which associates, either directly or indirectly, a DIT entry with the DSA in which it is located;

i) knowledge tree: the conceptual model of the knowledge information that a DSA holds to enable it to perform distributed name resolution;

j) multicasting: a mode of interaction which may optionally be used by a DSA which cannot perform an operation itself. The DSA multicasts the operation, i.e. invokes the same operation of several other DSAs (in series or in parallel) and passes an appropriate outcome to the original requestor;

k) name resolution: the process of locating an entry by sequentially matching each RDN in a purported name to a vertex of the DIT;

l) naming context: a partial sub-tree of the DIT which starts at a vertex and extends downwards to leaf and/or non-leaf vertices. Such vertices constitute the border of the naming context. Non-leaf vertices belonging to the border denote the start of further naming contexts;

m) non-specific subordinate reference: a knowledge reference that holds information about the DSA that holds one or more unspecified subordinate entries;

n) operation progress: a set of values which denotes the extent to which name resolution has taken place;

o) reference path: a continuous sequence of knowledge references;

p) referral: an outcome which can be returned by a DSA which cannot perform an operation itself, and which identifies one or more other DSAs more able to perform the operation;

q) request decomposition: decomposition of a request into subrequests each accomplishing a part of the distributed operation;

r) root context: the naming context for the vertex whose name comprises the empty sequence of RDNs;

s) subordinate reference: a knowledge reference containing information about the DSA that holds a specific subordinate entry;

t) subrequest: a request generated by request decomposition;

u) superior reference: a knowledge reference containing information about the DSA that holds a superior entry.

4 Abbreviations

The following abbreviations are used in this Recommendation:

DIB	Directory Information Base
DIT	Directory Information Tree
DSA	Directory System Agent
DUA	Directory User Agent
RDN	Relative Distinguished Name

5 Notation

The notation used in this paragraph is defined as follows:

a) the data syntax notation, encoding and macro notation are defined in Recommendation X.208;

b) the notations for abstract models and abstract services are defined in Recommendation X.407.

SECTION 2 - *Overview*

6 Overview

The Directory Abstract Service allows the interrogation, retrieval and modification of Directory information in the DIB. This service is described in terms of the abstract Directory object as specified in Recommendation X.511.

Necessarily, the specification of the abstract Directory object does not in any way address the physical realization of the Directory, in particular it does not address the specification of Directory System Agents (DSA) within which the DIB is stored and managed, and through which the service is provided. Furthermore, it does not consider whether the DIB is centralized, i.e. contained within a single DSA, or distributed over a number of DSAs. Consequently, the requirements for DSAs to have knowledge of, navigate to, and cooperate with other DSAs, in order to support the abstract service in a distributed environment, is also not covered by the service description.

This Recommendation specifies the refinement of the abstract Directory object, the refinement being expressed in terms of a set of one or more DSA objects which collectively constitute the distributed directory service. Inherent in this is the identification and specification of the DSA ports that are internal to the Directory object. For each such port, this Recommendation specifies the associated abstract services and its procedures.

In addition, this Recommendation specifies the permissible ways in which the DIB may be distributed over one or more DSAs. For the limiting case where the DIB is contained within a single DSA, the Directory is in fact centralized; for the case where the DIB is distributed over two or more DSAs, knowledge and navigation mechanisms are specified which ensure that the whole of the DIB is potentially accessible from all DSAs that hold constituent entries.

Additionally, request handling interactions are specified that enable particular operational characteristics of the Directory to be controlled by its users. In particular, the user has control over whether a DSA, responding to a directory enquiry pertaining to information held in other DSA(s), has the option of interrogating the other DSA(s) directly (chaining/multicasting) or, whether it should respond with information about other DSA(s) which could further progress the enquiry (referral).

Generally, the decision by a DSA to chain/multicast or refer is determined by the service controls set by the user, and by the DSA's own administrative, operational, or technical circumstances.

Recognizing that, in general, the Directory will be distributed, that directory enquiries will be satisfied by an arbitrary number of cooperating DSAs which may arbitrarily chain/multicast or refer according to the above criteria, this Recommendation specifies the appropriate procedures to be effected by DSAs in responding to distributed directory enquiries. These procedures will ensure that users of the distributed Directory service perceive it to be both user-friendly and consistent.

