INTERNATIONAL TELECOMMUNICATION UNION

CCITT X.650

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

DATA COMMUNICATION NETWORKS OPEN SYSTEMS INTERCONNECTIONS (OSI) SYSTEM ASPECTS

OPEN SYSTEMS INTERCONNECTION (OSI) – REFERENCE MODEL FOR NAMING AND ADDRESSING

Recommendation X.650

Geneva, 1992

FOREWORD

The CCITT (the International Telegraph and Telephone Consultative Committee) is a permanent organ of the International Telecommunication Union (ITU). CCITT is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The Plenary Assembly of CCITT which meets every four years, establishes the topics for study and approves Recommendations prepared by its Study Groups. The approval of Recommendations by the members of CCITT between Plenary Assemblies is covered by the procedure laid down in CCITT Resolution No. 2 (Melbourne, 1988).

Recommendation X.650 was prepared by Study Group VII and was approved under the Resolution No. 2 procedure on the 17 of January 1992.

CCITT NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication Administration and a recognized private operating agency.

© ITU 1992

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the ITU.

Recommendation X.650

OPEN SYSTEMS INTERCONNECTION (OSI) – REFERENCE MODEL FOR NAMING AND ADDRESSING[1)]

CONTENTS

Introduction

1 *Scope*

2 *References*

3 *Definitions*

4 *Abbreviations*

5 *Basic concepts of naming*

6 *OSI naming and addressing concepts and the correct use of addresses*

- 6.1 The naming of real open systems
- 6.2 The naming and addressign of elements of an (N)-layer
- 6.3 The correct use of addresses

7 *OSI addressing model*

- 7.1 Associations between peer (N)-entities
- 7.2 Attachment of (N)-entities to (N)-SAPs
- 7.3 (N)-addresses and (N)-SAPs
- 7.4 (N)-directory-functions and directory facilities

8 *Addressing information and (N)-services*

- 8.1 Introduction
- 8.2 Address parameters
- 8.3 Called-(N)-address
- 8.4 Calling-(N)-address
- 8.5 Responding-(N)-address

9 *Addressing information and (N)-protocols*

- 9.1 Introduction
- 9.2 Addressing information in (N)-PAI
- 9.3 Assignment of values to elements of (N)-PAI

---

1) Recommendations X.650 and ISO/IEC 7498-3 [Information Processing Systems – Open Systems Interconnection – Basic Reference Model – Part 3: Naming and Addressign] were developed in close collaboration and are technically aligned.

9.4 Network-addresses and network-PAI

9.5 (N)-addresses and (N)-PAI above the Network Layer

9.6 Obtaining (N)-PAI

10 *(N)-directory functions*

10.1 Introduction

10.2 The initiator (N)-directory-functions

10.3 The recipient (N)-directory-functions

11 *Addressing in specific OSI layers*

11.1 Application processes and the Application Layer

11.2 Presentation Layer

11.3 Session Layer

11.4 Transport Layer

11.5 Network Layer

11.6 Data Link Layer

11.7 Physical Layer

12 *Naming domains and authorities*

13 *Registration procedures for naming within OSI*

14 *Directory facility requirements*

14.1 Introduction

14.2 The application title directory facility

14.3 The network address directory facility

Introduction

This Recommendation extends the basic architectural concepts of identifiers described in Recommendation X.200.

This Recommendation states the architectural principles which are followed in the production ot any Recommendation which involves the identification (naming) and location (addressing) of objects for the purpose of interconnection within the Open System Interconnection Environment (OSIE).

This Recommendation has sufficient flexibility to accommodate advances in technology and expansion in user demands. This flexibility is also intended to allow the phased transition from existing implementations to OSI Recommendations.

Note – This Recommendation is expected to be subject to future expansion, in particular with regard to Multi-Peer Data Transmission (MPDT).

The architectural principles stated within this Recommendation will ensure that any CCITT Recommendation that involves the identification and location of objects within the OSIE for the purpose of interconnection will:

a) avoid any restrictions on:

1) the functionality that may be made available through current of future Recommendations,

2) the functionality of any real open system,

3) the internal design of any real open system;

b) preserve the principle of layer independence in the OSIE, i.e. the internal functioning of one layer is not constrained by any other layer;

c) preserve the principle of implementation independence in the OSIE, as expressed in 4.2 of Recommendation X.200. That is, no real open system (or administrator thereof) is required to know anything about the implementation design of any other real open system (or administration thereof), nor does any real open system impose such knowledge as a condition for communication using OSI Recommendations;

d) allow economical support for interconnection within the OSIE; in particular individual standards produced within the framework specified by this Recommendation should make it possible to provide facilities which give adequate levels of performance, reliability, and integrity and which ease the administration by humans with respect to identifying and locating objects within the OSIE for the purpose of interconnection.

The description of naming and addressing for the OSIE given in this Recommendation is developed in stages.

Note – This Recommendation provides clarifications of the basic architecture defined in Recommendation X.200 where this is necessary for a full understanding of the naming and addressing requirements within the OSIE.

1 Scope

This Recommendation:

a) defines general mechanisms for the use of names and addresses to identify and locate objects in the OSIE, and

b) defines the use of these mechanisms within the layered structure of the Basic Reference Model.

This Recommendation extends the concepts and principles defined in CCITT Recommendation X.200/ ISO 7498. This Recommendation is not intended to be either an implementation specification or a basis for appraising the conformance of actual implementations.

The specific form of names and addresses is not within the scope of this Recommendation.

2 References

2.1 *Paired CCITT Recommendations/International Standards equivalent in technical content*

Rec. X.200 (1988) – *Reference Model for Open Systems Interconnection for CCITT Applications.*

ISO 7498: 1984 – *Information Processing Systems – Open Systems Interconnection – Basic Reference Model.*

Rec. X.213 (1988) – *Network Service Definition for Open Systems Interconnection for CCITT Applications.*

ISO 8348/Add.2: 1990 – *Information Processing Systems – Data Communications – Network Service Definition – ADDENDUM 2: Network Layer Addressing.*

Rec. X.210 (1988) – *Open Systems Interconnection Layer Service Definition Conventions.*

ISO TR 8509: 1987 – *Information Processing Systems – Open Systems Interconnection – Service Conventions.*

2.2 *Additional references*

ISO/IEC 9545: 1989 – *Information Technology – Open Systems Interconnection – Application Layer Structure.*

ISO/IEC 7498-4: 1989 – *Information Processing Systems – Open Systems Interconnection – Basic Reference Model – Part 4: Management Framework.*

3 Definitions

3.1 This Recommendation makes use of the following terms defined in ISO/IEC 9545:

a) application-process-type;

b) application-process-invocation.

3.2 This Recommendation makes use of the following terms defined in CCITT Recommendation X.210/ ISO TR 8509:

a) (N)-service-request-primitive,

b) (N)-service-indication-primitive,

c) (N)-service-response-primitive,

d) (N)-service-confirm-primitive.

3.3 This Recommendation makes use of the following term defined in CCITT Rec. X.213/ISO/IEC 8348/Add 2:

a) subnetwork point of attachment.

3.4 For the purpose of this Recommendation, the following definitions apply:

3.4.1 (N)-address

A name unambiguous within the OSIE which is used to identify a set of (N)-service-access-points which are all located at a boundary between an (N)-subsystem and an (N+1)-subsystem in the same open system.

Note 1 – This definition of (N)-address is different from that in CCITT Rec. X.200/ISO 7498. This definition is the definitive one and will be moved to CCITT Rec. X.200/ISO 7498 to replace the existing definition when CCITT Rec. X.200/ISO 7498 is revised.

Note 2 – A name is unambiguous within a given scope when it identifies one and only one object within that scope. Unambiguity of a name does not preclude the existence of synonyms.

3.4.2 (N)-address-selector; (N)-selector

An element of addressing information that identifies a set of (N)-SAPs which are all in the same (N)-subsystem; an (N)-selector value is assigned by the local administration.

Note – The concept of (N)-address-selectors only applies above the Network Layer.

3.4.3 (N)-association

A cooperative relationship among (N)-entity-invocations.

Note – This may be formed by the exchange of (N)-protocol-control-information.

3.4.4 calling-(N)-address

A parameter which may appear in an (N)-service request or indication primitive and which identifies the (N)-address at the (N)-initiator.

Note – In the service definition of a particular layer, such a parameter may be referred to either as a “calling-(N)-address” or “source-address”. Throughout this Specification, however, only the term “calling-(N)-address” is used.

3.4.5 called-(N)-address

A parameter which may appear in an (N)-service request or indication primitive which identifies the (N)-address at the (N)-recipient.

Note – In the service definition of a particular layer, such a parameter may be referred to either as a “called-(N)-address” or “destination-address”. Throughout this Recommendation, however, only the term “called-(N)-address” is used.

3.4.6 **descriptive name**

A name that identifies a set of one or more objects by means of a set of assertions concerning the properties of the objects of the set.

3.4.7 **(N)-directory-function**

An (N)-function that processes (N)-addresses, (N–1)-addresses, (N)-entity-titles, and (N)-PAI to provide mappings among these categories of information.

3.4.8 **(N)-entity**

An active element within an (N)-subsystem embodying a set of capabilities defined for the (N)-layer that corresponds to a specific (N)-entity-type (without any extra capabilities being used.)

Note – This definition of (N)-entity is different from that in CCITT Rec. X.200/ISO 7498. This definition is the definitive one and will be moved to CCITT Rec. X.200/ISO 7498 to replace the existing definition when CCITT Rec. X.200/ISO 7498 is revised.

3.4.9 **(N)-entity-invocation**

A specific utilization of part or all capabilities of a given (N)-entity (without any extra capabilities being used).

