INTERNATIONAL TELECOMMUNICATION UNION

CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

X.200

(11/1988)

SERIES X: DATA COMMUNICATION NETWORKS: OPEN SYSTEMS INTERCONNECTION (OSI) – MODEL AND NOTATION, SERVICE DEFINITION

Open Systems Interconnection (OSI) – Model and notation, service definition

---

REFERENCE MODEL OF OPEN SYSTEMS INTERCONNECTION FOR CCITT APPLICATIONS

Reedition of CCITT Recommendation X.200 published in the Blue Book, Fascicle VIII.4 (1988)

NOTES

1 CCITT Recommendation X.200 was published in Fascicle VIII.4 of the *Blue Book*. This file is an extract from the *Blue Book*. While the presentation and layout of the text might be slightly different from the *Blue Book* version, the contents of the file are identical to the *Blue Book* version and copyright conditions remain unchanged (see below).

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

Recommendation X.200

REFERENCE MODEL OF OPEN SYSTEMS INTERCONNECTION FOR CCITT APPLICATIONS[1)]

The CCITT,

considering

(a) that Administrations in many countries are planning to establish telecommunication services which exploit the networks now existing or due to be available in the near future;

(b) that those services may be carried on different types of networks;

(c) that users of these services need to communicate with each other irrespective of the diversity of interconnected networks;

(d) that methodical representation of network services will further promote the effective and efficient use of networks;

(e) that accommodation of conflicting service requirements and network constraints can be effected through the analysis of service and network functions. This analysis should lead to a universally applicable logical structure which may be used in the development of compatible service, interface and procedure definitions;

(f) that this structure should as far as possible accommodate existing Recommendations to allow the steady evolution of networks providing the new services;

(g) that close collaboration and liaison with other bodies studying reference models is desirable to ensure the widest possible applicability of resulting Recommendations,

unanimously recommends

that for CCITT applications:

(1) the methodical representation of new services be undertaken in accordance with the principles and architecture of this Recommendation;

(2) the structure of this reference model contained within this Recommendation be employed in the specification of new interface and procedural definitions;

(3) the user and Administration requirements for both services and management of services can be satisfied by application of the principles of this Recommendation.

---

1) Recommendation X.200 and ISO 7498 [Information Processing Systems – Open Systems Interconnection – Basic Reference Model] were developed in close collaboration and are technically aligned.

CONTENTS

0 Introduction

0.1 *About this Recommendation*

This Recommendation presents the purpose, framework and function of a reference model structure (hereinafter called "the Reference Model") for the logical process in a communications system. Instances of communication system elements that have been defined using this Reference Model may be found in other Recommendations.

Communications systems which employ the standardized communication procedures and methods derived from the Reference Model are referred to as "open systems", and such interconnection is referred to as "Open Systems Interconnection" (OSI). This Reference Model is consistent with the principles established by ISO for Open Systems Interconnection.

This Recommendation allows standardized procedures to be defined enabling the interconnection and subsequent effective exchange of information between users. Such users are systems; i.e., a set of one or more computers, associated software, peripherals, terminals, human operators, physical processes, information transfer means, etc., that form an autonomous whole capable of performing information processing and/or information transfer. The Reference Model, in particular, will permit interworking between different networks, of the same or different types, to be defined such that communication may be achieved as easily over a combination of networks as over a single network.

The term "user" implies nothing in respect to the contractual customer-Administration relationship; a user may be either outside the Administration or part of the Administration.

Conformance with the Reference Model does not imply any particular implementation or technology, but rather refers to the support of standardized information exchange procedures derived according to its provisions as specified in this Recommendation. The Recommendation provides neither details nor definitions or interconnection protocols.

The Reference Model serves as a framework for the definition of services and protocols which fit within the boundaries established by the Reference Model. In those few cases where a feature is explicitly marked as "optional" in the Reference Model, it should remain optional in the corresponding service or protocol (even if at a given instant the two cases of the option are not yet documented).

1 Scope and field of application

1.1 *The scope of the Reference Model*

a) to specify a universally applicable logical structure encompassing the requirements of CCITT applications;

b) to act as a reference during the development of new communications services, including potential CCITT recommended services, and the definition of the corresponding procedures;

c) to enable different users to communicate with each other by encouraging the compatible implementation of communication features;

d) to enable the steady evolution of CCITT applications by allowing sufficient flexibility so that advancements in technology and expanding requirements of users can be accommodated;

e) to allow comparison of a proposed new user requirement with the services for existing user requirements, thus allowing the new requirements to be satisfied in a manner compatible with existing CCITT recommended services.

1.2 *The application of the Reference Model*

This model will be employed in the development of interconnection protocols for communicating services, including potential CCITT recommended services, as follows:

a) A new user requirement is first expressed in user oriented terms. The requirement is then analyzed to determine the underlying patterns which would allow it to be grouped into functional subsets;

b) While the specification of a requirement will contain much narrative text for clarification, there may also be a requirement formally stated using a formal description technique (FDT);

c) A set of service definitions and protocol specifications is being written for each layer. Extensions and new uses of OSI will be progressively incorporated as new CCITT X-series Recommendations;

d) New functions that are identified will be incorporated into the Reference Model to enhance its future applicability;

e) For new uses and applications of OSI where no appropriate protocol is contained in the Recommendations, new protocols, particularly for the application layer, will be needed.

2 Definitions

Definitions of terms are included at the beginning of individual divisions and sub-divisions. An index of these terms is provided in an Annex B for easy reference.

3 Notation

Layers are introduced in division 5. An (N)-, (N + 1)-, and (N – 1)-notation is used to identify and relate adjacent layers:

(N)-Layer: any specific layer;

(N + 1)-Layer: the next higher layer;

(N – 1)-Layer: the next lower layer.

This notation is also used for other concepts in the model which are related to these layers, e.g. (N)-protocol, (N + 1)-service.

Division 6 introduces names for individual layers. When referring to these layers by name, the (N)-, (N + 1)- and (N – 1)-prefixes are replaced by the names of the layers, e.g. transport-protocol, session-entity and network-service.

4 Introduction to Open Systems Interconnection (OSI)

Note – The general principles described in divisions 4 and 5 hold for all layers of the Reference Model, unless layer specific statements to the contrary are made in divisions 6 and 7.

4.1 *Definitions*

4.1.1 **real system**

A set of one or more computers, the associated software, peripherals, terminals, human operators, physical processes, information transfer means, etc., that forms an autonomous whole capable of performing information processing and/or information transfer.

4.1.2 **real open system**

A real system which complies with the requirements of OSI Recommendations in its communication with other real systems.

4.1.3 **open system**

The representation within the Reference Model of those aspects of a real open system that are pertinent to OSI.

4.1.4 **application-process**

An element within a real open system which performs the information processing for a particular application.

4.2 *Open Systems Interconnection environment*

In the concept of OSI, a real system is a set of one or more computers, associated software, peripherals, terminals, human operators, physical processes, information transfer means, etc., that forms an autonomous whole capable of performing information processing and/or information transfer.

An application-process is an element within an open system which performs the information processing for a particular application.

Application-processes can represent manual processes, computerized processes or physical processes. Some examples of application-processes that are applicable to this open system definition are the following:

a) a person operating a banking terminal is a manual application-process;

b) a FORTRAN program executing in a computer centre and accessing a remote database is a computerized application-process; the remote database management systems server is also an application-process; and

c) a process control program executing in a dedicated computer attached to some industrial equipment and linked into a plant control system is a physical application-process.

OSI is concerned with the exchange of information between open systems (and not the internal functioning of each individual real open system).

As shown in Figure 1/X.200, the physical media for open systems interconnection provides the means for the transfer of information between open systems.

Note – At this point, only telecommunications media have been considered. The use of other interconnection media is for further study.

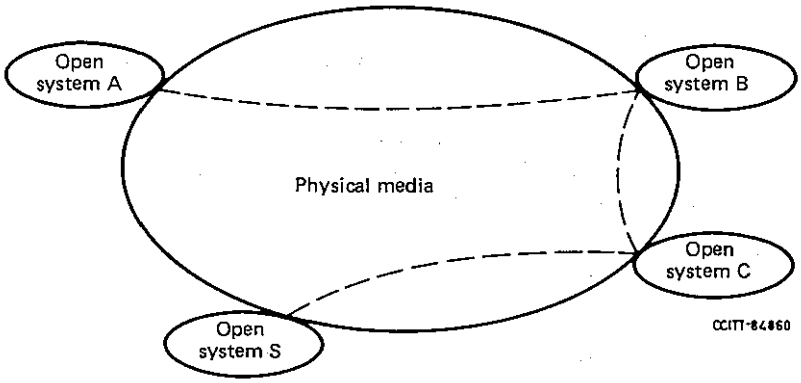


FIGURE 1/X.200

Open systems connected by physical media

OSI is concerned only with interconnection of systems. All other aspects of systems which are not related to interconnection are outside the scope of OSI.

OSI is concerned not only with the transfer of information between systems, i.e. transmission, but also with their capability to interwork to achieve a common (distributed) task. In other words, OSI is concerned with the interconnection aspects of cooperation (see Note) between systems, which is implied by the expression "systems interconnection".

The objective of OSI is to define a set of Recommendations to enable open systems to cooperate. A system which complies with the requirements of applicable OSI Recommendations in its cooperation with other systems is termed a real open system.

Note – Cooperation among open systems involves a broad range of activities of which the following have been identified:

a) interprocess communication, which concerns the exchange of information and the synchronization of activity between OSI application-processes;

b) data representation, which concerns all aspects of the creation and maintenance of data descriptions and data transformations for reformatting data exchanged between open systems;

c) data storage, which concerns storage media, and file and database systems for managing and providing access to data stored on the media;

d) process and resource management, which concerns the means by which OSI application-processes are declared, initiated and controlled, and the means by which they acquire OSI resources;

e) integrity and security, which concerns information processing constraints that must be preserved or assured during the operation of the open systems; and

f) program support, which concerns the definition, compilation, linking, testing, storage, transfer and access to the programs executed by OSI application-processes.

Some of these activities may imply exchange of information between the interconnected open systems and their interconnection aspects may, therefore, be of concern to OSI.

This Recommendation covers the elements of OSI aspects of these activities which are essential for early development of OSI Recommendations.

4.3 *Modelling the OSI environment*

The development of OSI Recommendations, i.e., Recommendations for the interconnection of real open systems, is assisted by the use of abstract models. To specify the external behaviour of interconnected real open systems, each real open system is replaced by a functionally equivalent abstract model of a real open system called an open system. Only the interconnection aspects of these open systems would strictly need to be described. However, to accomplish this, it is necessary to describe both the internal and external behaviour of these open systems. Only the external behaviour of open systems is retained as the standard of behaviour of real open systems. The description of the internal behaviour of open systems is provided in the Reference Model only to support the definition of the interconnection aspects. Any real system which behaves externally as an open system can be considered to be a real open system.

This abstract modelling is used in two steps.

First, basic elements of open systems and some key decisions concerning their organization and functioning, are developed. This constitutes the Reference Model of Open Systems Interconnection described in this Recommendation.

Then, the detailed and precise description of the functioning of the open system is developed in the framework formed by the Reference Model. This constitutes the services and protocols for Open Systems Interconnection which are the subject of other Recommendations.

It should be emphasized that the Reference Model does not, by itself, specify the detailed and precise functioning of the open system and, therefore, it does not *specify* the external behaviour of real open systems and does not imply the structure of the implementation of a real open system.

The reader not familiar with the technique of abstract modelling is cautioned that those concepts introduced in the description of open systems constitute an abstraction despite a similar appearance to concepts commonly found in real systems. Therefore real open systems need not be implemented as described by the Model.

Throughout the remainder of this Recommendation, only the aspects of real systems and application-processes which lie within the OSI environment are considered. Their interconnection is illustrated throughout this Recommendation as depicted in Figure 2/X.200.