SECTION 3 - Distributed directory models

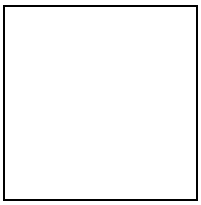
7 Distributed directory system model

The Directory abstract service as defined in Recommendation X.511 models the directory as an object which provides a set of directory services to its users. The services of the directory are modelled in terms of ports, where each port provides a particular set of directory services. Users of the directory access its services through an access point. The directory may have one or more access points and each access point is characterized by the services it provides and the mode of interaction used to provide these services.

This paragraph addresses the internal structure of the directory object, identifying its constituent objects and their ports, and thereby facilitates the specification of a distributed directory service.

Figure 1/X.518 illustrates the distributed directory which will be used as the basis for specifying the distributed aspects of the directory. It illustrates the directory object as comprising a set of one or more DSA-objects.

FIGURE 1/X.518 - T0706790-89



DSA objects are specified in detail in the subsequent clauses of this Recommendation. This clause merely states a number of their characteristics in order to serve as an introduction and to establish the relationship between this Recommendation and other Recommendations.

DSA objects are defined in order that distribution of the DIB can be accommodated and that a number of physically distributed DSAs can interact in a prescribed, cooperative manner to provide directory services to the users of the directory (DUAs).

DSA objects, like the Directory object, are characterized by their externally visible ports. The ports associated with a DSA-object are of two types: service-ports and chained-service-ports.

The service-ports of a DSA object are identical to those of the Directory object, namely, read, search and modify. Figure 1/X.518 illustrates that the service-ports associated with a DSA object constitute an access-point through which directory services are made available.

The detailed specification of the **read**, **search**, and **modify** service-ports of the DSA object can be found in Recommendation X.511. (The protocol specification for the corresponding OSI application service elements, as derived from these port definitions, can be found in Recommendation X.519.)

In addition to the service-ports of the DSA object which accommodate access to the Directory object, a second set of ports are defined, the chained-service-ports. These permit inter-DSA communication in order that the Directory abstract service can be realized in a distributed environment.

The chained-service-ports and the operations provided through them are in direct correspondence to the similarly named service-ports, and are, respectively, **chainedRead**, **chainedSearch**, and **chainedModify**.

The process of specifying the constituent objects of a more abstract object is termed "refinement". The specification of the refinement of the Directory object into its component parts (the DSAs), and the specification of the abstract service provided by each of them (the DSA Abstract Service) is contained in Section Four of this Recommendation. The protocol specification of the corresponding OSI application service elements, as derived from the chained port definitions, can be found in Recommendation X.519.

8 DSA interactions model

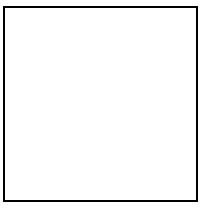
A basic characteristic of the Directory is that, given a distributed DIB, a user should potentially be able to have any service request satisfied (subject to security, access control and administrator policies) irrespective of the access point at which the request originates. In accommodating this requirement it is necessary that any DSA involved in satisfying a particular service request have some knowledge (as specified in 10 of this Recommendation) of where the requested information is located and either return this knowledge to the requestor or attempt to have the request satisfied on its behalf. (The requestor may either be a DUA or another DSA: in the latter case both DSAs must have a chained port.)

Three modes of DSA interaction are defined to meet these requirements, namely "chaining", "multicasting", and "referral". "Chaining" and "multicasting" are defined to meet the latter of the above requirements whilst referrals address the former.

8.1 *Chaining*

This mode of interaction (depicted in Figure 2/X.518) may be used by one DSA, to pass on a request to another DSA when the former has knowledge about naming contexts held by the latter. Chaining may be used to contact a single DSA pointed to in a cross reference, a subordinate reference, or a superior reference. Multicasting is a form of chaining, described in 8.2.

FIGURE 2/X.518 - T0704500-88



Note - In Figure 2/X.518, the order of interactions is defined by the numbers associated with the interaction lines.

8.2 *Multicasting*

This mode of interaction (depicted in Figures 3a/X.518 and 3b/X.518) may be used by a DSA, to chain an identical request in parallel (a) or sequential (b) to one or more DSAs, when the former does not know the complete naming contexts held by the other DSAs. Multicasting is only used by a DSA to contact other DSAs pointed to in a non-specific subordinate reference. Each of the DSAs is passed the identical request. Normally, during name resolution, only one of the DSAs will be able to continue processing the remote operation, all of the others returning the unableToProceed ServiceError. However, during the evaluation phase of search and list operations, all DSAs in a non- specific subordinate reference should be able to continue processing the request.

Note - In Figures 3a/X.518 and 3b/X.518, the order of interactions is defined by the numbers associated with the interaction lines.