Note – This definition will be moved to CCITT Rec. X.200/ISO 7498 to replace the existing definition when CCITT Rec. X.200/ISO 7498 is revised.

3.4.10 **(N)-entity-title**

A name that is used to identify unambiguously an (N)-entity.

3.4.11 **(N)-entity-type**

A description of a class of (N)-entities in terms of a set of capabilities defined for the (N)-layer.

Note – This definition will be moved to CCITT Rec. X.200/ISO 7498 to replace the existing definition when CCITT Rec. X.200/ISO 7498 is revised.

3.4.12 **generic name**

A name of a set of objects.

Note – A generic-title is a specific form of generic name.

3.4.13 **(N)-initiator**

An (N)-entity-invocation which issues an (N–1)-service request primitive.

3.4.14 **name**

A linguistic construct which corresponds to an object in some universe of discourse.

3.4.15 **naming-authority**

A registration authority which allocates names according to specified rules. Where the naming-authority allocates titles, it is known as a title-authority. Where the naming-authority allocates addresses, it is known as an addressing-authority.

3.4.16 **naming-domain**

The set of names that are assignable to objects of a particular type. Where the names are titles, the set is known as a title-domain. Where the names are addresses, the set is known as an addressing-domain.

3.4.17 **naming-subdomain**

A subset of a naming-domain, which is disjoint from all other naming-subdomains of that naming-domain.

3.4.18 **primitive name**

A name that identifies an object and which is assigned by a designated naming-authority. The internal structure of the name is not required to be understood or to have significance to users of the name.

3.4.19 **(N)-recipient**

An (N)-entity-invocation which receives an (N–1)-service indication primitive.

3.4.20 **(N)-protocol-addressing-information; (N)-PAI**

Those elements of (N)-PCI which contain addressing information.

3.4.21 **responding-(N)-address**

A parameter which may appear in an (N)-service response or confirm primitive and which identifies the (N)-address at the (N)-recipient.

Note – In the service definition of a particular layer, such a parameter may be referred to either as a "called-address" or "responding address". Throughout this Recommendation, however, only the term "responding-(N)-address" is used.

3.4.22 **(N)-service-access-point-address; (N)-SAP-address**

An (N)-address that is used to identify a single (N)-SAP.

Note 1 – This definition of (N)-service-access-point-address is different from that in CCITT Rec. X.200/ISO 7498. This definition is the definitive one and will be moved to CCITT Rec. X.200/ISO 7498 to replace the existing definition when CCITT Rec. X.200/ISO 7498 is revised.

Note 2 – (N)-address is the general term which applies to any set of (N)-SAPs, including sets of one and only one (N)-SAP. (N)-SAP address is only used where it is necessary to specify precisely that the address identifies one and only one (N)-SAP. Whether an (N)-address is an (N)-SAP address or not is a matter local to the (N)-subsystem and is not known to other open systems. Nevertheless, at some layers and because of their possible use in subsequent communications, calling-(N)-addresses and responding-(N)-addresses may be constrained to identify a single (N)-SAP (see §§ 8.4.4 and 8.5.5). The decision whether or not to apply this constraint is made on a layer-by-layer and protocol-by protocol basis.

3.4.23 **subnetwork-address**

An identifier assigned to a subnetwork point of attachment by the registration authority of the subnetwork.

3.4.24 **synonymous name; synonym**

A name that identifies an object that is also identified by another distinct name. Synonymous generic names are distinct generic names that name the identical set.

3.4.25 **system-title**

A name, unique within the OSIE, which is used to identify a single real open system.

4 Abbreviations

The following abbreviations apply to this Recommendation:

(N)-CEPI	(N)-Connection-EndPoint-Identifier
DLSAP	Data-Link-Service-Access-Point
NSAP	Network-Service-Access-Point
OSI	Open Systems Interconnection
OSIE	OSI Environment
(N)-PAI	(N)-Protocol-Addressing-Information
(N)-PCI	(N)-Protocol-Control-Information
PhSAP	Physical-Service-Access-Point
PSAP	Presentation-Service-Access-Point
(N)-SAP	(N)-Service-Access-Point
SNPA	SubNetwork Point of Attachment
SSAP	Session-Service-Access-Point
TSAP	Transport-Service-Access-Point

5 Basic Concepts of Naming

5.1 Names are linguistic constructs expressed in some language. They correspond to objects in some universe of discourse. The correspondence between names (in the language) and objects (in the universe of discourse) is the relation of identifying. A name identifies the object to which it is bound.

5.2 Within the context of OSI, names identify particular communications objects in the Open Systems Interconnection Environment (OSIE). There are two distinct kinds of names, primitive and descriptive.

5.3 Within any particular universe of discourse, a primitive name is a name assigned by a naming-authority to a specific object. A naming-authority is simply a source of names. The only architectural constraints imposed upon naming-authorities are that all of the names it provides:

a) are expressed in a prescribed language; and

b) are unambiguous (identify just one object).

5.4 A descriptive name consists of a set of assertions which are expressed in a formally defined language. The definition of the formal language determines those linguistic constructs which are well-formed descriptive names. A descriptive name may be incomplete, in that many objects satisfy all the assertions, or it may be complete, in that it serves to identify a single object. A complete descriptive name is equivalent to a primitive name in that it unambiguously identifies an object. Primitive names may be components of a descriptive name.

5.5 Although a primitive name is unambiguous, there may be more than one name that unambiguously identifies the same object.

5.6 A generic name is a primitive name or a descriptive name that identifies a set comprising more than one object with the intent that, when a generic name is used to denote an object, the result is that exactly one member of the set of objects will be selected. A generic name may be used to identify a set of objects of a particular type, which need not be located in the same open system.

5.7 A title is assigned to an object where the purpose of the name is to discriminate among different objects and to permit retrieval of information associated with an object from a Directory Facility. A title is assigned to an object type where the purpose of the name is to discriminate among different object types and to permit retrieval of information associated with an object type from a Directory Facility. The name may identify a system, application-process, application-process-type, (N)-entity, or (N)-entity-type.

Note – These objects and types are defined in either CCITT Rec. X.200/ISO 7498 or ISO/IEC 9545.

5.8 An identifier is assigned to an object where the purpose of the name is only to discriminate among occurences of this object. The name may identify an (N)-association, an application process-invocation, or an (N)-entity-invocation.

Note – These objects are defined in either CCITT Rec. X.200/ISO 7498 or ISO/IEC 9545.

6 OSI naming and addressing concepts and the correct use of addresses

6.1 *The naming of real open systems*

6.1.1 A system-title is a layer independent primitive name, i.e., the same identifier is used within various layers to identify the same real open system. A single real open system is named by one and only one system-title.

6.1.2 A system-title is used to identify a real open system as a whole. It may also be used:

a) in conjunction with other qualifiers to identify specific OSI resources in the relevant parts of the management information base within the real open system; or

b) as an attribute of a Directory Facility entry pertaining to an OSI resource associated with a single real open system.

6.2 *The naming and addressing of elements of an (N)-layer*

6.2.1 *Introduction*

6.2.1.1 Since an (N)-entity-type describes a class of (N)-entities, it needs to be named, but not located. Since (N)-entities and (N)-entity-invocations are active elements within an (N)-layer, they need to be unambiguously identified and located.

6.2.1.2 Within an open system, (N+1)-entities and (N)-entities are bound together at (N)-service-access-points [(N)-SAPs]. (N)-entities provide services to (N+1)-entities via the exchange of service primitives at (N)-SAPs.

6.2.1.3 An (N)-entity is identified unambiguously by an (N)-entity-title. An (N)-entity-type is identified by an (N)-entity-type-title. An (N)-entity-invocation is identified by an (N)-entity-invocation-identifier which is unambiguous within the scope of an (N)-entity.

6.2.2 *(N)-addresses*

6.2.2.1 An (N)-address identifies a set of (N)-SAPs which are all located at the boundary between an (N)-subsystem and an (N+1)-subsystem. An (N)-SAP-address is an (N)-address which identifies a set containing exactly one (N)-SAP.

6.2.2.2 While (N)-entities are the objects being addressed, the result of communication to an address is communication with an (N)-entity-invocation.

6.2.2.3 An (N+1)-entity is located by its binding to one or more (N)-SAPs. An (N)-SAP is identified by one or more (N)-addresses.

Note – A physical-address is used to access a data-link-entity;
a data-link-address is used to access a network-entity;
a network-address is used to access a transport-entity;
a transport-address is used to access a session-entity;
a session-address is used to access a presentation-entity; and
a presentation-address is used to access an application-entity.

6.2.3 *(N)-selectors*

An (N)-selector is that part of the addressing information which is specific to the (N)-subsystem. (N)-selectors are used to identify (N)-SAPs or sets of (N)-SAPs within an end open system, once this end open system is unambiguously identified. Since the end open system is implicitly known at the Network Layer, (N)-selectors are used

above the Network Layer, along with local information, to address the desired (N+1)-entity within the open system. (N)-selector values are exchanged between open systems as part of the (N)-PAI.

6.3 *The correct use of (N)-addresses*

6.3.1 (N)-addresses have a limited scope. They are used to distinguish among sets of (N)-SAPs, and only (N)-SAPs. Addressing rules are not used to make the structure of a real open system visible to the OSI environment.

6.3.2 (N)-addresses are used to identify sets of (N)-SAPs in order to locate (N+1)-entities. An (N+1)-subsystem is partitioned into (N+1)-entities:

a) to support different (N+1)-protocols or sets of (N+1)-protocols;

b) to accommodate security and/or management requirements; and

c) in the case of the application-subsystem, to distinguish between different application-processes and different application-entities of the same application-process.