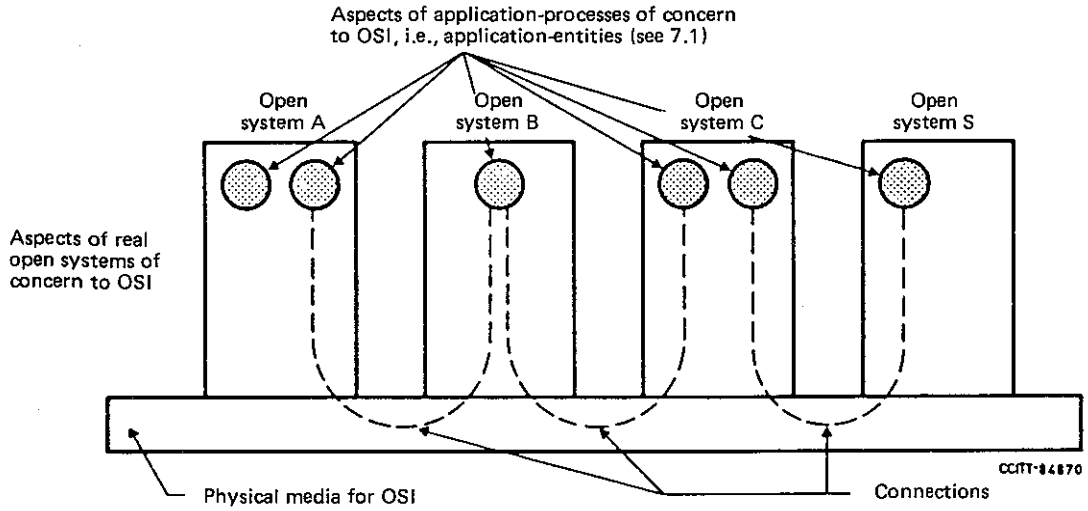


FIGURE 2/X.200

Basic elements of OSI

5 Concepts of a layered architecture

5.1 *Introduction*

Division 5 sets forth the architectural concepts that are applied in the development of the Reference Model of Open Systems Interconnection. Firstly, the concept of a layered architecture (with layers, entities, service-access-points, protocols, connections, etc.) is described. Secondly, identifiers are introduced for entities, service-access-points, and connections. Thirdly, service-access-points and data-units are described. Fourthly, elements of layer operation are described including connections, transmission of data, and error functions. Then, routing aspects are introduced and finally, management aspects are discussed.

The concepts described in division 5 are those required to describe the Reference Model of Open Systems Interconnection. However, not all of the concepts described are employed in each layer of the Reference Model.

Four elements are basic to the Reference Model (see Figure 2/X.200):

a) open systems;

b) the application-entities which exist within the Open Systems Interconnection environment;

c) the connections (see § 5.3) which join the application-entities and permit them to exchange information (see Note 1); and

d) the physical media for Open Systems Interconnection.

Note 1 – This Basic Reference Model of Open Systems Interconnection is based on the assumption that a connection is required for the transfer of data. An addendum to this Recommendation is currently being developed to extend the description to cover the connectionless forms of data transmission which may be found in a wide variety of data communications techniques (e.g. local area networks, digital radio, etc.) and applications (e.g. remote sensing and banking).

Note 2 – Security aspects which are also general architectural elements of protocols are not discussed in this Recommendation.

5.2 *Principles of layering*

5.2.1 *Definitions*

5.2.1.1 **(N)-subsystem**

An element in a hierarchical division of an open system which interacts directly only with elements in the next higher division or the next lower division of that open system.

5.2.1.2 **(N)-layer**

A sub-division of the OSI architecture, constituted by subsystems of the same rank (N).

5.2.1.3 **(N)-entity**

An active element within an (N)-subsystem.

5.2.1.4 **peer-entities**

Entities within the same layer.

5.2.1.5 **sublayer**

A sub-division of a layer.

5.2.1.6 **(N)-service**

A capability of the (N)-layer and the layers beneath it, which is provided to (N + 1)-entities at the boundary between the (N)-layer and the (N + 1)-layer.

5.2.1.7 **(N)-facility**

A part of an (N)-service.

5.2.1.8 **(N)-function**

A part of the activity of (N)-entities.

5.2.1.9 **(N)-service-access-point**

The point at which (N)-services are provided by an (N)-entity to an (N + 1)-entity.

5.2.1.10 **(N)-protocol**

A set of rules and formats (semantic and syntactic) which determines the communication behaviour of (N)-entities in the performance of (N)-functions.

5.2.2 *Description*

The basic structuring technique in the Reference Model of Open Systems Interconnection is layering. According to this technique, each open system is viewed as logically composed of an ordered set of subsystems, represented for convenience in the vertical sequence shown in Figure 3/X.200. Adjacent subsystems communicate through their common boundary. Subsystems of the same rank (N) collectively for the (N)-layer of the Reference Model

of Open Systems Interconnection. An (N)-subsystem consists of one or several (N)-entities. Entities exist in each layer. Entities in the same layer are termed peer-entities. Note that the highest layer does not have an (N + 1)-layer above it and the lowest layer does not have an (N – 1)-layer below it.

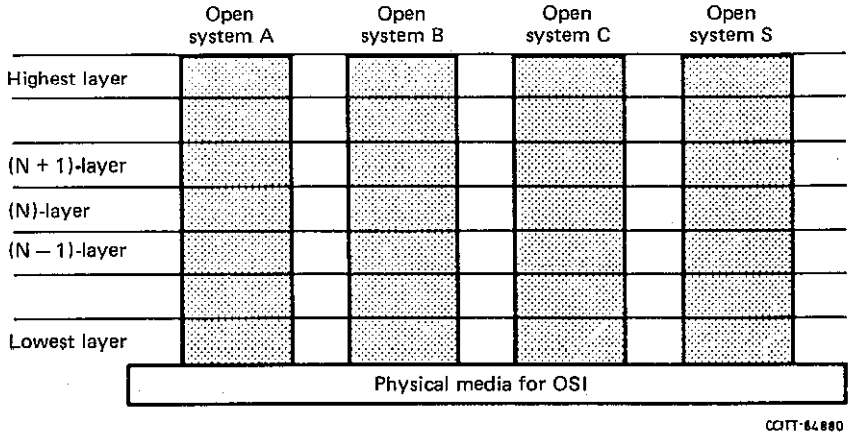


FIGURE 3/X.200

Layering in co-operating open systems

Not all peer (N)-entities need or even can communicate. There may be conditions which prevent this communication (e.g. they are not in interconnected open systems, or they do not support the same protocol subsets).

Note 1 – Type and Instance.

The distinction between the *type* of some object and an *instance* of that object is a distinction of significance for OSI. A type is a description of a class of objects. An instance of this type is any object that conforms to this description. The instances of the same type constitute a class. A type, and any instance of this type can be referred to by an individual name. Each nameable instance and the type to which this instance belongs carry distinguishable names.

For example, given that a programmer has written a computer program, that programmer has generated a type of something, where instances of that type are created every time that particular program is invoked into execution by a computer. Thus, a FORTRAN compiler is a type and each occasion where a copy of that program is invoked in a data processing machine displays an instance of that program.

Consider now an (N)-entity in the OSI context. It, too, has two aspects, a type and a collection of instances. The type of an (N)-entity is defined by the specific set of (N)-layer functions it is able to perform. An instance of that type of (N)-entity is a specific invocation of whatever it is within the relevant open system that provides the (N)-layer functions called for by its type for a particular occasion of communication. It follows from these observations that (N)-entity types refer only to the properties of an association between peer (N)-entities, while an (N)-entity instance refers to the specific, dynamic occasions of actual information exchange.

It is important to note that actual communication occurs only between (N)-entity instances at all layers. It is only at connection establishment time (or its logical equivalent during a recovery process) that (N)-entity types are explicitly relevant. Actual connections are always made to specific (N)-entity instances, although a request for a connection may well be made for arbitrary (N)-entity instances of a specified type. Nothing in Recommendation X.200, however, precludes the request for a connection with a specific (named) instance of a peer (N)-entity. If an (N)-entity instance is aware of the name of its peer (N)-entity instance, it is able to request another connection to that (N)-entity instance.

Note 2 – It may be necessary to further divide a layer into small substructures called sublayers and to extend the technique of layering to cover other dimensions of Open Systems Interconnection. A sublayer is described as a grouping of functions in a layer which may be bypassed. The bypassing of all the sublayers of a layer is not allowed. A sublayer uses the entities and connections of its layer. The detailed definition or additional characteristics of a sublayer are for further study.

Except for the highest layer, each (N)-layer provides (N + 1)-entities in the (N + 1)-layer with (N)-services. The highest layer is assumed to represent all possible uses of the services which are provided by the lower layers.

Note 1 – Not all open systems provide the initial source or final destination of data, such open systems need not contain the higher layers of the architecture (see Figures 6/X.200 and 13/X.200).

Note 2 – Classes of service may be defined within the (N)-services. The precise definition of the term "classes of service" is for further study.

Each service provided by an (N)-layer may be tailored by the selection of one or more (N)-facilities which determine the attributes of that service. When a single (N)-entity cannot by itself fully support a service requested by an (N + 1)-entity it calls upon the cooperation of other (N)-entities to help complete the service request. In order to cooperate, (N)-entities in any layer, other than those in the lowest layer, communicate by means of the set of services provided by the (N – 1)-layer (see Figure 4/X.200). The entities in the lowest layer are assumed to communicate directly via the physical media which connect them.

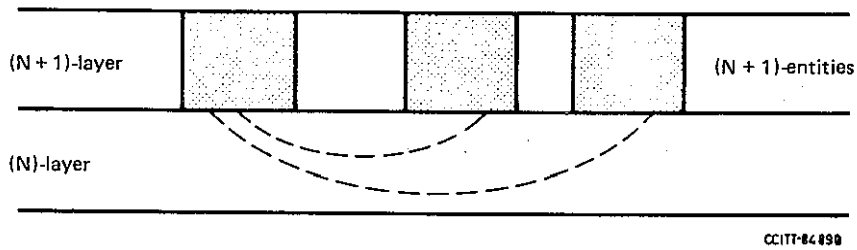


FIGURE 4/X.200

(N+ 1)-entities in the (N + 1)-layer communicate through the (N)-layer

The services of an (N)-layer are provided to the (N + 1)-layer, using the (N)-functions performed within the (N)-layer and as necessary the services available from the (N – 1)-layer.

An (N)-entity may provide services to one or more (N + 1)-entities and use the services of one or more (N – 1)-entities. An (N)-service-access-point is the point at which a pair of entities in adjacent layers use or provide services (see Figure 7/X.200).

Cooperation between (N)-entities is governed by one or more (N)-protocols. The entities and protocols within a layer are illustrated in Figure 5/X.200.

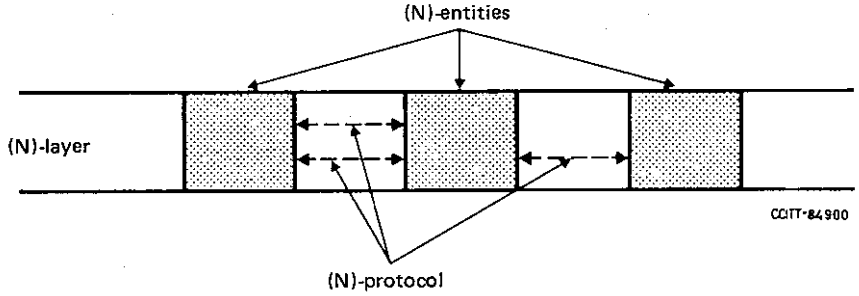


FIGURE 5/X.200

(N-protocols between (N)-entities

5.3 *Communication between peer entities*

5.3.1 *Definitions*

5.3.1.1 **(N)-connection**

An association established by the (N)-layer between two or more (N + 1)-entities for the transfer of data.