FIGURE 3a/X.518 - T0704510-88

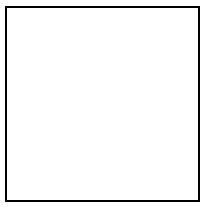
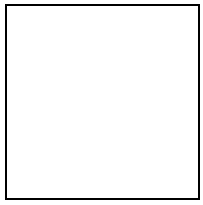


FIGURE 3b/X.518 - T0704520-88



8.3 *Referral*

A referral (depicted in Figures 4a/X.518 and 4b/X.518) is returned by a DSA in its response to a request which it had been requested to perform, either by a DUA, or by another DSA (in which case both DSAs must have a chained-service port). The referral may constitute the whole response (in which case it is categorized as an error) or just part of the response. The referral contains a knowledge reference, which may be either a superior, subordinate, cross or non-specific subordinate reference.

The DSA (Figure 4a/X.518) receiving the referral may use the knowledge reference contained therein, to subsequently chain or multicast (depending upon the type of reference) the original operation to other DSAs. Alternatively, a DSA receiving a referral, may in turn pass the referral back in its response. A DUA (Figure 4b/X.518) receiving a referral may use it to contact one or more other DSAs to progress the request.

FIGURE 4a/X.518 - T0704530-89

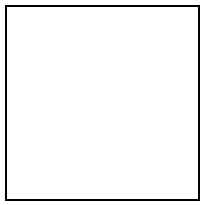
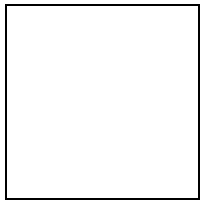


FIGURE 4b/X.518 - T0704540-89



Note - In Figures 4a/X.518 and 4b/X.518, the order of interactions is defined by the numbers associated with the interaction lines.

8.4 *Mode determination*

If a DSA cannot itself fully resolve a request, it must chain/multicast the request (or a request formed by decomposing the original one), to another DSA, unless:

a) chaining is prohibited by the user via the service controls, in which case the DSA must return a referral or a chainingRequired ServiceError (at its choice), or

b) the DSA has administrative, operational, or technical reasons for preferring not to chain, in which case the DSA must return a referral.

Note 1 - A "technical reason" for not chaining/multicasting is that the DSA identified in the knowledge reference has no chained service ports.

Note 2 - If the localScope service control is set, then the DSA (or DMD) must either resolve the request or return an error.

Note 3 - If the user prefers referrals, the user should set chainingProhibited.

9 Directory distribution

This paragraph defines the principles according to which the DIB can be distributed.

Each entry within the DIB is administered by one, and only one, DSA's Administrator who is said to have administrative authority for that entry. Maintenance and management of an entry must take place in a DSA administered by the administrative authority for the entry.

Although the Directory does not provide any support for the replication of entries, it is nevertheless possible to realize replication in two ways:

- Copies of an entry may be stored in other DSA(s) through bilateral agreement. The means by which these copies are maintained and managed is a function of the bilateral agreement and is not defined in this Recommendation.
- Copies of an entry may be acquired by storing (locally and dynamically) a copy of an entry which results from a request.

Note - The acquisition of cache entries is subject to access control.

The originator of the request is informed (via fromCopy) as to whether information returned in response to a request is from a replicated entry or not. A service control, dontUseCopy, is defined which allows the user to prohibit the use of replicated entries.

Each DSA within the Directory holds a fragment of the DIB. The DIB fragment held by a DSA is described in terms of the DIT and comprises one or more naming contexts. A naming context is a partial subtree of the DIT defined as starting at a vertex and extending downwards to leaf and/or non-leaf vertices. Such vertices constitute the border of the naming context. Subordinates of the non-leaf vertices belonging to the border denote the start of further naming contexts.

It is possible for a DSA's administrator to have administrative authority for several disjoint naming contexts. For every naming context for which a DSA has administrative authority, it must logically hold the sequence of RDNs which lead from the root of the DIT to the initial vertex of the subtree comprising the naming context. This sequence of RDNs is called the context prefix.

A DSA's administrator may delegate administrative authority for any immediate subordinates of any entry held locally to another DSA. A DSA that delegated authority is called a superior DSA and the context that holds the superior entry of one for which the administrative authority was delegated, is called the superior naming context. Delegation of administrative authority begins with the root and proceeds downwards in the DIT; that is, it can only occur from an entry to its subordinates.