6.3.3 (N)-addresses are not used:

a) to distinguish among aspects of protocols that are subject to negotiation (classes, subsets, quality of service, protocol versions) or parameter values;

b) to derive routing information above the Network Layer; or

c) to distinguish among hardware components.

Note – In some configurations, the use of an (N)-address as defined in § 6.3.2 can lead to an (N+1)-entity being wholly contained within a single hardware component. Nevertheless, within the OSIE, the (N)-address identifies the (N+1)-entity; it does not identify the hardware component.

7 OSI addressing model

7.1 *Associations between peer (N)-entities*

7.1.1 An (N)-association is a cooperative relationship between two (N)-entity-invocations. Cooperation between (N)-entity-invocations requires the establishment and maintenance of related state information in each (N)-entity-invocation. This state information supports an (N)-association between the (N)-entity-invocations.

7.1.2 An (N)-entity invocation may support one or more independent (N)-associations at any one time. The communications behavior of the (N)-entity-invocation with respect to a specific (N)-association is defined by the (N)-entity and by the state information which is maintained by the (N)-entity-invocation and is specific to that (N)-association.

7.1.3 An (N)-association-identifier is associated with each (N)-association. This identifier is unique within the scope of a pair of cooperating (N)-entity-invocations. It serves to identify the related state information associated with each (N)-entity-invocation. The identifier has two components, one being determined by each (N)-entity-invocation.

Note – Certain (N)-protocols may not need explicit (N)-association-identifiers.

7.1.4 Two (N)-entity-invocations can establish (N–1)-connection(s), or can make use of an (N–1)-connectionless service, to support an (N)-association. The lifetime of an (N)-association may exceed the lifetime of any supporting (N–1)-connection(s). The binding of an (N)-association with (N–1)-connection(s) may change with time.

Note – An (N)-association could be associated with a sequence of (N–1)-connections with a one-to-one binding at any point in time; alternatively in the case of splitting, there could be a one-to-many binding at any point in time.

7.1.5 When the operation of an (N)-association requires it, (N)-entity-titles are used to identify (N)-entities independent of their locations. When the operation of an (N)-association requires it, (N–1)-addresses are used in requests for (N–1)-services to identify the locations of the (N)-entities concerned.

7.2 *Attachment of (N)-entities to (N)-SAPs*

An (N)-entity may provide (N)-services through one or more (N)-SAPs and may use (N–1)-services through one or more (N–1)-SAPs. In consequence an (N)-entity may have the following relationships with (N)-SAPs and (N–1)-SAPs (see Figure 1/X.650):

a) an (N)-entity may provide (N)-services through one (N)-SAP making use of (N–1)-services through one (N–1)-SAP;

b) an (N)-entity may provide (N)-services through multiple (N)-SAPs making use of (N–1)-services through one (N–1)-SAP;

c) an (N)-entity may provide (N)-services through one (N)-SAP making use of (N–1)-services through multiple (N–1)-SAPs;

d) an (N)-entity may provide (N)-services through multiple (N)-SAPs making use of (N–1)-services through multiple (N–1)-SAPs.

Note 1 – There is no relationship between the SAP/entity correspondences identified above and multiplexing. An (N)-multiplexing function provides for the mapping of several (N)-connections onto a single (N–1)-connection. The (N)-connections may all terminate in a single (N)-SAP or they may terminate in separate (N)-SAPs. Multiplexed (N)-connections are distinguished from each other by elements of (N)-PCI at the service boundary and by elements of the (N)-PAI, e.g. an association-identifier within the (N)-protocol.

Note 2 – Logical channel numbers in CCITT Rec. X.25 and ISO/IEC 8208 and connection references in the OSI Transport Protocol (CCITT Rec. X.224/ISO/IEC 8073), are examples of information exchanged in (N)-PCI to distinguish connections when multiplexing is used.

7.3 *(N)-addresses and (N)-SAPs*

7.3.1 The OSI addressing structure allows:

a) (N)-addresses to identify the location of an (N+1)-entity without constraining the structure of lower layer subsystems in the open system concerned; and

b) multiple (N)-entities to be defined within an (N)-subsystem.

Note – The relevant addressing structure allows a presentation-address to identify the location of an application-entity without constraining the structure of presentation-, session-, and transport-subsystems within an open system; and allows the definition of a single set of addressing information for use in establishing communication with an application-entity in a recipient system.

7.3.2 An (N)-address identifies a set of (N)-SAPs all located at the boundary of a single (N)-subsystem. The exact membership of the set is an issue local to that (N)-subsystem. The set membership is not known to other open systems and may change over time.

7.3.3 The set of (N)-SAPs identified by an (N)-address may consist of:

a) a single (N)-SAP bound to an (N+1)-entity;

b) multiple (N)-SAPs that are bound to a single (N+1)-entity; or

c) multiple (N)-SAPs that are bound to different (N+1)-entities.

7.3.4 When an (N)-address is used as a called-(N)-address in a service primitive, the recipient (N)-subsystem will select a single (N)-SAP from the set identified by the (N)-address. The selection mechanism is a local issue transparent to the (N)-initiator.

7.3.5 Open systems are configured to ensure that all of the (N)-SAPs in the set identified by an (N)-address are bound to (N+1)-entities that are of the same type and, therefore, provide the same functions.

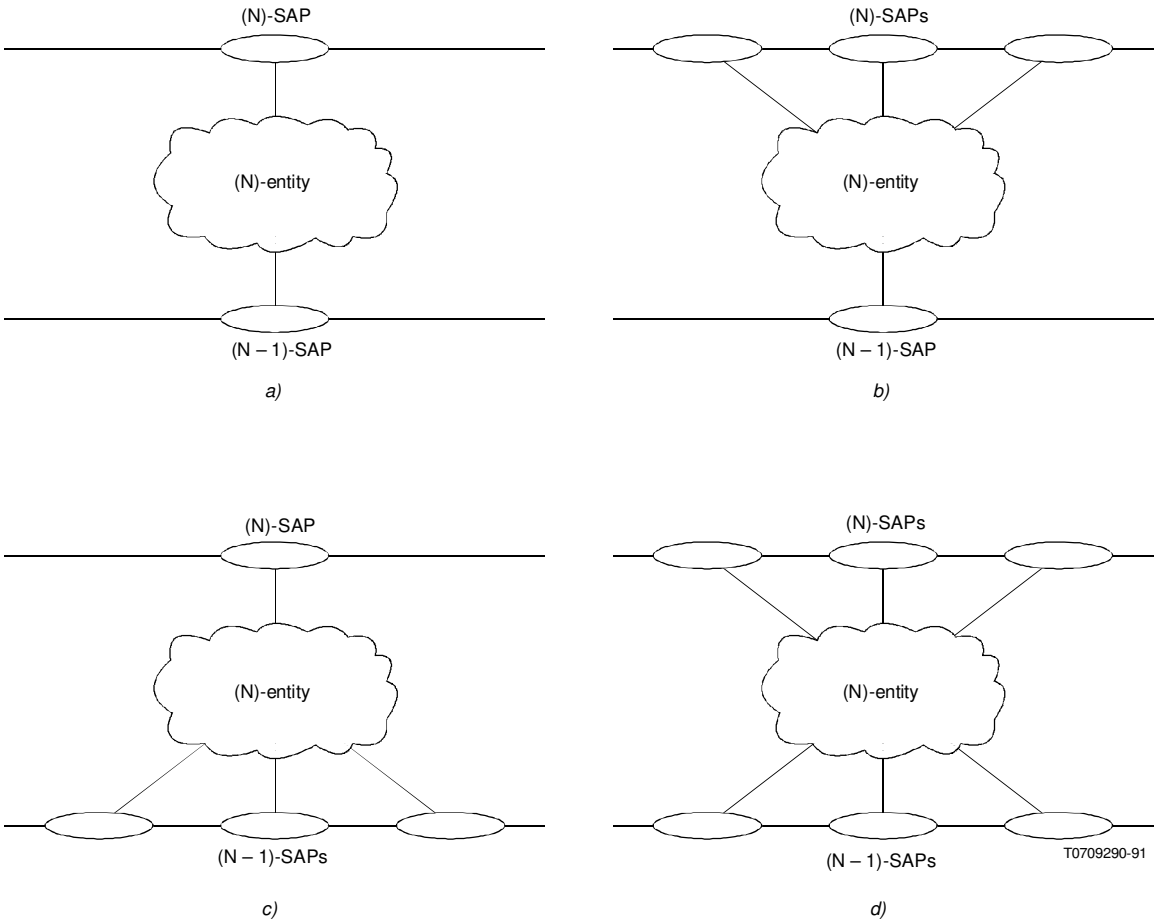

FIGURE 1/X.650

Relationships of (N)-entity to (N)-SAPs and to (N – 1)-SAPs

7.3.6 It is important to distinguish between the semantics of an (N)-address and the syntax used to represent an (N)-address within a given open system. (N)-addresses are passed across layer boundaries within open systems as parameters of (N)-service primitives. For (N)-service request/response primitives, the semantics of (N)-addresses are conveyed to the peer (N)-subsystem and passed across the layer boundary as parameters of the (N)-service indication/confirm primitives. Only the semantics of an (N)-address are conveyed by the (N)-service. The syntax of an (N)-address is a local issue and different representations may be used in different open systems.