5.3.1.2 **(N)-connection-endpoint**

A terminator at one end of an (N)-connection within an (N)-service-access-point.

5.3.1.3 **multi-endpoint-connection**

A connection with more than two connection-endpoints.

5.3.1.4 **correspondent (N)-entities**

(N)-entities with an (N – 1)-connection between them.

5.3.1.5 **(N)-relay**

An (N)-function by means of which an (N)-entity forwards data received from one correspondent (N)-entity to another correspondent (N)-entity.

5.3.1.6 **(N)-data-source**[2)]

An (N)-entity that sends (N – 1)-service-data-units (see § 5.6.1.7) on an (N – 1)-connection.

5.3.1.7 **(N)-data-sink**[3)]

An (N)-entity that receives (N – 1)-service-data-units on an (N – 1)-connection.

5.3.1.8 **(N)-data-transmission**[3)]

An (N)-facility which conveys (N)-service-data-units from one (N + 1)-entity to one or more (N + 1)-entities.

5.3.1.9 **(N)-duplex-transmission**[3)]

(N)-data transmission in both directions at the same time.

5.3.1.10 **(N)-half-duplex-transmission**[3)]

(N)-data transmission in either direction one direction at a time; the choice of direction is controlled by an (N + 1)-entity.

5.3.1.11 **(N)-simplex-transmission**[3)]

(N)-data-transmission in one pre-assigned direction.

5.3.1.12 **(N)-data-communication**[3)]

An (N)-function which transfers (N)-protocol-data-units (see § 5.6.1.3) according to an (N)-protocol over one or more (N – 1)-connections.

5.3.1.13 **(N)-two-way-simultaneous-communication**

(N)-data-communication in both directions at the same time.

5.3.1.14 **(N)-two-way alternate communication**

(N)-data communication in both directions, one direction at a time.

5.3.1.15 **(N)-one-way communication**

(N)-data communication in one pre-assigned direction.

5.3.2 *Description*

For information to be exchanged between two or more (N + 1)-entities, an association is established between them in the (N)-layer using an (N)-protocol.

Note – Classes of protocols may be defined within the (N)-protocols. The precise definition of the term "classes of protocols" is for further study.

This association is called an (N)-connection. (N)-connections are provided by the (N)-layer between two or more (N)-service-access-points. The terminator of an (N)-connection at an (N)-service-access-point is called an (N)-connection-endpoint. A connection with more than two connection-endpoints is termed a multi-endpoint-connection. (N)-entities with a connection between them are termed correspondent (N)-entities.

(N + 1)-entities can communicate only by using the services of the (N)-layer. There are instances where services provided by the (N)-layer do not permit direct access between all of the (N + 1)-entities which have to communicate. If this is the case, communication can still occur if some other (N + 1)-entity can act as a relay between them (see Figure 6/X.200).

---

2) This definitions are not for use in this Recommendation but are for use in future OSI Recommendations.

3) This definitions are not for use in this Recommendation but are for use in future OSI Recommendations.

The fact that communication is relayed by a chain of (N + 1)-entities is known neither by the (N)-layer nor by the (N + 2)-layer.

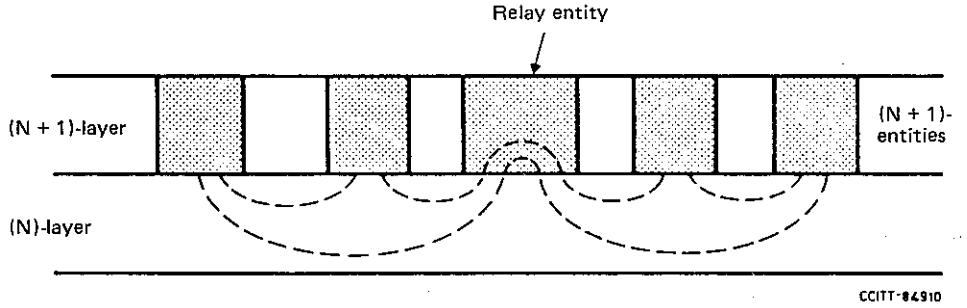


FIGURE 6/X.200

Communication through a relay

5.4 *Identifiers*

5.4.1 *Definitions*

5.4.1.1 **title**

A permanent identifier for an entity.

5.4.1.2 **title-domain**

A subset of the title space of the OSI environment.

5.4.1.3 **title-domain-name**

An identifier which uniquely identifies a title-domain with the OSI environment.

Note – Title-domains of primary importance are the layers. In this specific case, the title-domain-name identifies the (N)-layer.

5.4.1.4 **local-title**

A title which is unique within a title-domain.

5.4.1.5 **global-title**

A title which is unique within the OSI environment and comprises two parts, a title-domain-name and a local-title.

5.4.1.6 **(N)-address; (N)-service-access-point-address**

An identifier which tells where an (N)-service-access-point may be found.

5.4.1.7 **(N)-directory**

An (N)-function by which the global title of an (N)-entity is translated into the (N – 1)-address of an (N – 1)-service-access-point to which the (N)-entity is attached.

5.4.1.8 **(N)-address-mapping**

An (N)-function which provides the mapping between the (N)-addresses and the (N – 1)-addresses associated with an (N)-entity.

5.4.1.9 **routing**

A function within a layer which translates the title of an entity or the service-access-point-address to which the entity is attached into a path by which the entity can be reached.

5.4.1.10 **(N)-connection-endpoint-identifier**

An identifier of an (N)-connection-endpoint which can be used to identify the corresponding (N)-connection at an (N)-service-access-point.

5.4.1.11 **(N)-connection-endpoint-suffix**

A part of an (N)-connection-endpoint-identifier which is unique within the scope of an (N)-service-access-point.

5.4.1.12 **multi-connection-endpoint-identifier**

An identifier which specifies the connection-endpoint of a multi-endpoint-connection which should accept the data that is being transferred.

5.4.1.13 **(N)-service-connection-identifier**

An identifier which uniquely specifies an (N)-connection within the environment of the correspondent (N + 1)-entities.

5.4.1.14 **(N)-protocol-connection-identifier**

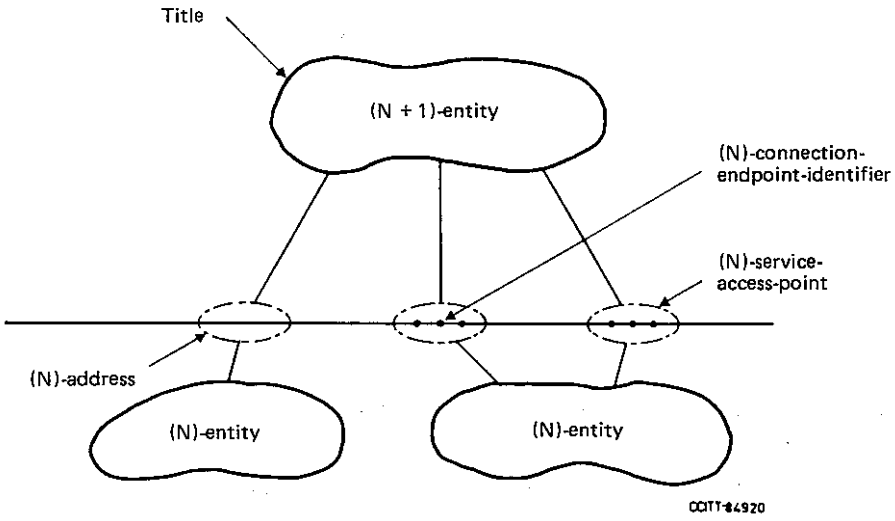
An identifier which uniquely specifies an individual (N)-connection within the environment of the multiplexed (N – 1)-connection.

5.4.1.15 **(N)-suffix**

A part of an (N)-address which is unique within the (N)-service-access-point.

5.4.2 *Description*

An (N)-service-access-point-address, or (N)-address for short, identifies a particular (N)-service-access-point to which an (N + 1)-entity is attached (see Figure 7/X.200). When the (N + 1)-entity is detached from the (N)-service-access-point, the (N)-address no longer provides access to the (N + 1)-entity. If the (N)-service-access-point is reattached to a different (N + 1)-entity, then the (N)-address identifies the new (N + 1)-entity and not the old one.



Note – Dashed arrows refer to identifiers

FIGURE 7/X.200

Entities, service-access-points and identifiers

The use of an (N)-address to identify an (N + 1)-entity is the most efficient mechanism if the permanence of the attachment between the (N + 1)-entity and the (N)-service-access-point can be assured. If there is a requirement to identify an (N + 1)-entity regardless of its current location, then the global-title assures correct identification.

An (N)-directory is an (N)-function which translates global-titles of peer (N)-entities into the (N – 1)-addresses through which they cooperate.

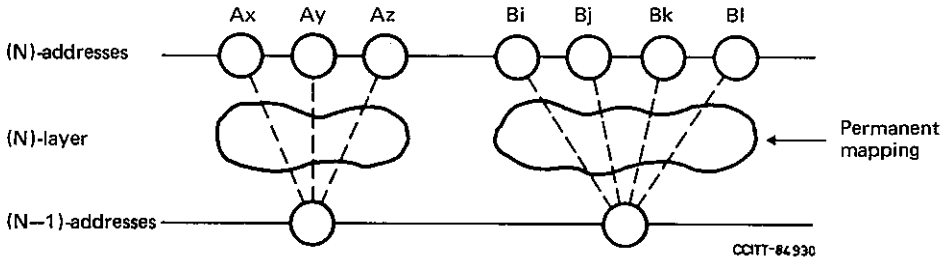
Interpretation of the correspondence between the (N)-addresses served by an (N)-entity and the (N – 1)-addresses used for accessing (N – 1)-services is performed by an (N)-address-mapping function.

Two particular kinds of (N)-address-mapping functions may exist within a layer:

a) hierarchical (N)-address-mapping; and

b) (N)-address-mapping by tables.

If an (N)-address is always mapped into only one (N – 1)-address then hierarchical construction of addresses can be used (see Figure 8/X.200). The (N)-address-mapping function need only recognize the hierarchical structure of an (N)-address and extract the (N – 1)-address it contains.



FIGUR E 8/X.200

Hierarchical (N)-address-mapping

In this case, an (N)-address consists of two parts:

a) an (N – 1)-address of the (N)-entity which is supporting the current (N)-service-access-point of the (N + 1)-entity;

b) an (N)-suffix which makes the (N)-service-access-point uniquely identifiable within the scope of the (N – 1)-address.

Within a given layer, a hierarchical structure of addresses simplifies (N)-address-mapping functions because of the permanent nature of the mapping it presupposes. It is not imposed by the model in all layers in order to allow more flexibility in (N)-address-mappings and to cover the case where an (N)-entity attached to more than one (N – 1)-service-access-point supports only one (N)-service-access-point.

If the previous condition is not true, i.e. either an (N)-address can be mapped into several (N – 1)-addresses, or an (N)-address is not permanently mapped into the same (N – 1)-address, then hierarchical construction of an address is not possible and the (N)-address-mapping function may use tables to translate (N)-addresses into (N – 1)-addresses.

The structure of an (N)-address is known by the (N)-entity which is attached to the identified (N)-service-access-point. However, the (N + 1)-entity does not know this structure.

If an (N + 1)-entity has two or more (N)-service-access-points with either the same (N)-entity or different (N)-entities, the (N)-entities have no knowledge of this fact. Each (N)-service-access-point is considered to identify a different (N + 1)-entity from the perspective of the (N)-entities.

A routing function translates the (N)-address of an (N + 1)-entity into a path or route by which the (N + 1)-entity may be reached.