Figure 5/X.518 illustrates a hypothetical DIT logically partitioned into five naming contexts (named A, B, C, D and E), which are physically distributed over three DSAs (DSA1, DSA2, and DSA3).

From the example it can be seen that the naming contexts held by particular DSAs may be configured so as to meet a wide range of operational requirements. Certain DSAs may be configured to hold those entries that represent higher level naming domains within some logical part(s) of the DIB, the organizational structure of a large company say, but not necessarily all the subordinate entries. Alternatively, DSAs may be configured to hold only those naming contexts representing primarily leaf entries.

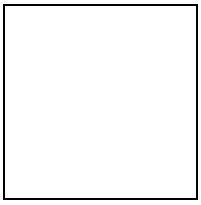
From the above definitions, the limiting case for a naming context can be either a single entry or the whole of the DIT.

Whilst the logical to physical mapping of the DIT onto DSAs is potentially arbitrary, the task of information location and management is simplified if the DSAs are configured to hold a small number of naming contexts.

In order for a DUA to begin processing a request it must hold some information, specifically the presentation address, about at least one DSA that it can contact initially. How it acquires and holds this information is a local matter.

During the process of modification of entries it is possible that the directory may become inconsistent. This will be particularly likely if modification involves aliases or aliased objects which may be in different DSAs. The inconsistency must be corrected by specific administrator action, for example to delete aliases if the corresponding aliased objects have been deleted. The Directory continues to operate during this period of inconsistency.

FIGURE 5/X.518 - T0704550-88



Note - The Root is not held by any DSA, however some indication must exist at the local level to distinguish those vertices (e.g. C = VV, C = WW) which are immediate subordinates of the Root.

10 Knowledge

The DIB is potentially distributed across multiple DSAs with each DSA holding a DIB fragment; the principles that govern distribution of the DIB are specified in 9 of this Recommendation.

It is a requirement of the Directory that, for particular modes of user interaction, the distribution of the directory be rendered transparent, thereby giving the effect that the whole of the DIB appears to be within each and every DSA.

In order to support the operational requirements described above, it is necessary that each DSA holding a fragment of the DIB be able to identify and optionally interact with other fragments of the DIB held by other DSAs.

This paragraph defines knowledge as the basis for the mapping of a name to its location within a fragment of the DIT.

Conceptually DSAs hold two types of information:

a) Directory Information;

b) Knowledge Information.

Directory Information is the collection of entries comprising the Naming Context(s) for which the Administrator of a particular DSA has Administrative Authority.

Knowledge Information embodies the Naming Context(s) held by a particular DSA and denotes how these fit into the overall DIT hierarchy. Name Resolution, the process of locating the DSA which has Administrative Authority for a particular entry given that entry's name, is based on knowledge information.

A Context Prefix is the sequence of RDNs leading from the Root of the DIT to the initial vertex of a naming context and corresponds to the distinguished name of that vertex.

A Naming Context comprises a collection of knowledge references and a Context Prefix. A Naming Context must contain exactly the following knowledge references:

- All the internal references which define the internal structure of the portion of the DIT included in the Naming Context.
- All the subordinate and non-specific subordinate references to other Naming Contexts.

10.1 *Minimal knowledge references*

It is a property of the Directory that each entry can be accessed independently of where a request is generated.

To accomplish this, each DSA shall at least maintain the following knowledge references:

- subordinate references as defined in 10.3.2 and/or non-specific subordinate references as defined in 10.3.5; and
- superior references as defined in 10.3.3.

It is then possible to establish a reference path, as a continuous sequence of knowledge references, to all naming contexts within the Directory.

Optionally, cross references, as defined in 10.3.4 may form part of a reference path to optimize performance.

10.2 *Root context*

Because of the autonomy of the different countries or global organizations, there is likely to be no "single" DSA which holds the root context. The functionality of a "root-DSA" concerning the name resolution process has to be provided by those DSAs which have administrative authority for naming contexts that are immediately subordinate to the root. These DSAs are called First Level DSAs. Each First Level DSA must be able to simulate the functionality of the "root-DSA". This requires full knowledge about the root naming context. The root context is replicated onto each First Level DSA and therefore has to be administered commonly by the autonomous first level administrative authorities. Administration procedures have to be determined by multilateral agreements outside the scope of this Recommendation.

- Each first level DSA shall hold the root context, which implies a reference path to each other first level DSA.
- Each non-first level DSA shall have a superior reference, which implies a reference path to any arbitrary first level DSA.