7.3.7 When an (N+1)-entity establishes an (N)-connection with another (N+1)-entity, each (N+1)-entity is given an (N)-connection-endpoint-identifier [(N)-CEPI] by its supporting (N)-entity. (See § 5 of CCITT Recommendation X.200/ISO 7498). An (N)-CEPI is a local identifier determined at connection establishment time. An (N)-CEPI cannot be used as a substitute for an (N)-address. In the case that the calling-(N)-address and called-(N)-address on an (N)-connection are identical, the (N)-connection has two (N)-connection-end-points and two (N)-CEPIs (connection of an (N)-entity to itself). How the two (N)-CEPIs in the (N)-subsystems are distinguished is an entirely local matter.

7.4 *(N)-Directory-Functions and Directory Facilities*

7.4.1 (N)-directory-functions process (N)-addresses, (N–1)-addresses, (N)-entity-titles, and (N)-PAI to provide mappings among these categories of information. Information used for these mappings is held by a Directory Facility. It is a local system management responsibility to access the Directory Facility to retrieve the information and make it available to an (N)-directory-function.

7.4..2 Some of this information represents the logical structure of the local end system and influences local operation. This information is stored locally. Other information represents the logical structure of the remote end system and influences the generation of (N)-PAI. This information may be stored locally or remotely. If it is stored remotely, OSI protocols are used to access that information.

8 Addressing information and (N)-services

8.1 *Introduction*

8.1.1 This section provides a layer independent description of the use of (N)-addresses in (N)-service primitives.

8.1.2 (N+1)-entities use (N)-services by issuing (N)-service primitives at (N)-SAPs. Issuing an (N)-service request/response primitive may cause an (N)-service indication/confirm primitive to be issued at an (N)-SAP to which a peer (N+1)-entity is attached.

8.1.3 The (N)-address derived from information provided by a Directory Facility may be invalid. An (N)-address derived from the calling/ responding-(N)-address parameter of a previously received (N)-service indication/confirm primitive has to be valid at the time it is issued, but no guarantee can be given for a subsequent use of this address. Therefore, an (N+1)-entity using an (N)-address should, under all circumstances, check that it has led to communication with the desired correspondent in the (N+1)-layer. It is normally sufficient to do this in the Application Layer by exchanging application-entity-titles.

8.1.4 The use of an (N)-address is not sufficient by itself to identify a particular (N+1)-entity-invocation. An (N+1)-entity may be satisfied to communicate with any (N+1)-entity-invocation of the desired (N+1)-entity at the (N)-address. In some (N+1)-layers, it may be necessary to reference the (N+1)-entity-invocation using the (N+1)-entity-invocation-identifier.

8.2 *Address parameters*

8.2.1 It is important to distinguish between the (N)-addresses passed as called-(N)-address parameters and those passed as calling-(N)-address or responding-(N)-address parameters.

8.2.2 Called-(N)-addresses are used in the initiation of communication between (N+1)-entity-invocations. The (N+1)-initiator provides the called-(N)-address the semantics of which are conveyed to the peer (N+1)-recipient.

8.2.3 Calling- and responding-(N)-addresses are primarily used for identification and recall purposes and may identify the specific (N)-SAPs used in an instance of communication.

8.3 *Called-(N)-address*

8.3.1 The called-(N)-address parameter in connection-mode service primitives is equivalent to the destination (N)-address parameter in connectionless-mode service primitives.

8.3.2 The called-(N)-address is provided by the (N+1)-initiator. The semantics of the (N)-address are conveyed to the recipient (N)-subsystem and passed to the (N+1)-subsystem in an (N)-service indication primitive.

8.3.3 The called-(N)-address conveyed in the (N)-service indication primitive parameter is not restricted to be the same address as was specified in the associated request primitive. However, an (N)-service definition may impose such a restriction.

8.3.4 Above the Network Layer, address processing is confined to end systems:

a) at the initiating open system, the processing of the called-(N)-address is not dependent on the complexity of the address structures supported by the recipient open system; and

b) at the recipient open system, the processing of the called-(N)-address depends on the complexity of the address structures supported by this system.

8.3.5 In the Network Layer, although some processing of the called-(N)-address may occur in an intermediate system, this processing is not dependent on the complexity of the address structures supported by the recipient open system.

8.3.6 The called-(N)-address identifies a set of (N)-SAPs at the recipient-(N)-subsystem. Any one of the (N)-SAPs in this set may be used to support the communication. The resolution of the address to select a particular (N)-SAP is the responsibility of the recipient (N)-subsystem.

8.3.7 The called-(N)-address may have been derived from information obtained from the Directory Facility. In this case, the semantics of the (N)-address are related to a directory entry published on behalf of the recipient system. The attributes associated with the Directory Facility entry are known within the recipient system. The called-(N)-address identifies a set of (N)-SAPs that provide access to (N+1)-entities which support communication in a manner that is consistent with the information obtained from the Directory Facility.

8.3.8 The called-(N)-address may have been previously passed as a calling- or responding-(N)-address parameter by the recipient (N)-subsystem in a previous instance of communication. In this case, the (N)-address identifies the set of (N)-SAPs consistent with the requirements concerning calling- or responding-(N)-addresses described in §§ 8.4 and 8.5.

8.3.9 The called-(N)-address may have been obtained by private arrangement. In this case, the called-(N)-address identifies a set of (N)-SAPs that provide access to (N+1)-entities that support communication in a manner consistent with that private arrangement.

8.4 *Calling-(N)-address*

8.4.1 The calling-(N)-address parameter in connection-mode service primitives is equivalent to the source (N)-address parameter in connectionless-mode service primitives.

8.4.2 The calling-(N)-address is provided by the (N+1)-initiator. The semantics of the (N)-address are conveyed to the recipient (N)-subsystem and passed to the (N+1)-subsystem in an (N)-service indication primitive.

8.4.3 Provided (N)-protocol specifications and OSI management specifications do not impose constraints, the recipient (N+1)-subsystem may use the calling-(N)-address in any of the following ways:

a) as a called-(N)-address in a subsequent request primitive that is not related to the initial communication;

b) as a called-(N)-address in a subsequent request primitive that is related to the initial communication, in order to facilitate connection re-establishment or splitting;

c) as a called-(N)-address which is forwarded to some other open system;

d) for management purposes.

8.4.4 While valid, the calling-(N)-address will identify a set of (N)-SAPs at the initiating (N)-entity. The set of (N)-SAPs identified may be constrained by layer specific requirements concerning calling-(N)-addresses. For example, a layer may require a calling-(N)-address to identify the single (N)-SAP actually used to support the original communication.

8.4.5 When the calling-(N)-address received in an (N)-service indication primitive is used by the recipient (N)-subsystem as a called-(N)-address in a subsequent (N)-service request primitive, this subsystem should be aware of the possibility that this address may be no longer valid in the sense defined in § 8.4.4 and should take appropriate measures.

8.5 *Responding-(N)-address*

8.5.1 The responding-(N)-address is used in (N)-service response/confirm primitives.

Note – In some OSI service definitions the term "called address" is used in response and confirm primitives to denote the responding-(N)-address parameter.

8.5.2 The responding-(N)-address is provided by the (N+1)-recipient. The semantics of the (N)-address are conveyed to the initiating (N)-subsystem and passed to the initiating (N+1)-subsystem in a confirm service primitive.

8.5.3 Provided (N)-layer protocol specifications and OSI management specifications do not impose constraints, the initiating (N+1)-subsystem may use the responding-(N)-address in any of the following ways:

a) as a called-(N)-address in a subsequent request primitive not related to the initial communication;

b) as a called-(N)-address in a subsequent request primitive related to this communication, e.g. to facilitate connection re-establishment or splitting;

c) as a called-(N)-address which is forwarded to some other open system;

d) for management purposes.

8.5.4 The responding-(N)-address may differ from the called-(N)-address specified in the related (N)-service indication primitive.

8.5.5 While valid, the responding-(N)-address will identify a set of (N)-SAPs at the recipient (N)-entity. The set of (N)-SAPs identified may be constrained by layer specific requirements concerning responding-(N)-addresses. For example, a layer may require a responding-(N)-address to identify the single (N)-SAP actually used to support the communication.

8.5.6 When the responding-(N)-address received in an (N)-service confirm primitive is used by the initiating (N)-subsystem as a called-(N)-address in a subsequent (N)-service request primitive, this subsystem should be aware of the possibility that this address may be no longer valid in the sense defined in § 8.5.5 and should take appropriate measures.

9 Addressing information and (N)-protocols

9.1 *Introduction*

This section provides a layer independent description of the use of addressing information in (N)-Protocol-Addressing-Information [(N)-PAI]. (N)-PAI are those elements of (N)-PCI which contain addressing information.

9.2 *Addressing information in (N)-PAI*

9.2.1 The semantics of an (N)-address are conveyed by means of (N)-protocol exchanges between (N)-entity-invocations. For some layers, the full semantics of (N)-addresses are conveyed in (N)-PAI. In other layers, it is not necessary for the full semantics of (N)-addresses to be represented in the (N)-PAI and, for these layers, the full semantics of (N)-addresses are conveyed by a combination of:

a) the exchange of (N)-PAI; and

b) local information about the scope of the (N)-PAI.

Note 1 – For example, the network-entities exchange network-addresses. In this case, the Network-PAI includes the network-address.

Note 2 – The values of (N)-PAI may include information relevant to the operation of both the (N)-layer and the (N+1)-layer. Nevertheless, a given layer only makes use of the information relevant to that layer.

9.2.2 Below the Network Layer, communicating (N)-entities are confined to a single subnetwork. The (N)-PAI exchanged need not be globally applicable, since it can be interpreted within the scope of the subnetwork.

9.2.3 At the Network Layer, communicating (N)-entities may be attached to different subnetworks. Consequently, the (N)-PAI exchanged has to be globally applicable. For this reason, the Network-PAI alone provides for the exchange of the complete semantics of a network-address.