An (N + 1)-entity may establish an (N)-connection with another (N + 1)-entity by using an (N)-service. When an (N +)-entity establishes an (N)-connection with another (N + 1)-entity, each (N + 1)-entity is given an (N)-connection-endpoint-identifier by its supporting (N)-entity. The (N + 1)-entity can then distinguish the new connection from all other (N)-connections accessible at the (N)-service-access-point it is using. This (N)-connection-endpoint-identifier is required to be unique within the scope of the (N + 1)-entity which will use the (N)-connection.

The (N)-connection-endpoint-identifier consists of two parts:

a) the (N)-address of the (N)-service-access-point which will be used in conjunction with the (N)-connection; and

b) an (N)-connection-endpoint-suffix which is unique within the scope of the (N)-service-access-point.

A multi-endpoint-connection requires multi-connection-endpoint-identifiers. Each such identifier is used to specify which connection-endpoint should accept the data which is being transferred. A multi-connection-endpoint-identifier must be unique within the scope of the connection within which it is used.

The (N)-layer may provide to the (N + 1)-entities an (N)-service-connection-identifier which uniquely specifies the (N)-connection within the environment of the correspondent (N + 1)-entities.

5.5 *Properties of* service-access-points

An (N + 1)-entity requests (N)-services via an (N)-service-access-point which permits the (N + 1)-entity to interact with an (N)-entity.

Both the (N)- and (N + 1)-entities attached to an (N)-service-access-point are in the same system.

An (N + 1)-entity may concurrently be attached to one or more (N)-service-access-points attached to the same or different (N)-entities.

An (N)-entity may concurrently be attached to one or more (N + 1)-entities through (N)-service-access-points.

An (N)-service-access-point is attached to only one (N)-entity and to only one (N + 1)-entity at a time.

An (N)-service-access-point may be detached from an (N + 1)-entity and reattached to the same or another (N + 1)-entity.

An (N)-service-access-point may be detached from an (N)-entity and reattached to the same or another (N)-entity.

An (N)-service-access-point is located by means of its (N)-address. An (N)-address is used by an (N + 1)-entity to request an (N)-connection.

5.6 *Data Units*

5.6.1 *Definitions*

5.6.1.1 **(N)-protocol-control-information**

Information exchanged between (N)-entities, using an (N – 1)-connection, to coordinate their joint operation.

5.6.1.2 **(N)-user-data**

The data transferred between (N)-entities on behalf of the (N + 1)-entities for whom the (N)-entities are providing services.

5.6.1.3 **(N)-protocol-data-unit**

A unit of data specified in an (N)-protocol and consisting of (N)-protocol-control-information and possibly (N)-user-data.

5.6.1.4 **(N)-interface-control-information**

Information transferred between an (N + 1)-entity and an (N)-entity to coordinate their joint operation.

5.6.1.5 **(N)-interface-data**

Information transferred from an (N + 1)-entity to an (N)-entity for transmission to a correspondent (N + 1)-entity over an (N)-connection, or conversely, information transferred from an (N)-entity to an (N + 1)-entity after being received over an (N)-connection from a correspondent (N + 1)-entity.

5.6.1.6 **(N)-interface-data-unit**

The unit of information transferred across the (N)-service-access-point between an (N + 1)-entity and an (N)-entity in a single interaction. Each (N)-interface-data-unit contains (N)-interface-control-information and may also contain the whole or part of an (N)-service-data-unit.

5.6.1.7 **(N)-service-data-unit**

An amount of (N)-interface-data whose identity is preserved from one end of an (N)-connection to the other.

5.6.1.8 **expedited (N)-service-data-unit, (N)-expedited-data-unit**

A small (N)-service-data-unit whose transfer is expedited. The (N)-layer ensures that an expedited-data-unit will not be delivered after any subsequent service-data-unit or expedited unit sent on that connection.

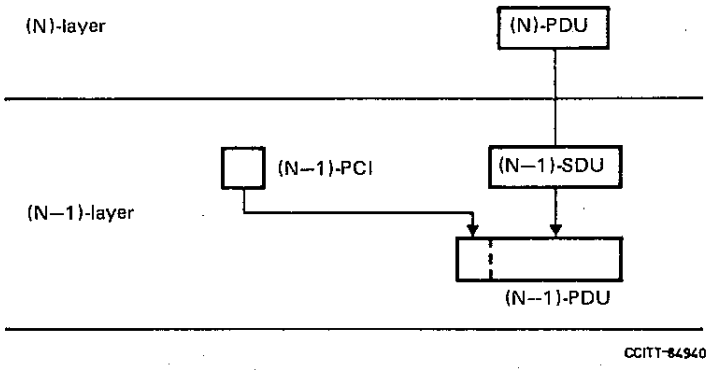
5.6.2 *Description*

Information is transferred in various types of data units between peer-entities and between entities attached to a specific service-access-point. The data-units are defined in sub-division 5.6.1 and the relationships among them are illustrated in Figures 9/X.200 and 10/X.200.

	Control	Data	Combined
(N)-(N) peer-entities	(N)-protocol-control-information	(N)-user-data	(N)-protocol-data-units
(N + 1)-(N) adjacent layers	(N)-interface-control-information	(N)-interface-data	(N)-interface-data-unit

FIGURE 9/X.200

Relationships among data-units



PCI = protocol-control-information
PDU = protocol-data-unit
SDU = service-data-unit

Note 1 – This figure assumes that neither segmenting nor blocking of (N)-service-data-units is performed (see § 5.7.6.5).

Note 2 – This figure does not imply any positional relationship between protocol-control-information and user-data in protocol-data-units.

Note 3 – An (N)-protocol-data-unit may be mapped one-to-one into an (N–1)-service-data-unit, but other relationships are possible (see Figure 11/X.200).

FIGURE 10/X.200

An illustration of mapping between data units in adjacent layers

Except for the relationships defined in Figures 9/X.200 and 10/X.200, there is no overall architectural limit to the size of data units. There may be other size limitations at specific layers.

The size of (N)-interface-data-units is not necessarily the same at each end of the connection.

Data may be held within a connection until a complete service-data-unit is put into the connection.

5.7 *Elements of layer operation*

5.7.1 *Definitions*

5.7.1.1 **(N)-protocol-identifier**

An identifier used between correspondent (N)-entities to select a specific (N)-protocol to be used on a particular (N – 1)-connection.

5.7.1.2 **centralized multi-endpoint-connection**

A multi-endpoint-connection where data sent by the entity associated with the central connection-endpoint is received by all other entities, while data sent by one of the other entities is only received by the central entity.

5.7.1.3 **decentralized multi-endpoint-connection**

A multi-endpoint-connection such that data sent by an entity associated with a connection-endpoint is received by all other entities.

5.7.1.4 **multiplexing**

A function within the (N)-layer by which one (N – 1)-connection is used to support more than one (N)-connection.

Note – The term multiplexing is also used in a more restricted sense to refer to the function performed by the sending (N)-entity while the term demultiplexing is used to refer to the function performed by the receiving (N)-entity.

5.7.1.5 **demultiplexing**

The function performed by an (N)-entity which identifies (N)-protocol-data-units for more than one (N)-connection within (N – 1)-service-data-units received on a single (N – 1)-connection. It is the reverse function of the multiplexing function performed by the (N)-entity sending the (N – 1)-service-data-units.

5.7.1.6 **splitting**

A function within the (N)-layer by which more than one (N – 1)-connection is used to support one (N)-connection.

Note – The term splitting is also used in a more restrictive sense to refer to the function performed by the sending (N)-entity while the term recombining is used to refer to the function performed by the receiving (N)-entity.

5.7.1.7 **recombining**

The function performed by an (N)-entity which identifies (N)-protocol-data-units for a single (N)-connection in (N – 1)-service-data-units received on more than one (N – 1)-connection. It is the reverse function of the splitting function performed by the (N)-entity sending the (N – 1)-service-data-units.

5.7.1.8 **flow control**

A function which controls the flow of data within a layer or between adjacent layers.

5.7.1.9 **segmenting**

A function performed by an (N)-entity to map one (N)-service-data-unit into multiple (N)-protocol-data-units.

5.7.1.10 **reassembling**

A function performed by an (N)-entity to map multiple (N)-protocol-data-units into one (N)-service-data-unit. It is the reverse function of segmenting.

5.7.1.11 **blocking**

A function performed by an (N)-entity to map multiple (N)-service-data-units into one (N)-protocol-data-unit.

5.7.1.12 **deblocking**

A function performed by an (N)-entity to identify multiple (N)-service-data-units which are contained in one (N)-protocol-data-unit. It is the reverse function of blocking.

5.7.1.13 **concatenation**

A function performed by an (N)-entity to map multiple (N)-protocol-data-units into one (N – 1)-service-data-unit.

5.7.1.14 **separation**

A function performed by an (N)-entity to identify multiple (N)-protocol-data-units which are contained in one (N – 1)-service-data-unit. It is the reverse function of concatenation.

5.7.1.15 **sequencing**

A function performed by the (N)-layer to preserve the order of (N)-service-data-units that were submitted to the (N)-layer.

5.7.1.16 **acknowledgement**

A function of the (N)-layer which allows a receiving (N)-entity to inform a sending (N)-entity of the receipt of an (N)-protocol-data-unit.

5.7.1.17 **reset**

A function which sets the correspondent (N)-entities to a predefined state with a possible loss or duplication of data.

Note – Blocking and concatenation, though close to each other (they both permit grouping of data-units) are different (see definitions §§ 5.7.1.11 and 5.7.1.13). These two functions may serve different purposes. For instance, concatenation permits the (N)-layer to group one or several acknowledgement (N)-PDUs with one (or several) (N)-PDUs containing user data, and this would not be possible with the blocking function only. Note also that the two functions may be combined so that the (N)-layer performs blocking and concatenation.

5.7.2 *Protocol selection*

One or more (N)-protocols may be defined for the (N)-layer. An (N)-entity may employ one or more (N)-protocols.

Meaningful communication between (N)-entities over an (N – 1)-connection requires the agreed selection of one (N)-protocol. (N)-protocol-identifiers name the specific protocols defined.

5.7.3 *Properties of connections*

An (N)-connection is an association established for communication between two or more (N + 1)-entities, identified by their (N)-addresses. An (N)-connection is offered as a service by the (N)-layer, so that information may be exchanged between the (N + 1)-entities.

An (N + 1)-entity may have, simultaneously, one or more (N)-connections with other (N + 1)-entities, with any given (N + 1)-entity, and with itself.

An (N)-connection is established by referencing, either explicitly or implicitly, an (N)-address for the source (N + 1)-entity and an (N)-address for each of one or more destination (N + 1)-entities.

The source (N)-address and one or more of the destination (N)-addresses may be the same. One or more of the destination (N)-addresses may be the same while the source (N)-address is different. All may be different.

One (N)-connection-endpoint is constructed for each (N)-address referenced explicitly or implicitly when an (N)-connection is established.

An (N + 1)-entity accesses an (N)-connection via an (N)-service-access-point.

An (N)-connection has two or more (N)-connection-endpoints.

An (N)-connection-endpoint is not shared by (N + 1)-entities or (N)-connections.

An (N)-connection-endpoint relates three elements:

a) an (N + 1)-entity;

b) an (N)-entity; and

c) an (N)-connection.

The (N)-entity and the (N + 1)-entity related by an (N)-connection-endpoint are those implied by the (N)-address referenced when the (N)-connection is established.

An (N)-connection-endpoint has an identifier, called an (N)-connection-endpoint-identifier, which is unique within the scope of the (N + 1)-entity which is bound to the (N)-connection-endpoint.

An (N)-connection-endpoint-identifier is not the same as an (N)-address.

An (N + 1)-entity references an (N)-connection using its (N)-connection-endpoint-identifier.

Multi-endpoint-connections are connections which have three or more connection-endpoints. Two types of multi-endpoint-connection are defined.[4)]

a) centralized; and

b) decentralized.

[4)] Other types of multi-endpoint-connections are for further study.