10.3 *Knowledge references*

The knowledge possessed by a DSA is defined in terms of a set of one or more knowledge references where each reference associates, either directly or indirectly, entries of the DIB with DSAs which hold those entries.

To be able to fulfill the requirements to reach every DIB entry from any DSA, every DSA is required to have knowledge about the entries which it itself holds, and about subordinates and possibly superiors thereof. This gives rise to the following types of knowledge references:

- Internal references
- Subordinate references
- Superior references
- Non-specific subordinate references.

Additionally, for optimization purposes the following type of optional reference is defined:

- Cross references

In the event that the set of knowledge references associated with a particular DSA contain only internal references, the DSA has no knowledge of other DSAs and the DIB is therefore centralized.

10.3.1 *Internal references*

An internal reference consists of:

- the RDN corresponding to a DIB entry;

- an internal pointer to where the entry is stored in the local DIB. (The specification of this pointer is outside the scope of this Recommendation.)

All entries for which a particular DSA has Administrative Authority are represented by internal references in the knowledge information of that DSA.

10.3.2 *Subordinate references*

A subordinate reference consists of:

- an RDN corresponding to an immediate subordinate DIB entry;
- the Access Point of the DSA to which Administrative Authority for that entry was delegated.

All subordinate entries held by another DSA to which this DSA has delegated Administrative Authority, must be represented by subordinate references (or non-specific subordinate references as described in 10.3.5).

10.3.3 *Superior references*

A superior reference consists of:

- the Access Point of a DSA.

Each non-first level DSA maintains precisely one superior reference. The superior reference shall form part of a reference path to the root. Unless some method outside of the standard is employed to ensure this, for example within a DMD, this shall be accomplished by referring to a DSA which holds a naming context whose context prefix has less RDNs than the context prefix with fewest RDNs held by this DSA.

If a new non-first level DSA is introduced, it must have a minimal initial knowledge, which is represented by the superior reference. Any further knowledge will be added by subordinate references or cross references (as described in 10.3.4). If a new first level DSA is introduced, it must acquire the root context and advise all other first level DSAs. How this is accomplished is outside the scope of this Recommendation.

10.3.4 *Cross reference*

A cross reference consists of:

- a Context Prefix;
- the Access Point of a DSA which has Administrative Authority for that Naming Context.

This type of reference is optional and serves to optimize Name Resolution. A DSA may hold any number (including zero) of cross references.

10.3.5 *Non-specific subordinate references*

A non-specific subordinate reference consists of:

- The Access Point of a DSA which holds one or more immediately subordinate Naming Contexts.

This type of reference is optional, to allow for the case in which a DSA is known to contain some subordinate entries but the specific RDNs of those entries is not known.

For each naming context which it holds, a DSA may hold any number (including zero) of non- specific subordinate references, which will be evaluated if all specific internal and subordinate references have been pursued. DSAs accessed via a non-specific reference must be able to resolve the request directly (either success or failure). In the event of failure a ServiceError reporting a problem of unableToProceed is returned to the requestor.

10.4 *Knowledge administration*

To operate a widely distributed Directory with an acceptable degree of consistency and performance, procedures are required to maintain and extend the knowledge held by each DSA. The same procedures are appropriate for creating initial knowledge.

Knowledge can be maintained by:

a) The DSA or its administrative authority propagating changes of knowledge to those DSAs holding all kinds of references to it, whenever changes at that DSA cause the references to become invalid. This is the only way superior, subordinate and non-specific subordinate references can be maintained.

b) DSAs requesting and obtaining cross references to improve the performance through ordinary directory operations.

This Recommendation does not define any procedures for propagating knowledge changes as described in a). Bilateral agreements must be established locally for this.

10.4.1 *Requesting cross reference*

To improve the performance of the Directory System, the local set of cross references can be expanded using ordinary Directory operations. If a DSA has a chained port it may request another DSA (which also must have a chained port) to return those knowledge references which contain information about the location of naming contexts related to the target object name of an ordinary Directory operation.

If the **returnCrossReference** component of the **ChainingArgument** is set to **TRUE**, the **crossReference** component of the **ChainingResult** may be present, consisting of a sequence of cross reference items.

If a DSA is not able to chain a request to the next DSA a referral is returned to the originating DSA. If the **returnCrossReference** component of the chaining argument was **TRUE**, the referral may contain additionally the context prefix of the naming context which the referral refers to. The **contextPrefix** component is absent if the referral is based on a non-specific subordinate reference. The cross reference returned by a referral is only based on knowledge held by the DSA which generated the referral.