9.2.4 Above the Network Layer, the scope of the (N)-PAI is restricted to the communicating end systems. At these layers, the semantics of the (N)-address comprise:

a) the identification of a set of (N)-SAPs; this identification is unambiguous within the scope of the (N)-subsystem which contains the (N)-SAPs and is provided by (N)-selectors exchanged in (N)-PAI and local information about the scope of the (N)-selectors within the (N)-subsystem; and

b) the identification of the end systems, derived from the Network Layer exchange of network-addresses.

Note – At the Application Layer, titles and identifiers, not addressing information, are exchanged.

9.3 *Assignment of values to elements of (N)-PAI*

9.3.1 Layer protocol specifications define elements of (N)-PAI to be used for the exchange of addressing information. Different elements of (N)-PAI are used to convey the semantics of:

– called-(N)-addresses,

– calling-(N)-addresses, and

– responding-(N)-addresses.

9.3.2 The values of the elements used to convey calling-(N)-address semantics are provided by the (N)-initiator. These values may be retained by the recipient (N)-subsystem and used in a subsequent request primitive to convey called-(N)-address semantics to the original initiating (N)-subsystem.

9.3.3 The values of the elements used to convey responding-(N)-address semantics are provided by the recipient (N)-subsystem. These values may be retained by the initiating (N)-subsystem and used in a subsequent (N)-service request primitive to convey called-(N)-address semantics to the recipient (N)-subsystem.

9.3.4 The values of the elements used to convey called-(N)-address semantics may be obtained:

a) from a Directory Facility,

b) by private arrangement, or

c) from previously issued (calling-) responding-(N)-addresses.

9.4 *Network-Addresses and Network-PAI*

The complete semantics of the network-address is conveyed in the Network-PAI. The network-address is globally applicable and is issued by the appropriate registration authority.

9.5 *(N)-Addresses and (N)-PAI above the Network Layer*

9.5.1 (N)-selectors are unambiguous within the scope of an (N)-subsystem. The values of (N)-selectors are chosen by the local administration of an open system and there is no requirement for an OSI addressing authority, although the chosen values have to be known by systems which want to communicate. When an (N)-selector identifies a set of (N)-SAPs, the resolution of that (N)-selector is the responsibility of the recipient (N)-subsystem.

Note 1 – All (N)-entities refer to a particular set of (N)-SAPs in the same way, (i.e. the (N)-selector value which is associated with this set of (N)-SAPs is the same regardless of the (N)-entity which handles the communication).

Note 2 – The way unambiguity is achieved is a local matter. Local administration of the open system may decide to achieve that goal by defining (N)-selectors that are unique within the scope of the (N)-subsystem. In such a case, the semantics of the (N)-selector is directly derived from the value carried in the (N)-PAI, regardless of the (N)-entity which handles the communication. Where (N)-selectors are unambiguous within the scope of the (N)-subsystem, without being unique within that scope, additional information local to the recipient open system, is necessary (in particular, the semantics of the (N)-selector depend on the (N)-entity which handles the communication).

9.5.2 Protocol specifications may designate (N)-PAI as optional and therefore it may be absent. Since the (N)-PAI in an (N)-protocol is the (N)-selector, the (N)-selector may be absent. There is no distinction between the absence of an (N)-selector and the presence of a nil (N)-selector value in connectionless-mode operation. In connection mode operation, the absence of an (N)-selector is equivalent to the presence of a nil (N)-selector value for calling- and called-(N)-addresses. For responding-(N)-addresses, the absence of an (N)-selector indicates that the responding-(N)-address is equivalent to the called-(N)-address.

Note – Within layers which use the Type-Length-Value (TLV) encoding technique:

a) "absence of selector" means that no parameter of the type used to convey this selector is present;

b) the "nil selector value" corresponds to a zero value for the length field of the parameter which is of the type used to convey this selector; and

c) if the parameter type which corresponds to the type used to convey the selector is present and if the associated parameter length is not zero, then the selector value is not considered as nil whatever the encoding of this value might be.

9.5.3 The nil (N)-selector value (or the absence of a value) is only used in (N)-PAI to convey called-(N)-address semantics when the nil value has been specified:

a) by a value from an entry in a Directory Facility;

b) as the (N)-PAI used to convey the semantics of a previously issued calling/responding-(N)-address; or

c) by private arrangement.

9.5.4 The (N)-recipient uses the nil (N)-selector value according to local information to select an (N)-SAP.

Note – The use of a nil (N)-selector value does not preclude the use of other (N)-selector values by the local administration of an open system.

9.6 *Obtaining (N)-PAI*

9.6.1 Information about application-entities is obtained from the Application Title Directory Facility (see § 14). Included in this information is a single tuple specifying the addressing (N)-PAI values required to access the application-entities through a PSAP. The tuple is of the form:

(P-selector, S-selector, T-selector, list of network-addresses)

Note – Each of the (N)-PAI values derived from the tuple may be used by the relevant (N)-recipient to identify a set of (N)-SAPs. The fact that the (N)-addressing information may identify a set of (N)-SAPs is known only by the recipient (N)-subsystem.

9.6.2 All of the network-addresses of the list belong to a single open system. At the initiating open system, one of the network-address values is selected by local system management for a given instance of communication.

9.6.3 The T-selector is the single T-selector value that, when used in transport-PAI, identifies the set of TSAPs at the open system to which the set of network-addresses in the tuple applies. The selector value is equally valid regardless of which network-address is used.

9.6.4 The S-selector is the single S-selector value that, when used in session-PAI, identifies the set of SSAPs at the open system to which the set of network-addresses in the tuple applies. The selector value is equally valid regardless of which network-address is used.

9.6.5 The P-selector is the single P-selector value that, when used in presentation-PAI, identifies the set of PSAPs at the open system to which the set of network-addresses in the tuple applies. The selector value is equally valid regardless of which network-address is used.

10 (N)-directory-functions

10.1 *Introduction*

10.1.1 (N)-directory-functions process (N)-addresses, (N–1)-addresses, (N)-entity-titles, (N)-PAI, and possibly routing information to provide mappings among these categories of information. These functions are performed by the (N)-entity within the (N)-layer during connection establishment or connectionless data transmission:

a) when an (N)-service request primitive is received from the (N+1)-layer, or an (N–1)-service confirmation primitive is received from the (N–1)-layer [the initiator (N) directory-functions]; and

b) when an (N–1)-service indication primitive is received from the (N–1)-layer, or an (N)-service response primitive is received from the (N+1)-layer [the recipient (N)-directory-functions].

10.1.2 Information on these mappings may be held by local system management and made available for access by (N)-directory-functions or it may be held by a Directory Facility. If information is needed from a Directory Facility, it is obtained by local system management and made available to (N)-directory-functions.

10.2 *The initiator (N)-directory-functions*

10.2.1 The parameters of the initiator (N)-directory-functions used for connection establishment or connectionless data transmission are:

a) the called-(N)-address supplied by the (N+1)-layer, (CALLED-(N)-ADDRESS);

b) the calling-(N)-address supplied by the (N+1)-layer, (CALLING-(N)-ADDRESS);

c) the called-(N)-entity-title supplied by the (N)-layer, (CALLED-(N)-ENTITY-TITLE);

d) the called-(N–1)-address generated by an initiator (N)-directory-function, (CALLED-(N–1)-ADDRESS);

e) the responding-(N)-PAI supplied by the (N–1)-layer, (RESPONDING-(N)-PAI);

f) the responding-(N–1)-address supplied by the (N–1)-layer, (RESPONDING-(N–1)-ADDRESS); and

g) information (LOCAL) made available by local system management to the (N)-directory-functions on loading, quality of service requirements, and other local information.

Note – A layer need not use all of these parameters for its initiator (N)-directory-functions.

10.2.2 Taking these parameters as input the initiator (N)-directory-functions generate the following information:

a) the called-(N)-PAI to be carried in (N)-PCI, (CALLED-(N)-PAI);

b) the calling-(N)-PAI to be carried in (N)-PCI, (CALLING-(N)-PAI);

c) the calling-(N–1)-address to be passed in the (N–1)-service request primitive, (CALLING-(N–1)-ADDRESS);

Note 1 – The choice of the (N–1)-SAP at which this (N–1)-service request primitive is issued is a local matter. This choice has to be consistent with the calling-(N–1)-address.

d) the called-(N–1)-address to be passed in the (N–1)-service-request-primitive, (CALLED-(N–1)-ADDRESS); and

e) the responding-(N)-address to be passed in the (N)-service confirmation primitive, (RESPONDING-(N)-ADDRESS);

f) routing information, (ROUTING INFORMATION).

Note 2 – The nature of the ROUTING INFORMATION and its use by the (N)-layer depend on the detailed architecture of the routing function within the (N)-layer.

10.2.3 There are seven initiator (N)-directory-functions:

a) the Initiator Addressing Function 1 – **IAF1**. For this function:

1) the input parameters are: CALLED-(N)-ENTITY-TITLE and LOCAL,

2) the output is: CALLED-(N–1)-ADDRESS;

b) the Initiator Addressing Function 2 – **IAF2**. For this function:

1) the input parameters are: CALLED-(N)-ADDRESS and LOCAL,

2) the output is: CALLED-(N–1)-ADDRESS;

c) the Initiator Addressing Function 3 – **IAF3**. For this function:

1) the input parameters are: CALLED-(N–1)-ADDRESS, CALLING-(N)-ADDRESS and LOCAL,

2) the output is: CALLING-(N–1)-ADDRESS;

d) the Initiator Addressing Function 4 – **IAF4**. For this function:

1) the input parameters are: RESPONDING-(N–1)-ADDRESS and RESPONDING-(N)-PAI,

2) the output is: RESPONDING-(N)-ADDRESS;

e) the Initiator PAI Function 1 – **IPF1**. For this function:

1) the input parameter is: CALLED-(N)-ADDRESS,

2) the output is: CALLED-(N)-PAI;

f) the Initiator PAI Function 2 – **IPF2**. For this function:

1) the input parameters is: CALLING-(N)-ADDRESS,

2) the output is: CALLING-(N)-PAI;

g) the Initiator Routing Function 1 – **IRF1**. For this function:

1) the input parameters are: CALLED-(N)-ADDRESS and LOCAL,

2) the output is: ROUTING INFORMATION.