A centralized multi-endpoint-connection has a central connection-endpoint. Data sent by the entity associated with the central connection-endpoint is received by the entities associated with all other connection-endpoints. The data sent by an entity associated with any other connection-endpoint is received only by the entity associated with the central connection-endpoint.

On a decentralized multi-endpoint-connection, data sent by an entity associated with any connection-endpoint is received by the entities associated with all of the other connection-endpoints.

5.7.4 *Connection establishment and release*

The establishment of an (N)-connection by peer-entities of an (N)-layer requires the following:

a) the availability of an (N – 1)-connection between the supporting (N)-entities; and

b) both (N)-entities be in a state in which they can execute the connection establishment protocol exchange.

If it is not already available, an (N – 1)-connection has to be established by peer-entities of the (N – 1)-layer. This requires, for the (N – 1)-layer, the same conditions as described above for the (N)-layer.

The same consideration applies downwards until either an available connection or the physical medium for Open Systems Interconnection is encountered.

Depending upon the characteristics of the (N – 1)-service and of the establishment protocol exchange, the establishment of an (N)-connection may or may not be done in conjunction with the establishment of the (N – 1)-connection.

The characteristics of the (N)-service with regard to the establishment of the (N)-connection vary depending upon whether or not (N)-user-data can be transferred by the connection establishment protocol exchange for each direction of the (N)-connection.

Where (N)-user-data is transferred by the (N)-connection establishment protocol exchange, the (N + 1)-protocol may take advantage of this to allow an (N + 1)-connection to be established in conjunction with the establishment of the (N)-connection.

The release of an (N)-connection is normally initiated by one of the (N + 1)-entities associated in it.

The release of an (N)-connection may also be initiated by one of the (N)-entities supporting it as a result of an exception condition occurring in the (N)-layer or the layers below.

Depending upon the conditions, release of an (N)-connection may result in the discarding of (N)-user-data.

The orderly release of an (N)-connection requires either the availability of an (N – 1)-connection, or a common reference to time (e.g. time of failure of the (N – 1)-connection and common time-out). In addition, it requires that both (N)-entities are in a state in which they can execute the connection release protocol exchange. It is important to note, however, that the release of an (N – 1)-connection does not necessarily cause the release of the (N)-connection(s) which were using it; the (N – 1)-connection can be re-established, or another (N – 1)-connection substituted.

The characteristics of the (N)-service with regard to the release of an (N)-connection can be of two kinds:

a) (N)-connections are either released immediately when the release protocol exchange is initiated ((N)-user-data not yet delivered may be discarded); or

b) release is delayed until all (N)-user-data sent previous to the initiation of the release protocol exchange has been delivered (i.e., delivery confirmation has been received.)

(N)-user-data may be transferred by the connection release protocol exchange.

Some (N)-protocols may provide for the combining of connection establishment and connection release protocol exchanges.

5.7.5 *Multiplexing and splitting*

Within the (N)-layer, (N)-connections are mapped onto (N – 1)-connections. The mapping may be one of three kinds:

a) one-to-one;

b) many (N)-connections to one (N – 1)-connection (multiplexing); and

c) one (N)-connection to many (N – 1)-connections (splitting).

Multiplexing may be needed in order to:

a) make more efficient or more economic use of the (N – 1)-service; and

b) provide several (N)-connections in an environment where only a single (N – 1)-connection exists.

Splitting may be needed in order to:

a) improve reliability where more than one (N – 1)-connection is available;

b) provide the required grade of performance, through the utilization of multiple (N – 1)-connections; and

c) obtain cost benefits by the utilization of multiple low cost (N – 1)-connections each with less than the required grade of performance.

Multiplexing and splitting each involve a number of associated functions which may not be needed for one-to-one connection mapping.

The functions associated with multiplexing are:

a) identification of the (N)-connection for each (N)-protocol-data-unit transferred over the (N – 1)-connection, in order to ensure that (N)-user-data from the various multiplexed (N)-connections are not mixed. This identification is distinct from that of the (N)-connection-endpoint-identifiers and is called an (N)-protocol-connection-identifier;

b) flow control on each (N)-connection in order to share the capacity of the (N – 1)-connection (see § 5.7.6.4); and

c) scheduling the next (N)-connection to be serviced over the (N – 1)-connection when more than one (N)-connection is prepared to send data.

The functions associated with splitting are:

a) scheduling the utilization of multiple (N – 1)-connections used in splitting a single (N)-connection; and

b) resequencing of (N)-protocol-data-units associated with an (N)-connection since they may arrive out of sequence even when each (N – 1)-connection guarantees sequence of delivery (see § 5.7.6.6).

5.7.6 *Transfer of data*

5.7.6.1 *Normal Data transfer*

Control information and user data are transferred between (N)-entities in (N)-protocol-data-units. An (N)-protocol-data-unit is a unit of data specified in an (N)-protocol and contains (N)-protocol-control-information and possibly (N)-user-data.

(N)-protocol-control-information is transferred between (N)-entities using the (N – 1)-connection. (N)-protocol-control-information is any information that supports the joint operation of (N)-entities. (N)-user-data is passed transparently between (N)-entities over an (N – 1)-connection.

An (N)-protocol-data-unit has an arbitrary, but finite, size. (N)-protocol-data-units are mapped into (N – 1)-service-data-units. The interpretation of an (N)-protocol-data-unit is defined by the (N)-protocol in effect for the (N – 1)-connection.

An (N)-service-data-unit is transferred between an (N + 1)-entity and an (N)-entity, through an (N)-service-access-point, in the form of one or more (N)-interface-data-units. The (N)-service-data-unit is transferred as (N)-user-data in one or more (N)-protocol-data-units.

The exchange of data under the rules of an (N)-protocol can only occur if an (N – 1)-connection exists. If an (N – 1)-connection does not exist, it must be established before an exchange of data can occur (see § 5.7.4).

5.7.6.2 *Data transfer during connection establishment and release*

(N)-user-data may be transferred in the (N)-connection establishment protocol exchange and in the (N)-connection release protocol exchange.

The connection release protocol exchange may be combined with the connection establishment protocol exchange (see § 5.7.4) to provide means for the delivery of a single unit of (N)-user-data between correspondent (N + 1)-entities with a confirmation of receipt.

5.7.6.3 *Expedited transfer of data*

An expedited-data-unit is a service-data-unit which is transferred and/or processed with priority over normal service-data-units. An expedited data transfer service may be used for signalling and interrupt purposes.

Expedited data flow is independent of the states and operation of the normal data flow, although the data sent on the two flows may be logically related. Conceptually, a connection that supports expedited flow can be viewed as having two subchannels, one for normal data, the other for expedited data. Data sent on the expedited channel is assumed to be given priority over normal data.

The transfer guarantees that an expedited-data-unit will not be delivered after any subsequent normal service-data-unit or expedited-data-unit sent on the connection.

Because the expedited flow is assumed to be used to transfer small amounts of data infrequently, simplified flow control mechanisms may be used on this data flow.

An expedited (N)-service-data-unit is intended to be processed by the receiving (N + 1)-entity with priority over normal (N)-service-data-units.

5.7.6.4 *Flow control*

If flow control functions are provided, they can operate only on protocol-data-units and interface-data-units.

Two types of flow control are identified:

a) peer flow control which regulates the rate at which (N)-protocol-data-units are sent to the peer (N)-entity on the (N)-connection. Peer flow control requires protocol definitions and is based on protocol-data-unit size; and

b) (N)-interface flow control which regulates the rate at which (N)-interface-data are passed between an (N + 1)-entity and an (N)-entity. (N)-interface flow control is based on the (N)-interface-data-unit size.

Multiplexing in a layer may require a peer flow control function for individual flows (see § 5.7.5).

Peer flow control functions require that flow control information be included in the (N)-protocol-control-information of an (N)-protocol-data-unit.

If the size of service-data-units exceeds the maximum size of the (N)-user-data portion of an (N)-protocol-data-unit, then first segmentation must be performed on the (N)-service-data-unit to make it fit within the (N)-protocol-data-units. Peer flow control can then be applied on the (N)-protocol-data-units.

5.7.6.5 *Segmenting, blocking and concatenation*

Data units in the various layers are not necessarily of compatible size. It may be necessary to perform segmenting, i.e., to map an (N)-service-data-unit into more than one (N)-protocol-data-unit. Similarly, segmenting may occur when (N)-protocol-data-units are mapped into (N – 1)-interface-data-units. Since it is necessary to preserve the identity of (N)-service-data-units on an (N)-connection, functions shall be available to identify the segments of an (N)-service-data-unit, and to allow the correspondent (N)-entities to reassemble the (N)-service-data-unit.

Segmenting may require that information be included in the (N)-protocol-control-information of an (N)-protocol-data-unit. Within a layer, (N)-protocol-control-information is added to an (N)-service-data-unit to form an (N)-protocol-data-unit when no segmenting or blocking is performed (see Figure 11a/X.200). If segmenting is performed, an (N)-service-data-unit is mapped into several (N)-protocol-data-units with added (N)-protocol-control-information (see Figure 11b/X.200).

Conversely, it may be necessary to perform blocking whereby several (N)-service-data-units with added (N)-protocol-control-information form an (N)-protocol-data-unit (see Figure 11c/X.200).

The Reference Model also permits concatenation whereby several (N)-protocol-data-units are concatenated into a single (N – 1)-service-data-unit (see Figure 11d/X.200).

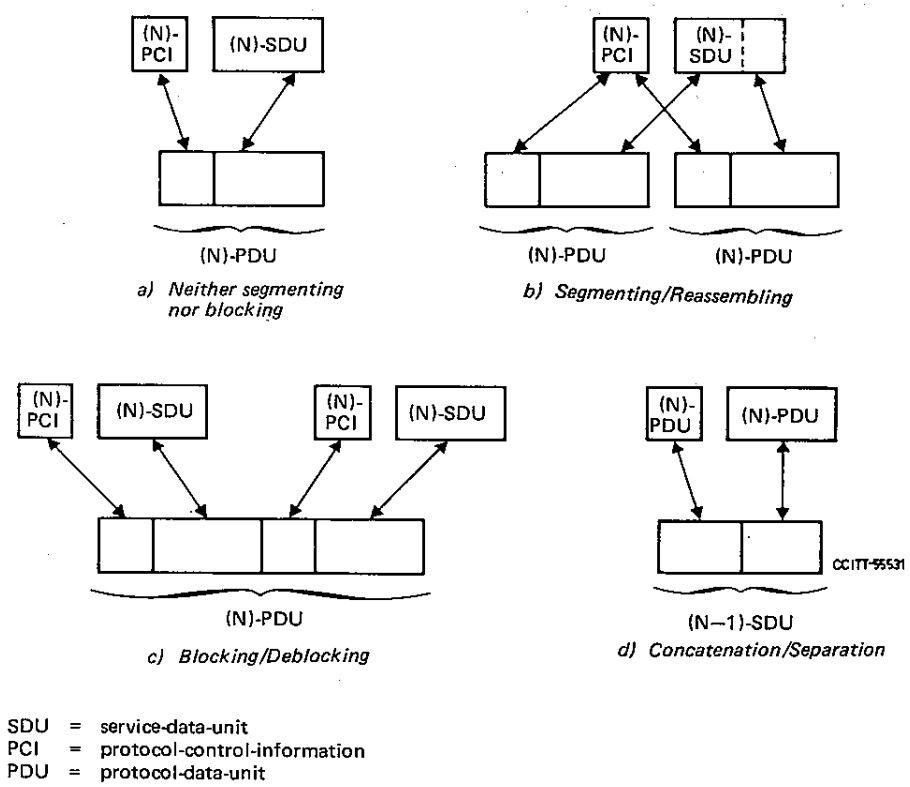
5.7.6.6 *Sequencing*

The (N – 1)-services provided by the (N – 1)-layer of the OSI architecture may not guarantee delivery of (N – 1)-service-data-units in the same order as it was submitted by the (N)-layer. If the (N)-layer needs to preserve the order of (N – 1)-service-data-units transferred through the (N – 1)-layer, sequencing mechanisms must be present in the (N)-layer. Sequencing may require additional (N)-protocol-control-information.