10.3 *The recipient (N)-directory-functions*

10.3.1 The parameters of the recipient (N)-directory-functions used for connection establishment or connectionless data transmission are:

a) the called-(N–1)-address supplied by the (N–1)-layer, (CALLED-(N–1)-ADDRESS);

b) the calling-(N–1)-address supplied by the (N–1)-layer, (CALLING-(N–1)-ADDRESS);

c) the called-(N)-PAI carried in the (N)-PCI, (CALLED-(N)-PAI);

d) the calling-(N)-PAI carried in the (N)-PCI, (CALLING-(N)-PAI);

e) the responding-(N)-address supplied by the (N+1)-layer, (RESPONDING-(N)-ADDRESS); and

f) information (LOCAL) locally known, identifying the scope of (N)-PAI.

Note – A layer need not use all of these parameters for its recipient (N)-directory-functions.

10.3.2 Taking these parameters as input the recipient (N)-directory-functions generate the following information:

a) the called-(N)-address to be passed in the (N)-service indication primitive, (CALLED-(N)-ADDRESS);

Note – The choice of the (N)-SAP at which this (N)-service indication primitive is issued is a local matter. This choice must be consistent with the called-(N)-address.

b) the calling-(N)-address to be passed in the (N)-service indication primitive (CALLING-(N)-ADDRESS);

c) the responding-(N–1)-address to be passed in the (N–1)-service response primitive, (RESPONDING-(N–1)-ADDRESS); and

d) the responding-(N)-PAI carried in (N)-PCI, (RESPONDING-(N)-PAI).

10.3.3 There are four recipient (N)-directory-functions.

a) the Recipient Addressing Function 1 – **RAF1**. For this function:

1) the input parameters are: CALLED-(N)-PAI, CALLED-(N–1)-ADDRESS, and LOCAL,

2) the output is: CALLED-(N)-ADDRESS;

b) the Recipient Addressing Function 2 – **RAF2**. For this function:

1) the input parameters are: CALLING-(N)-PAI and CALLING-(N–1)-ADDRESS,

2) the output is: CALLING-(N)-ADDRESS;

c) the Recipient Addressing Function 3 – **RAF3**. For this function:

1) the input parameters are: CALLED-(N–1)-ADDRESS and LOCAL,

2) the output is: RESPONDING-(N–1)-ADDRESS;

d) The Recipient PAI Function 1 – **RPF1**. For this function:

1) the input parameter is: RESPONDING-(N)-ADDRESS,

2) the output is: RESPONDING-(N)-PAI.

11 Addressing in specific OSI layers

11.1 *Application-processes and the Application Layer*

This section is concerned with the naming of elements of Application-processes and the Application Layer. A complete description of these elements is given in ISO/IEC 9545.

11.1.1 *Application-processes and Application Layer elements*

11.1.1.1 Application-processes are identified by application-process-titles that are unambiguous throughout the OSIE. An application-process-title is a single name which, for convenience, may be structured internally. In particular, for some application-processes the internal structure of the application-process-title may be based on system-title.

Note 1 – The purpose of structuring an application-process-title from a system-title is to make it possible to register application-processes within the system on which they reside once its system-title has been registered.

Note 2 – Application-process-titles can have synonyms. That is, an application-process may be known to one or more application-processes by different application-process-titles.

11.1.1.2 Application-entities are identified by application-entity-titles that are unambiguous throughout the OSIE. An application-entity-title is composed of an application-process-title and an application-entity-qualifier. This division into two components permits the user of the application-entity-title to obtain information specific to the application-process or to the application-entity. The application-entity-qualifier is unambiguous within the scope of the application-process. Each application-entity-title is associated with a presentation-address.

Note – Application-entity-titles can have synonyms. That is, an application-entity may be known to one or more application-entities by different application-entity-titles.

11.1.1.3 Where application-process-invocations have to be identified, this is done by means of application-process-invocation-identifiers that are unambiguous within the scope of an application-process. An application-process-invocation is identified unambiguously within the OSIE by the application-process invocation-identifier qualified by the application-process-title.

11.1.1.4 Where application-entity-invocations have to be identified, this is done by means of application-entity-invocation-identifiers that are unambiguous within the scope of a pair: (application-process-invocation, application-entity). An application-entity-invocation is identified unambiguously within the OSIE by the application-entity-invocation-identifier qualified by the application-entity-qualifier, the application-process-invocation-identifier, and the application-process-title (see Table 1/X.650).

TABLE 1/X.650

Summary of Identifiers

Identified by AE / Item	APT	APII	AEQ	AEII
Application Process	+			
Application Process Invocation	+	+		
Application Entity	+		+	
Application Entity Invocation	+	+	+	+

APT application-process-title

APII application-process-invocation-identifier

AEQ application-entity-qualifier

AEII application-entity-invocation-identifier

11.1.1.5 Where application-associations have to be identified, this is done by means of application-association-identifiers that are unambiguous within the scope of the application-entity invocations at the endpoints of the association.

11.1.1.6 Where application-process-types have to be identified, this is done by means of an application-process-type-title that is unambiguous throughout the OSIE.

An application-process-type-title may be used to denote the distributed processing capabilities of an application-process.

11.1.1.7 Where application-entity-types have to be identified, this is done by means of an application-entity-type-title that is unambiguous throughout the OSIE. An application-entity-type-title may be used to denote the communications capabilities of an application-entity.

11.1.1.8 At any instant of time, each application-entity-title is bound to a single presentation-address that identifies the set of PSAPs to which the application-entity is attached. This binding is recorded in the Application Title Directory Facility (see § 14).

11.1.2 *Application-association*

11.1.2.1 In order for an application-entity-invocation to establish an application-association with another application-entity-invocation, it uses the presentation-address of the called application-entity to establish a presentation-connection or to use the presentation connectionless-mode service. This presentation-address may be obtained from the application-directory-function, IAF1, using the called-application-entity-title.

11.1.2.2 If it is necessary to confirm that the required application-entity is still attached to the PSAP identified by the presentation-address, then the initiating application-entity-invocation may pass the called-application-entity-title as a part of the application-PCI exchanged to establish the application-association.

11.1.2.3 Application-entity-invocations may exchange calling- and responding-application-entity-titles for use in future communication. Such titles may be recognized by a recipient system as naming specific responding-application-entities. This is a matter for agreement between the communicating applications.

11.1.2.4 If required in establishing an application-association, the application-entity-invocations may exchange the following identifiers as specification of the application-PCI exchanged to establish the application-association:

– application-process-invocation-identifier,

– application-entity-invocation-identifier,

– application-association-identifier.

11.1.3 *Application Layer use of (N)-directory-functions*

11.1.3.1 In the initiating system, on request from the application-process, the application-entity :

a) uses IAF1 to derive the called-presentation-address (which is passed in a presentation-service request primitive) from the called-application-entity-title and from local information; and

b) uses IAF3 to derive the calling-presentation-address (which is passed in a presentation-service request primitive) and the local PSAP through which the presentation-service request primitive is issued from the called-presentation-address and from local information.

Note 1 – The choice of the PSAP at which this primitive is issued is a local matter. This choice has to be consistent with the calling-presentation-address.

Note 2 – At the Application Layer, the calling-(N)-address parameter does not apply.

11.1.3.2 The initiator application-directory-function, IAF1, yields a single presentation-address. Where a generic application-entity-title (e. g., application-entity-type-title) is used as the called-application-entity-title, resolution of this title to a single presentation-address is the responsibility of the recipient local system.

11.1.3.3 In the recipient system, on receipt of a presentation-service indication primitive, the application-entity uses RAF3 to derive the responding-presentation-address from the called-presentation-address and from local information.

11.2 *Presentation Layer*

11.2.1 In the initiating system, on receipt of a presentation-service request primitive, the presentation-entity:

a) uses IAF2 to derive a called-session-address (which is passed in a session-service request primitive) from the called-presentation-address and from local information;

b) uses IAF3 to derive the calling-session-address (which is passed in a session-service request primitive) and the local SSAP through which the session-service request primitive is issued from the called-session-address, the calling-presentation-address, and local information;

Note – The choice of the SSAP at which this primitive is issued is a local matter. This choice has to be consistent with the calling-session-address.

c) uses IPF1 to derive the called-presentation-selector, (which is sent in presentation-PAI), from the called-presentation-address; and

d) uses IPF2 to derive the calling-presentation-selector, (which is sent in presentation-PAI), from the calling-presentation-address.

11.2.2 In the recipient system, on receipt of a session-service indication primitive, the presentation-entity:

a) uses RAF1 to derive the called-presentation-address from the called-presentation-selector (which is received in presentation-PAI) the called-session-address, and from local information;

Note – Local information may be used to resolve a called-presentation-selector, that refers to a set of PSAPs to a single PSAP.

b) uses RAF2 to derive the calling-presentation-address from the calling-presentation-selector (which is received in presentation-PAI) and from the calling-session-address;

c) uses RAF3 to derive the responding-session-address from the called-session-address and from local information; and

d) uses RPF1 to derive the responding-presentation-selector (which is sent in presentation-PAI) from the responding-presentation-address (which is received in a presentation-service response primitive).