5.7.7 *Error functions*

5.7.7.1 *Acknowledgement*

An acknowledgement function may be used by peer (N)-entities using an (N)-protocol to obtain a higher probability of detecting protocol-data-unit loss than is provided by the (N – 1)-layer. Each (N)-protocol-data-unit transferred between correspondent (N)-entities is made uniquely identifiable, so that the receiver can inform the sender of the receipt of the (N)-protocol-data-unit. An acknowledgement function is also able to infer the non-receipt of (N)-protocol-data-units and take appropriate remedial action.



Note 1 – This figure does not imply any positional relationship between protocol-control-information and user-data in protocol-data-units.

Note 2 – In the case of concatenation, (N)-protocol-data-unit does not necessarily include an (N)-service-data-unit.

FIGURE 11/X.200

Relationship between (N)-service-data-unit, (N)-protocol-data-unit and (N – 1)-service-data-unit within a layer

An acknowledgement function may require that information be included in the (N)-protocol-controlinformation of (N)-protocol-data-units.

The scheme for uniquely identifying (N)-protocol-data-units may also be used to support other functions such as detection of duplicate data-units, segmenting and sequencing.

Note – Other forms of acknowledgement such as confirmation of delivery and confirmation of performance of an action are for further study.

5.7.7.2 *Error detection and notification*

Error detection and notification functions may be used by an (N)-protocol to provide a higher probability of both protocol-data-unit error detection and data corruption detection than is provided by the (N – 1)-service.

Error detection and notification may require that additional information be included in the (N)-protocol-control-information of the (N)-protocol-data-unit.

5.7.7.3 *Reset*

Some services require a reset function to recover from a loss of synchronization between correspondent (N)-entities. A reset function sets the correspondent (N)-entities to a predefined state with a possible loss or duplication of data.

Note – Additional functions may be required to determine at what point reliable data transfer was interrupted.

A quantity of (N)-user-data may be conveyed in association with the (N)-reset function.

The reset function may require that information be included in the (N)-protocol-control-information of the (N)-protocol-data-unit.

5.8 *Routing*

A routing function within the (N)-layer enables communication to be relayed by a chain of (N)-entities. The fact that communication is being routed by intermediate (N)-entities is known by neither the lower layers nor by the higher layers. An (N)-entity which participates in a routing function may have a routing table.

5.9 *Management aspects of OSI*

5.9.1 *Definitions*

5.9.1.1 **application-management**

Functions in the Application Layer (see § 6.1) related to the management of OSI application-processes.

5.9.1.2 **application-management-application-entity**

An application-entity which executes application-management functions.

5.9.1.3 **OSI resources**

Data processing and data communication resources which are of concern to OSI.

5.9.1.4 **systems-management**

Functions in the Application Layer related to the management of various OSI resources and their status across all layers of the OSI architecture.

5.9.1.5 **systems-management-application-entity**

An application-entity which executes systems-management functions.

5.9.1.6 **layer-management**

Functions related to the management of the (N)-layer partly performed in the (N)-layer itself according to the (N)-protocol of the layer (activities such as activation and error control) and partly performed as a subset of systems-management.

5.9.2 *Introduction*

Within the OSI architecture there is a need to recognize the special problems of initiating, terminating, and monitoring activities and assisting in their harmonious operations, as well as handling abnormal conditions. These have been collectively considered as the management aspects of the OSI architecture. These concepts are essential to the operation of the interconnected open systems and therefore are included in the comprehensive description of the Reference Model described in subsequent divisions of this Recommendation.

The management activities which are of concern are those which imply actual exchanges of information between open systems. Only the protocols needed to conduct such exchanges are candidates for standardization within the OSI architecture.

This sub-division describes key concepts relevant to the management aspects, including the different categories of management activities and the positioning of such activities within the OSI architecture.

5.9.3 *Categories of management activities*

Only those management activities which imply actual exchanges of information between remote management entities are pertinent to the OSI architecture. Other management activities local to particular open systems are outside its scope.

Similarly, not all resources are pertinent to OSI. This Recommendation considers only OSI resources, i.e., those data processing and data communication resources which are of concern to OSI.

The following categories of management activities are identified:

a) application-management;

b) systems-management; and

c) layer-management.

5.9.3.1 *Application-management*

Application-management relates to the management of OSI application-processes. The following list is typical of activities which fall into this category but it is not exhaustive:

a) initialization of parameters representing application-processes;

b) initiation, maintenance and termination of application-processes;

c) allocation and de-allocation of OSI resources to application-processes;

d) detection and prevention of OSI resource interference and deadlock;

e) integrity and commitment control;

f) security control; and

g) checkpointing and recovery control.

The protocols for application-management reside within the Application Layer, and are handled by application-management-application-entities.

5.9.3.2 *Systems-management*

Systems-management relates to the management of OSI resources and their status across all layers of the OSI architecture. The following list is typical of activities which fall into this category but it is not exhaustive:

a) activation/deactivation management which includes:

   1) activation, maintenance and termination of OSI resources distributed in open systems, including physical media for OSI;

   2) some program loading functions;

   3) establishment/maintenance/release of connections between management entities; and

   4) open systems parameter initialization/modification;

b) monitoring which includes:

   1) reporting status or status changes; and

   2) reporting statistics; and

c) error control which includes:

   1) error detection and some of the diagnostic functions; and

   2) reconfiguration and restart.

The protocols for systems-management reside in the Application Layer, and are handled by systems-management-application-entities.

5.9.3.3 *Layer-management*

There are two aspects of layer-management. One of these is concerned with layer activities such as activation and error control. This aspect is implemented by the layer protocol to which it applies.

The other aspect of layer-management is a subset of systems-management. The protocols for these activities reside within the Application Layer and are handled by system-management-application-entities.

5.9.4 *Principles for positioning management functions*

Several principles are important in positioning management functions in the Reference Model of Open Systems Interconnection. They include the following:

a) both centralization and decentralization of management functions are allowed. Thus, the OSI architecture does not dictate any particular fashion or degree of centralization of such functions. This principle calls for a structure in which each open system is allowed to include any (subset of) systems-management functions and each subsystem is allowed to include any (subset of) layer-management functions; and

b) if it is necessary, connections between management entities are established when an open system which has been operating in isolation from other open systems, becomes part of the OSI environment.

Note – Other principles are for further study.

6 Introduction

6.1 *Specific layers*

The general structure of the OSI architecture described in division 5 provides architectural concepts from which the Reference Model of Open Systems Interconnection has been derived, making specific choices for the layers and their contents.

The Reference Model contains seven layers:

a) the Application Layer (layer 7);

b) the Presentation Layer (layer 6);

c) the Session Layer (layer 5);

d) the Transport Layer (layer 4);

e) the Network Layer (layer 3);

f) the Data Link Layer (layer 2); and

g) the Physical Layer (layer 1).

These layers are illustrated in Figure 12/X.200. The highest layer is the Application Layer and it consists of the application-entities that cooperate in the OSI environment. The lower layers provide the services through which the application-entities cooperate.

Layers 1-6, together with the physical media for OSI provide a step-by-step enhancement of communication services. The boundary between two layers identifies a stage in this enhancement of services at which an OSI service Recommendation is defined, while the functioning of the layers is governed by OSI protocol Recommendations.

Not all open systems provide the initial source or final destination of data. When the physical media for OSI do not link all open systems directly, some open systems act only as relay open systems, passing data to other open systems. The functions and protocols which support the forwarding of data are then provided in the lower layers. This is illustrated in Figure 13/X.200.

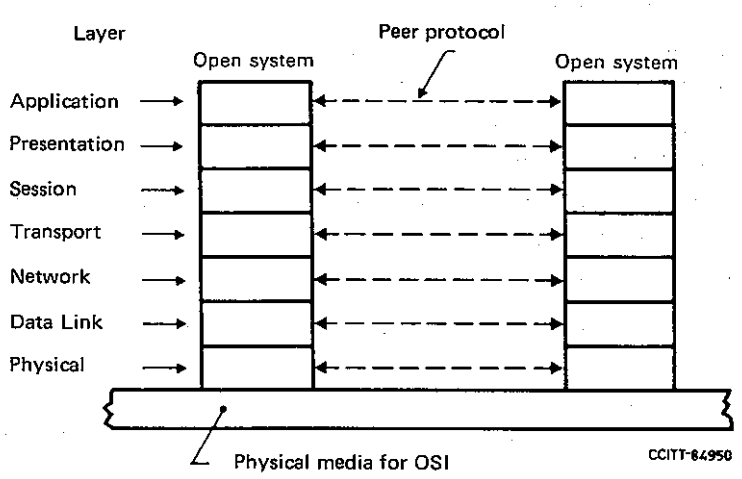


FIGURE 12/X.200

Seven layer reference model and peer protocols

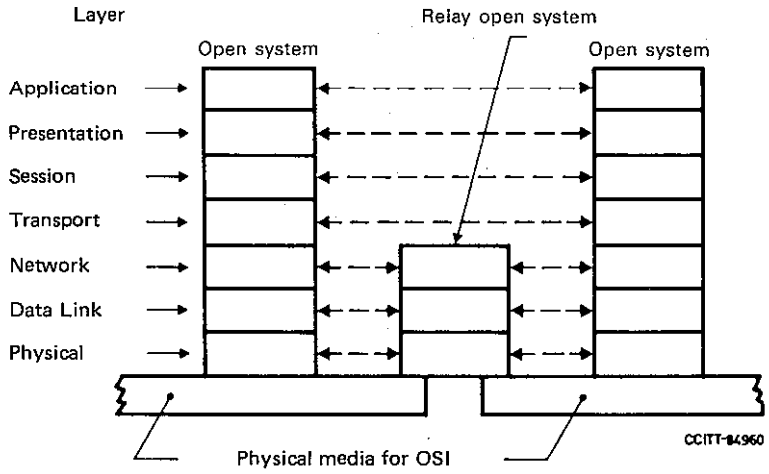


FIGURE 13/X.200

Communication involving relay open systems

6.2 *The principles used to determine the seven layers in the Reference Model*

The following principles have been used to determine the seven layers in the Reference Model and are felt to be useful for guiding further decisions in the development of OSI Recommendations.

Note – It may be difficult to prove that any particular layering selected is the best possible solution. However, there are general principles which can be applied to the question of where a boundary should be placed and how many boundaries should be placed.

P1: do not create so many layers as to make the system engineering task of describing and integrating the layers more difficult than necessary;

P2: create a boundary at a point where the description of services can be small and the number of interactions across the boundary are minimized;

P3: create separate layers to handle functions that are manifestly different in the process performed or the involved technology;

P4: collect similar functions into the same layer;

P5: select boundaries at a point which past experience has demonstrated to be successful;

P6: create a layer of easily localized functions so that the layer could be totally redesigned and its protocols changed in a major way to take advantage of new advances in architectural, hardware or software technology without changing the services expected from and provided to the adjacent layers;

P7: create a boundary where it may be useful at some point in time to have the corresponding interface standardized;

Note 1 – Advantages and drawbacks of standardizing internal interfaces within open systems are not considered in this Recommendation. In particular, mention of, or reference to principle P7, should not be taken to imply usefulness of standards for such internal interfaces.

Note 2 – It is important to note that OSI per se does not require interfaces within open systems to be standardized. Moreover, whenever standards for such interfaces are defined, adherence to such internal interface standards can in no way be considered as a condition of openness.

P8: create a layer where there is a need for a different level of abstraction in the handling of data, e.g., morphology, syntax, semantics;

P9: allow changes of functions or protocols to be made within a layer without affecting other layers; and

P10: create for each layer, boundaries with its upper and lower layer only.