11.2.3 In the initiating system for connection-mode operation, on receipt of a session-service confirmation primitive, the presentation-entity uses IAF4 to derive the responding-presentation-address from the responding-presentation-selector (which is received in presentation-PAI) and from the responding-session-address received from the session-service confirmation primitive.

11.3 *Session Layer*

11.3.1 In the initiating system, on receipt of a session-service request primitive, the session-entity:

a) uses IAF2 to derive a called-transport-address (which is passed in a transport-service request primitive) from the called-session-address and from local information;

b) uses IAF3 to derive the calling-transport-address (which is passed in a transport-service request primitive) and the local TSAP through which the transport-service request primitive is issued from the called-transport-address, the calling-session-address and local information;

Note – The choice of the TSAP at which this primitive is issued is a local matter. This choice has to be consistent with the calling-transport-address.

c) uses IPF1 to derive the called-session-selector (which is sent in session-PAI) from the called-session-address; and

d) uses IPF2 to derive the calling-session-selector (which is sent in session-PAI) from the calling-session-address.

11.3.2 In the recipient system, on receipt of a transport-service indication primitive, the session-entity:

a) uses RAF1 to derive the called-session-address from the called-session-selector (which is received in session-PAI), the called-transport-address, and from local information;

Note – Local information may be used to resolve a called-session-selector that refers to a set of session-SAPs to a single session-SAP.

b) uses RAF2 to derive the calling-session-address from the calling-session-selector (which is received in session-PAI) and from the calling-transport-address;

c) uses RAF3 to derive the responding-transport-address from the called-transport-address and from local information; and

d) uses RPF1 to derive the responding-session-selector (which is sent in session-PAI) from the responding-session-address, which is received in a session-service response primitive.

11.3.3 In the initiating system for connection-mode operation, on receipt of a transport-service confirmation primitive, the session-entity uses IAF4 to derive the responding-session-address from the responding-session-selector (which is received in session-PAI) and from the responding-transport-address which is received in the transport-service confirmation primitive.

11.4 *Transport Layer*

11.4.1 In the initiating system, on receipt of a transport-service request primitive, the transport-entity:

a) uses IAF2 to derive a called-network-address (which is passed in a network-service request primitive) from the called-transport-address and from local information;

b) uses IAF3 to derive the calling-network-address (which is passed in a network-service request primitive) and the local NSAP through which the service primitive is issued from the called-network-address, the calling-transport-address and local information;

Note – The choice of the NSAP at which this primitive is issued is a local matter. This choice has to be consistent with the calling-network-address.

c) uses IPF1 to derive the called-transport-selector (which is sent in transport-PAI) from the called-transport-address; and

d) uses IPF2 to derive the calling-transport-selector (which is sent in transport-PAI) from the calling-transport-address.

Note – The initiator transport-directory-function IAF2 always yields a single network-address. Where the addressing information provided by the Application Title Directory Facility or by private arrangement specifies a list of network-addresses, resolution of these addresses to a single network-address is the responsibilities of local system management. (See § 9.6.2).

11.4.2 In the recipient system, on receipt of a network-service indication primitive, the transport-entity:

a) uses RAF1 to derive the called-transport-address from the called-transport-selector (which is received in a transport-PAI), the called-network-address, and from local information;

Note – Local information may be used to resolve a called-transport-selector that refers to a set of TSAPs to a single TSAP.

b) uses RAF2 to derive the calling-transport-address from the calling-transport-selector (which is received in transport-PAI) and from the calling-network-address;

c) uses RAF3 to derive the responding-network-address from the called-network-address and from local information; and

d) uses RPF1 to derive the responding-transport-selector (which is sent in transport-PAI) from the responding-transport-address (which is received in a transport-service response primitive).

11.4.3 In the initiating system for connection-mode operation, on receipt of a network-service confirmation primitive, the transport-entity uses IAF4 to derive the responding-transport-address from the responding-transport-selector (which is received in transport-PAI) and from the responding-network-address received in the network-service confirmation primitive.

11.5 *Network Layer*

11.5.1 *Introduction*

11.5.1.1 The internal architecture of the Network Layer is complex. At each higher layer of the OSI architecture, an instance of communication (i.e. communication over a connection-mode, or a connectionless-mode data transmission) involves only a single pair of peer entities, which are located in end systems and communicate by means of a peer protocol. Within the Network Layer, however, communication often requires the participation of network-entities not only in the end systems but in intermediate systems as well. The necessary interactions between network-entities may be achieved through the operation of single protocols between pairs of network-entities, or may need more complicated combinations of layered protocols within the Network Layer.

11.5.1.2 The task of the network-directory-functions is, for any instance of communication, to use the called- and calling-network-addresses, together with other information, to determine which network-entities participate in the communication (this determination may be achieved by other methods).

11.5.2 *Properties of network-addresses*

11.5.2.1 *Introduction*

The properties are stated in terms of network-addresses. Since network-addresses refer to sets of NSAP-addresses, they extend naturally to a single NSAP-address.

11.5.2.2 *Global non-ambiguity*

At any instant of time, a network-address identifies just one set of NSAPs in the global domain. A set of NSAPs may have more than one network-address, i.e., synonyms may exist.

11.5.2.3 *Global applicability*

It is possible at any set of NSAPs to identify any other set of NSAPs, in any end system, by means of the network-address of the other set of NSAPs. If the other set of NSAPs has synonymous network-addresses, any of the synonyms will identify the set of NSAPs. In particular, therefore,

a) a network-address identifies the same set of NSAPs whenever it is used:

b) any network-address may be used anywhere to identify the same set of NSAPs;

c) a network-service user, when it receives a calling-network-address in a network-service indication primitive, may use that network-address in another instance of communication with that set of NSAPs.

For a set of NSAPs with synonymous network-addresses, in some circumstances the possibility of communication may depend upon which synonym is used.

Note – The global applicability of network-addresses does not imply that a communication to a given set of NSAPs can always be made. Restrictions can occur because of lack of physical media, lack of directory (routing) information, security procedures, or charging requirements.

11.5.2.4 *Route independence*

Network-service users cannot derive routing information from a network-address. They cannot control the Network Layer's choice of route by means of network-addresses, i.e. by choice of synonym. Similarly they cannot deduce from the network-addresses the route that was used by the network-service provider.

11.5.3 *Network-addresses and SNPAs*

11.5.3.1 Given the requirement to provide communications between two sets of NSAPs, it is a Network Layer function to determine which network-entities have to participate and how they must operate together. In general, this function requires the use of Network Layer Directory Facilities.

11.5.3.2 Network-entities exist within end open systems and intermediate systems. In the real world, end open systems are realized by real end systems; intermediate systems are realized by real subnetworks or by (real) interworking units. An important concept in the relationships between such equipment is that of the Subnetwork Point of Attachment (SNPA) and the associated SNPA address.

11.5.3.3 A SNPA is a point of attachment between a real subnetwork and another piece of equipment, which may be a real end system, an interworking unit or another real subnetwork. Points of attachment to a real subnetwork may be identified within the context of the real subnetwork by an address assigned by the administrative authority of the real subnetwork. This address, both in the real world and in abstract usage, is referred to as the Subnetwork Point of Attachment Address, the SNPA-address or simply subnetwork address.

Note 1 – To give an example, where the real subnetwork is a public data network, an SNPA is called a DTE/DCE interface, and its SNPA-address is called a DTE-address.

Note 2 – In the case of two real subnetworks attached at an SNPA, different SNPA-addresses may be assigned by the authorities of the two real subnetworks to that SNPA.

11.5.3.4 An SNPA is not a service-access-point, and an SNPA-address is not a network-address. Configurations of physical equipment determine the relationships between NSAPs and SNPAs in the Network Layer. Since a real end system may be attached, possibly multiply, to more than one real subnetwork, the relationships between NSAPs and SNPAs may be many-to-many and complex.

11.5.3.5 The determination of the NPAI for different Network Layer protocols is an important Network Layer addressing operation. In many circumstances, multiple protocols are needed to support one instance of communication in the Network Layer. The type of addressing information that each such protocol conveys, and how it is used, are determined by the role of the protocol in the complete protocol structure.

11.5.4 *Network Layer use of directory functions*

11.5.4.1 At the Network Layer, the network-directory-functions directly derive the network-address from the network-PAI.

11.5.4.2 In the initiating system, on receipt of a network-service request primitive, the network-entity:

a) uses IRF1 and IAF2 to derive the called-data-link-address (which is passed in a data-link-service request primitive) from the called-network-address and local information;

b) uses IRF1 and IAF3 to derive the calling-data-link-address (which is passed in a data-link-service request primitive) and the local DLSAP through which the data-link-service primitive is issued from the calling-network-address, the called-network-address, the called-data-link-address, and local information;

c) uses IRF1 and IPF1 to derive the called-network-PAI and the called-subnetwork-address information from the called-network-address and local information;

d) uses IRF1 and IPF2 to derive the calling-network-PAI and the calling-subnetwork-PAI from the called-network-address, the calling-network-address and local information.

11.5.4.3 In the recipient system on receipt of a data-link-service indication primitive, the network-entity:

a) uses RAF1 to derive the called-network-address from the called-network-PAI, and from local information;

Note – Local information may be used to resolve a called-network-address (that refers to a set of NSAPs) to a single NSAP.

b) uses RAF2 to derive the calling-network-address from the calling-network-PAI;

c) uses RAF3 to derive the responding-data-link-address from the called-data-link-address and from local information.

d) uses RPF1 to derive the responding-network-PAI from the responding-network-address which is received in a network-service response primitive.