Similar principles have been applied to sublayering:

P11: create further subgrouping and organization of functions to form sublayers within a layer in cases where distinct communications services need it;

P12: create, where needed, two or more sublayers with a common, and therefore minimal functionality to allow interface operation with adjacent layers; and

P13: allow by-passing of sublayers.

6.3 *Layer descriptions*

For each of the seven layers identified above, division 7 provides:

a) an outline of the purpose of the layer;

b) a description of the services offered by the layer to the layer above; and

c) a description of the functions provided in the layer and the use made of the services provided by the layer below.

The descriptions, by themselves, do not provide a complete definition of the services and protocols for each layer. These are the subject of separate Recommendations.

7 Detailed description of the resulting OSI architecture

7.1 *Application layer*

7.1.1 *Definitions*

7.1.1.1 **application-entity**

The aspects of an application-process pertinent to OSI.

7.1.1.2 **application-service-element**

A part of an application-entity which provides an OSI environment capability, using underlying services where appropriate.

7.1.1.3 **user-element**

The representation of that part of the application-process which uses those application-service-elements needed to accomplish the communications objectives of that application-process.

7.1.2 *Purpose*

As the highest layer in the Reference Model of Open Systems Interconnection, the application layer provides a means for the application-processes to access the OSI environment. Hence the application layer does not interface with a higher layer. The application layer is the sole means for the application-processes to access the OSI environment.

The purpose of the application layer is to serve as the window between correspondent application-processes which are using the OSI to exchange meaningful information.

Each application-process is represented to its peer by the application-entity.

All specifiable application-process parameters of each OSI environment communications instance are made known to the OSI environment (and, thus, to the mechanisms implementing the OSI environment) via the application layer.

7.1.3 *Services provided to applications-processes*

Application-processes exchange information by means of application-entities, application-protocols, and presentation services.

As the only layer in the Reference Model that directly provides services to the application-processes, the application layer necessarily provides all *OSI* services directly usable by application-processes.

The application-entity contains one user-element and a set of application-service-elements. The user-element represents that part of the application-process which uses those application-service-elements needed to accomplish the communications objectives of that application-process. Application-service-elements may call upon each other and/or upon presentation services to perform their function.

The only means by which user-elements in different systems may communicate is through the exchange of application-protocol-data-units. These application-protocol-data-units are generated by application-service- elements.

Note – The application services differ from services provided by other layers in neither being provided to an upper layer nor being associated with a service-access-point.

In addition to information transfer, such services may include, but are not limited to the following:

Note – Some of the services listed below are provided by OSI management.

a) identification of intended communications partners (e.g., by name, by address, by definite description, by generic description);

b) determination of the current availability of the intended communication partners;

c) establishment of the authority to communicate;

d) agreement on privacy mechanisms;

e) authentication of intended communication partners;

f) determination of cost allocation methodology;

g) determination of the adequacy of resources;

h) determination of the acceptable quality of service (e.g., response time, tolerable error rate, cost vis-a-vis the previous considerations);

i) synchronization of cooperating applications;

j) selection of the dialogue discipline including the initiation and release procedures;

k) agreement on the responsibility for error recovery;

l) agreement on procedures for control of data integrity; and

m) identification of constraints on data syntax (character sets, data structure).

7.1.4 *Functions within the application layer*

The application layer contains all functions which imply communication between open-systems and are not already performed by the lower layers. These include functions performed by programs as well as functions performed by human beings.

When a specific instance of an application-process wishes to communicate with an instance of an application-process in some other open-system, it must invoke an instance of an application-entity in the application layer of its own open-system. It then becomes the responsibility of this instance of the application entity to establish an association with an instance of an appropriate application-entity in the destination open-system. This process occurs by invocation of instances of entities in the lower layers. When the association between the two application-entities has been established, the application-processes can communicate.

7.1.4.1 *Grouping of functions in the application layer*

An application-entity can be structured internally into groups of functions. The technique used to express this structure is not constrained by this Recommendation. Use of one grouping of functions may depend on use of some other functions, and the active functions may vary during the lifetime of an association.

The structuring of application-entities into application-service-elements and the user-element provides an organization of functions in application-entities. Furthermore, any given subset of application-service-elements, along with the user-element, constitutes an application-entity type. Each application-entity type, and each instance thereof, are unambiguously identifiable.

An application-process may determine the grouping of functions comprising the application-entity.

Two categories of application-service-elements are recognized: common-application-service-elements and specific-application-service-elements. Common-application-service-elements provide capabilities that are generally useful to a variety of applications. Specific-application-service-elements provide capabilities required to satisfy the particular needs of specific applications (for example, file-transfer, data base access, job transfer, banking, order-entry). Application entities may contain application-service-elements from both categories, as illustrated in Figure 14/X.200.

The partitioning of application-service-elements into these two categories does not imply the existence of two independent protocols.

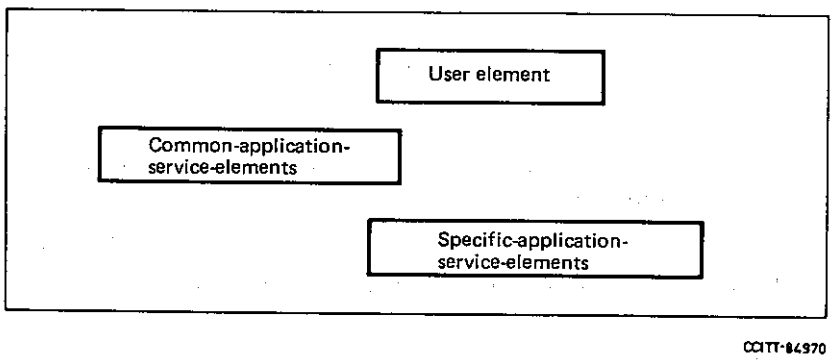


FIGURE 14/X.200

Application-entity

7.1.4.2 *Systems-management and application-management*

Systems-management functions and application-management functions are located in the application layer. For details see sub-division 5.9.

7.1.4.3 *Application layer management*

In addition to systems- and application-management, there are other activities specifically related to application layer management (such as activation and error control). See sub-division 5.9 for the relationship with other management aspects.

7.2 *Presentation layer*

7.2.1 *Definitions*

7.2.1.1 **concrete syntax**

Those aspects of the rules used in the formal specification of data which embody a specific representation of that data.

7.2.1.2 **transfer syntax**

That concrete syntax used in the transfer of data between open systems.

7.2.2 *Purpose*

The presentation layer provides for the representation of information that application-entities either communicate or refer to in their communication.

The presentation layer covers two complementary aspects of this representation of information:

a) the representation of data to be transferred between application-entities; and

b) the representation of the data structure which application-entities refer to in their communication, along with the representations of the set of actions which may be performed on this data structure.

The complementary aspects of the representation of information outlined above refer to the general concept of transfer syntax.

The presentation layer is concerned only with the syntax, (i.e., the representation of the data) and not with its semantics, (i.e., their meaning to the application layer), which is known only by the application-entities.

The presentation layer provides for a common representation to be used between application-entities. This relieves application-entities of any concern with the problem of "common" representation of information, i.e., it provides them with syntax independence. This syntax independence can be described in two ways:

a) the presentation layer provides common syntactical elements which are used by application-entities; and

b) the application-entities can use any syntax and the presentation layer provides the transformation between these syntaxes and the common syntax needed for communication between applicationentities. This transformation is performed inside the open systems. It is not seen by other open systems and therefore has no impact on the standardization of presentation-protocols.

In this Recommendation the approach outlined in b) is used.

7.2.3 *Services provided to the application layer*

The presentation layer provides session-services (see § 7.3) and the following facilities:

a) transformation of syntax; and

b) selection of syntax.

Tranformation of syntax is concerned with code and character set conversions, with the modification of the layout of the data and the adaptation of actions on the data structures. Selection of syntax provides the means of initially selecting a syntax and subsequently modifying the selection.

Session-services are provided to application entities in the form of presentation-services.

7.2.4 *Functions within the presentation layer*

The presentation layer peforms the following functions to help accomplish the presentation-services:

a) session establishment request;

b) data transfer;

c) negotiation and renegotiation of syntax;

d) transformation of syntax including data transformation, formatting and special purpose transformations (e.g., compression); and

e) session termination request.

7.2.4.1 *Transformation of syntax*

The fact that there is or is not actual transformation of syntax has no impact on the presentation protocol.

There are three syntactic versions of the data: the syntax used by the originating application-entity, the syntax used by the receiving application-entity and the syntax used between presentation-entities (the transfer syntax). It is clearly possible that any two or all three of these syntaxes may be identical. The presentation layer contains the functions necessary to transform between the transfer syntax and each of the other syntaxes as required.

There is not a single predetermined transfer syntax for all OSI. The transfer syntax to be used on a presentation-connection is negotiated between the correspondent presentation-entities. Thus, a presentation-entity must know the syntax of its application-entity and the agreed transfer syntax. Only the transfer syntax needs to be referred to in the presentation layer protocols.

To meet the service requirements specified by the application-entities during the intitiation phase, the presentation layer may utilize any transfer syntax available to it. To accomplish other service objectives (e.g., data volume reduction to reduce data transfer cost), syntax transformation may be performed either as a specific syntax-matching service provided to the application-entities, or as a function internal to the presentation layer.

7.2.4.2 *Negotiation of syntax*

Negotiation of syntax is carried out by communication between presentation-entities on behalf of the application-entities to determine the form that data will have while in the OSI environment. The negotiations will determine what transformations are needed (if any) and where they will be performed. Negotiations may be limited to the initiation phase or they may occur any time during a session.

In OSI, the syntaxes used by application-entities that wish to communicate may be very similar or quite dissimilar. When they are similar, the transformation functions may not be needed at all; however, when they are dissimilar, the presentation layer services provide the means to converse and decide where needed transformations will take place.

7.2.4.3 *Addressing and multiplexing*

There is a one-to-one correspondence between presentation-address and session-address. There is no multiplexing or splitting in the presentation layer.

7.2.4.4 *Presentation layer management*

The presentation layer protocols deal with some management activities of the layer (such as activation and error control). See sub-division 5.9 for relationship with other management aspects.

7.3 *Session layer*

7.3.1 *Definitions*

7.3.1.1 **quarantine service**

A facility of the session-service by which an integral number of session-service-data-units sent on a session-connection are not made available to the receiving presentation-entity until explicitly released by the sending presentation-entity.

7.3.1.2 **interaction management**

A facility of the session-service which allows correspondent presentation-entities to control explicitly whose turn it is to exercise certain control functions.

7.3.1.3 **two-way-simultaneous interaction**

A mode of interaction where both presentation-entities may concurrently send and receive.

7.3.1.4 **two-way-alternate interaction**

A mode of interaction where the presentation-entity with the turn may send and its correspondent is permitted only to receive.

7.3.1.5 **one-way interaction**

A form of operation of two-way-alternate interaction in which the turn can never be exchanged.

7.3.1.6 **session-connection synchronization**

A facility of the session-service which allows presentation-entities to define and identify synchronization points and to reset a session-connection to a predefined state and to agree on a resynchronization point.

7.3.2 *Purpose*

The purpose of the session layer is to provide the means necessary for cooperating presentation-entities to organize and synchronize their dialogue and to manage their data exchange. To do this, the session layer provides services to establish a session-connection between two presentation-entities, and to support orderly data exchange interactions.

To implement the transfer of data between the presentation-entities, the session-connection is mapped onto and uses a transport-connection (see § 7.3.4.1).

A session-connection is created when requested by a presentation-entity at a session-service-access-point. During the lifetime of the session-connection, session services are used by the presentation-entities to regulate their dialogue, and to ensure an orderly message exchange on the session-connection. The session-connection exists until it is released by either the presentation-entities or the session-entities. While the session-connection exists, session services maintain the state of the dialogue even over data loss by the transport layer.