11.5.4.4 In the initiating system, for connection-mode operation, on receipt of a data-link-service confirmation primitive, the network-entity derives the responding-network-address directly from the responding-network-PAI.

11.5.4.5 Routing is required within the Network Layer when communication between a pair of sets of NSAPs is relayed by a chain of network-entities. Routing functions use the network-address of a called set of NSAPs to select a sequence of relay entities forming a path to the called-network-address.

11.6 *Data Link Layer*

11.6.1 *Introduction*

11.6.1.1 A data-link-address identifies a set of Data-Link-Service-Access-Points (DLSAPs). The network-entities bound to these DLSAPs are thus located by means of that data-link-address. Such a binding has to be known to the initiating open system. It may be recorded in the Network Address Directory Facility.

Note – For some data-link protocols, data-link-addresses are implicit, that is the semantics of the data-link-addresses are not physically carried in the DLPCI. The use of these implicit data-link-addresses is consistent with the characterization of nil selectors in § 9.5.2.

11.6.1.2 A data-link-entity may be bound to more than one DLSAP and more than one Physical-SAP (PhSAP), thus creating a many-to-many correspondence between DLSAPs and PhSAPs.

Note – This correspondence may be constrained to simpler configurations through data-link-protocol specification.

11.6.1.3 Data-link-addresses need only be unique within the scope of the set of open systems attached to a common Data-Link Layer, and within this scope the data-link-address must be equally valid regardless of which physical-address is used.

11.6.2 *Data-Link Layer use of directory-functions*

11.6.2.1 In the initiating system, on receipt of a data-link-service request primitive, the data-link-entity:

a) uses IAF2 to derive a called-physical-address, (which is passed in a physical-service request primitive) from the called-data-link-address, and from local information;

b) uses IAF3 to derive the calling-physical-address, (which is passed in a physical-service request primitive), and the local PhSAP (through which the physical-service primitive is issued), from the called-physical-address, the calling-data-link-address, and local information.

Note – The choice of the PhSAP at which this primitive is issued is a local matter. This choice has to be consistent with the calling-physical-address.

c) uses IPF1 to derive the called-data-link-PAI from the called-data-link-address;

d) uses IPF2 to derive the calling-data-link-PAI from the calling-data-link-address;

11.6.2.2 In the recipient system, on receipt of a physical-service indication primitive, the data-link-entity:

a) uses RAF1 to derive the called-data-link-address from the called-data-link-PAI, the called-physical-address, and from local information;

Note – Local information may be used to resolve the called-data-link-address (that refers to a set of DLSAPs) to a single DLSAP.

b) uses RAF2 to derive the calling-data-link-address from the calling-data-link-PAI and the calling-physical-address; and

c) uses RPF1 to derive the responding-data-link-PAI from the responding-data-link-address which is received in a data-link-service response primitive.

11.6.2.3 In the initiating system, for connection-mode operation, on receipt of a physical-service confirmation primitive, the data-link-entity derives the responding-data-link-address from the responding-data-link-PAI and, possibly, the responding-physical-address.

11.7 *Physical Layer*

11.7.1 A physical-address identifies a set of PhSAPs. The data-link-entities bound to these PhSAPs are thus located by means of that physical-address. Such a binding may be recorded in the Network Address Directory Facility.

11.7.2 Physical-addresses need only be unique within the scope of open systems attached to a common physical medium communication path. Therefore, (N)-directory-functions are not used at the Physical Layer.

Note – Physical-addresses may be implicit.

12 Naming domains and authorities

12.1 A Naming Authority allocates names according to specified rules. A Naming Authority merely dispenses names but does not perform the binding of names to the objects they name.

12.2 Naming-domains may be hierarchically decomposed into naming-subdomains. The naming-domain at the top of the hierarchy is known as a global naming-domain. Each subset (subdomain) of a global naming-domain is put under the control of a naming-authority and does not intersect with other subsets allocated to different naming-authorities.

12.3 The global naming-domain is the set of all possible names, within the OSIE, for objects of a specific type. For example, the set of all application-entity-titles. Independent global naming-domains may exist within the OSIE for objects of different types.

12.4 The global naming-domain may be partitioned (hierarchically decomposed) into naming-subdomains. Thus, any naming-subdomain is also a naming-domain.

12.5 Names taken from different subdomains of the global naming-domain may be bound to the same object. Thus, synonyms may arise.

Note – The requirements for synonyms have been recognized, in particular in the naming of network-service-access-points (synonymous network-addresses), application-entities (synonymous application-entity-titles) and application-processes (synonymous application-process-titles).

12.6 Each naming-domain is administered by a naming-authority. A naming-authority is a registration authority, which only registers names and only has an administrative role. Although naming-authorities register the use of names, they do not participate in the binding of the name to an object. A naming-authority may register names itself or it may partition the naming-domain into naming-subdomains and delegate to a subdomain naming-authority, the responsibility for naming within each naming-subdomain. The procedures for a naming-authority ensure the registration of unambiguous names and, if necessary, provide any rules that subdomain naming-authorities must comply with to meet the registration requirements.

Note – There are a number of ways by which the procedures for a naming-authority may ensure that names registered by a sub-authority are unambiguous. Particular examples are:

a) the allocation of a subset of names from the total set controlled by the naming-authority;

b) the definition of a name component to be added to the name determined by the sub-authority.

12.7 The establishment of naming-authorities requires agreement on the rules for specifying names within the naming-domain and for the creation of further subdomains.

12.8 Within a hierarchy of naming-authorities the operation of each authority is independent of that of the other naming-authorities at the same level, subject only to any common rules established by the registration procedures imposed by the parent authorities.

12.9 A user of a naming-authority may request an allocation of names from a naming-authority, leaving the choice of names to the naming-authority. Alternatively, a user of a naming-authority may request an allocation of particular names. The naming-authority may grant that request if it chooses (provided the names have not previously been issued). The user of a naming-authority may interpret the names issued by a naming-authority in any way it chooses. The use of a name can be terminated and then the name re-used at a later time. The precise rules and constraints on the re-use of a name to ensure that no ambiguity results are specified by the procedures for the naming-authority.

13 Registration procedures for naming within OSI

13.1 The operation of naming within OSI requires the establishment of registration procedures:

a) for the assignment of titles which are unambiguous throughout the OSIE for the following objects:

1) real open systems (system-title),

2) application-processes,

3) application-process-types,

4) applicaton-entity-types; and

b) for the assignment of network-addresses which are unambiguous throughout the OSIE.

13.2 Registered titles or addresses may or may not have synonyms:

a) a single real open system has only one system-title;

b) a single application-process may have more than one application-process-title;

c) a single NSAP may be identified by more than one network-address.

14 Directory facility requirements

14.1 *Introduction*

14.1.1 Two Directory Facilities are required:

a) the Application Title Directory Facility that processes either an application-process-title or an application-entity-title and returns addressing information, as described in § 14.2; and

b) the Network Address Directory Facility that processes a network-address and provides the information used below the network-service boundary to access the remote NSAP, as described in § 14.3.

Note – In the case where a real open system provides both Directory Facilities, the retrieval of information from both Directory Facilities may be made in a single inquiry.

14.1.2 Directory Facilities and the information held by them may be centralized or distributed, and non-replicated, partially replicated or fully replicated. Where communications are needed among Directory Facilities and systems which use these facilities, these communications utilize the normal OSI communication methods available to all application-processes.

14.1.3 Although the user of a primitive name need not be aware of the structure of the name, e.g., the structure resulting from the registration authority, the Directory Facility may physically and logically organize Directory Facility information according to that structure.

14.2 *The Application Title Directory Facility*

14.2.1 The input to the Application Title Directory Facility is an application-process-title or an application-entity-title. This may be a primitive name or a descriptive name. If it is a descriptive name, it is not necessarily complete, i.e., some attributes may be “don't cares”. Possible attributes for a descriptive name are the system-title, the application-process-type-title, and the application-entity-type-title. Descriptive names will be given in a standard descriptive language.

14.2.2 The Application Title Directory Facility supports both generic application-process-titles (e.g. application-process-type-titles) and non-generic application-process-titles. The Application Title Directory Facility supports both generic application-entity-titles (e.g. application-entity-type-titles) and non-generic application-entity-titles.

14.2.3 Using a generic application-process-title as input to the Application Title Directory Facility results in the return of a list of the associated application-process-titles. Any of these may then be subsequently used as input to the Application Title Directory Facility. Using a generic application-entity-title as input to the Application Title Directory Facility results in the return of a list of the associated application-entity-titles. Any of these may then be subsequently used as input to the Application Title Directory Facility.

14.2.4 Using a non-generic application-process-title as input to the Application Title Directory Facility, results in the return of the list of application-entity-titles for the application-entities belonging to the application-process.

14.2.5 Using a non-generic application-entity-title as input to the Application Title Directory Facility results in the associated addressing information in the form of the tuple:

[P-selector, S-selector, T-selector, (list of Network Addresses)].

14.2.6 The Application Title Directory Facility is composed of two different components:

a) a "name resolver" which can resolve an application-process-title/application-entity-title that is a descriptive name into the application-process-title/application-entity-title that is the primitive name of the application-process/application-entity; and

b) a "directory" that returns the information associated with an application-process-title/application-entity-title that is a primitive name.

14.3 *The Network Address Directory Facility*

The input to the Network Address Facility is a network-address which is a primitive name. Using a network-address as input to the Network Address Directory Facility results in the associated addressing information necessary at the Network Layer and the layers below.