A presentation-entity can access another presentation-entity only by initiating or accepting a session-connection. A presentation-entity may be associated with several session-connections simultaneously. Both concurrent and consecutive session-connections are possible between two presentation-entities.

The initiating presentation-entity designates the destination presentation-entity by a session-address. In many systems, a transport-address may be used as the session-address, i.e., there is a one-to-one correspondence between the session-address and the transport-address. In general, however, there is a many-to-one correspondence between session-addresses and transport-address. This does not imply multiplexing of session-connections onto transport-connections, but does imply that at session-connection establishment time, more than one presentation-entity is a potential target of a session-connection establishment request arriving on a given transport-connection.

7.3.3 *Services provided to the presentation layer*

The following services provided by the session layer are described below:

a) session-connection establishment;

b) session-connection release;

c) normal data exchange;

d) quarantine service;

e) expedited data exchange;

f) interaction management;

g) session-connection synchronization; and

h) exception reporting.

7.3.3.1 *Session-connection establishment*

The session-connection establishment service enables two presentation-entities to establish a session-connection between themselves. The presentation-entities are identified by session-addresses used to request the establishment of the session-connection.

The session-connection establishment service allows the presentation-entities cooperatively to determine the unique values of session-connection parameters at the time the session-connection is established.

Note – The provision for change of session parameters after session-connection establishment is a candidate for further extension.

Simultaneous session-connection establishment requests typically result in a corresponding number of session-connections, but a session-entity can always reject an incoming request.

The session-connection establishment service provides to the presentation-entities a session-service-connection identifier which uniquely specifies the session-connection within the environment of the correspondent presentation-entities, with a lifetime which may be greater than the lifetime of the session-connection. This identifier may be used by the presentation- entities, to refer to the session-connection during the lifetime of the session-connection, and may also be used by management-entities for administrative purposes such as accounting.

7.3.3.2 *Session-connection release*

The session-connection release service allows presentation-entities to release a session-connection in an orderly way without loss of data. It also allows either presentation-entity to request at any time that a session-connection be aborted; in this case, data may be lost.

The release of a session-connection may also be initiated by one of the session-entities supporting it.

7.3.3.3 *Normal data exchange*

The normal data exchange service allows a sending presentation-entity to transfer a session-service-data-unit to a receiving presentation-entity. This service allows the receiving presentation-entity to ensure that it is not overloaded with data.

7.3.3.4 *Quarantine service*

The quarantine service allows the sending presentation-entity to request that an integral number of session-service-data-units (one or more) sent on a session-connection should not be made available to the receiving presentation-entity until explicitly released by the sending presentation-entity. The sending presentation-entity may request that all data currently quarantined be discarded. The receiving presentation-entity receives no information that data being received has been quarantined or that some data was discarded.

7.3.3.5 *Expedited data exchange*

The expedited data exchange service provides expedited handling for the transfer of expedited session-service-data-units. A specific size restriction is placed on expedited session-service-data-units. This service may be used by either presentation-entity at any time that a session-connection exists.

7.3.3.6 *Interaction management*

The interaction management service allows the presentation-entities to control explicitly whose turn it is to exercise certain control functions.

The service provides for voluntary exchange of the turn where the presentation-entity which has the turn relinquishes it voluntarily. This service also provides for forced exchange of the turn where, upon request from the presentation-entity which does not have the turn, the session-service may force the presentation-entity with the turn to relinquish it. In the case of forced exchange of the turn, data may be lost.

The following types of session-service-data-unit exchange interaction are defined:

a) two-way-simultaneous (TWS);

b) two-way-alternate (TWA); and

c) one-way interaction.

7.3.3.7 *Session-connection synchronization*

The session-connection synchronization service allows presentation-entities to:

a) define and identify synchronization points; and

b) reset the session-connection to a defined state and agree on a resynchronization point.

The session layer is not responsible for any associated checkpointing or commitment action associated with synchronization.

7.3.3.8 *Exception reporting*

The exception reporting service permits the presentation-entities to be notified of exceptional situations not covered by other services, such as unrecoverable session malfunctions.

Note – The following services are candidates for future extensions:

a) session-service-data-unit sequence numbering;

b) brackets;

c) stop-go; and

d) security.

7.3.4 *Functions within the session layer*

The functions within the session layer are those which are performed by session-entities in order to provide the session services.

Most of the functions required are readily implied by the services provided. Additional description is given below for the following functions:

a) session-connection to transport-connection mapping;

b) session-connection flow control;

c) expedited data transfer;

d) session-connection recovery;

e) session-connection release; and

f) session layer management.

7.3.4.1 *Session-connection to transport-connection mapping*

There is a one-to-one mapping between a session-connection and a transport-connection at any given instant. However, the lifetime of a transport-connection and that of a related session-connection can be distinguished so that the following cases are defined:

a) a transport-connection supports several consecutive session-connections (see Figure 15/X.200); and

b) several consecutive transport-connections support a session-connection (see Figure 16/X.200).

Note 1 – It is also possible to consider cases in which one transport-connection is used to support several session-connections (i.e., n-to-1 mapping). In this case peer flow control would be required in the session layer. This case is for future development if needed.

Note 2 – To implement the mapping of a session-connection onto a transport-connection, the session layer maps session-service-data-units into session-protocol-data-units, and session-protocol-data-units into transport-service-data-units. These mappings may require the session-entities to perform functions such as segmenting. These functions are visible only in the session-protocols, therefore they are transparent to the presentation and transport layers.

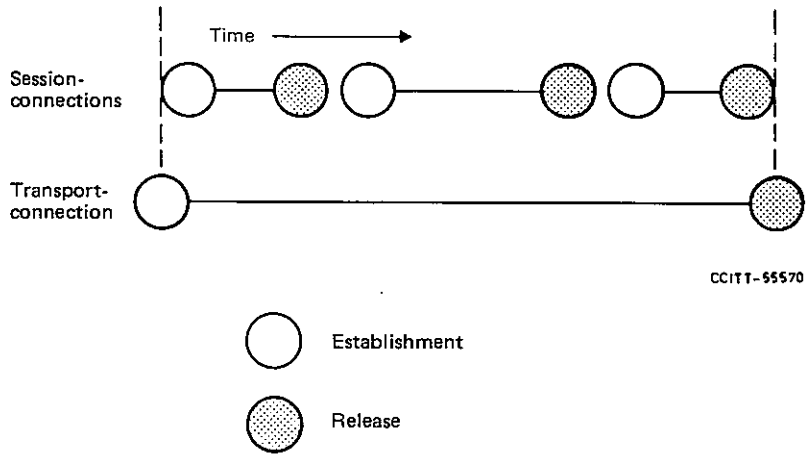


FIGURE 15/X.200

Several consecutive session-connections

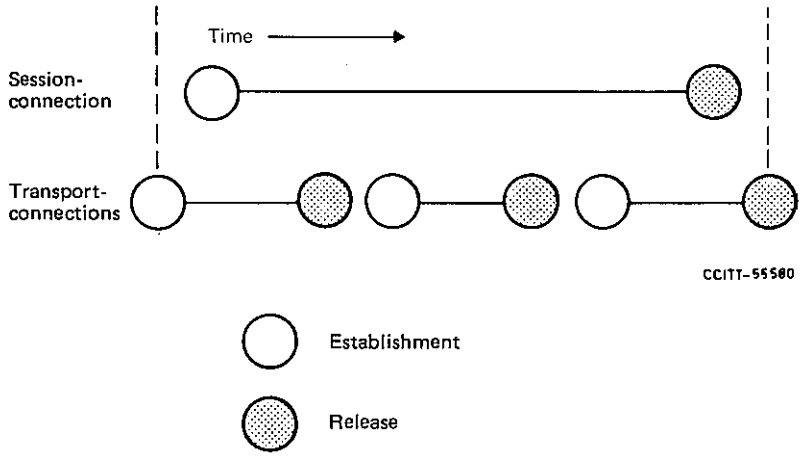


FIGURE 16/X.200

Several consecutive transport-connections

7.3.4.2 *Session-connection flow control*

There is no peer flow control in the session layer. To prevent the receiving presentation-entity from being overloaded with data, the receiving session-entity applies back pressure across the transport-connection using the transport flow control.

7.3.4.3 *Expedited data transfer*

The transfer of expedited session-service-data-units is generally accomplished by use of the expedited transport service.

7.3.4.4 *Session-connection recovery*

In the event of reported failure of an underlying transport-connection, the session layer may contain the functions necessary to re-establish a transport connection to support the session-connection, which continues to exist. The session-entities involved notify the presentation-entities via the exception reporting service that service is interrupted and restore the service only as directed by the presentation-entities. This permits the presentationentities to resynchronize and continue from an agreed state.

7.3.4.5 *Session-connection release*

The session layer contains the functions necessary to release the session-connection in an orderly way, without loss of data, upon request by the presentation-entities. The session layer also contains the necessary functions to abort the session-connection with the possible loss of data.

7.3.4.6 *Session layer management*

The session layer protocols deal with some management activities of the layer (such as activation and error control). See sub-division 5.9 for the relationship with other management aspects.

7.4 *The transport layer*

7.4.1 *Definitions*

No transport layer specific terms are identified.

7.4.2 *Purpose*

The transport-service provides transparent transfer of data between session-entities and relieves them from any concern with the detailed way in which reliable and cost effective transfer of data is achieved.

The transport layer optimizes the use of the available network-service to provide the performance required by each session-entity at minimum cost. This optimization is achieved within the constraints imposed by the overall demands of all concurrent session-entities and the overall quality and capacity of the network-service available to the transport layer.

All protocols defined in the transport layer have end-to-end significance, where the ends are defined as correspondent transport-entities. Therefore the transport layer is OSI end open system oriented and transport-protocols operate only between OSI end open systems.

The transport layer is relieved of any concern with routing, and relaying since the network-service provides network-connections from any transport-entity to any other, including the case of tandem subnetworks (see § 7.5.1).

The transport functions invoked in the transport layer to provide a requested service quality depend on the quality of the network-service. The quality of the network-service depends on the way the network-service is achieved (see § 7.5.3).

7.4.3 *Services provided to the session layer*

The transport layer uniquely identifies each session-entity by its transport-address. The transport-service provides the means to establish, maintain and release transport-connections. Transport-connections provide duplex transmission between a pair of transport-addresses.

More than one transport-connection can be established between the same pair of transport-addresses. A session-entity uses transport-connection-endpoint-identifiers provided by the transport layer to distinguish between transport-connection-endpoints.

The operation of one transport-connection is independent of the operation of all others except for the limitations imposed by the finite resources available to the transport layer.

The quality of service provided on a transport-connection depends on the service class requested by the session-entities when establishing the transport-connection. The selected quality of service is maintained throughout the lifetime of the transport-connection. The session-entity is notified of any failure to maintain the selected quality of service on a given transport-connection.

The following services provided by the transport layer are described below:

a) transport-connection establishment;

b) data transfer; and

c) transport-connection release.

7.4.3.1 *Transport-connection establishment*

Transport-connections are established between session-entities identified by transport-addresses. The quality of service of the transport-connection is negotiated between the session-entities and the transport-service.

At the time of establishment of a transport-connection the class of transport service to be provided can be selected from a defined set of available classes of service.

These service classes are characterized by combinations of selected values of parameters such as throughput, transit delay, and connection set-up delay and by guaranteed values of parameters such as residual error rate and service availability.

These classes of service represent globally predefined combinations of parameters controlling quality of service. These classes of service are intended to cover the transport-service requirements of the various types of traffic generated by the session-entities.

7.4.3.2 *Data transfer*

This service provides data transfer in accordance with the agreed quality of service. When the quality of service cannot be maintained and all possible recovery attempts have failed, the transport-connection is terminated and the session-entities are notified.

a) The transport-service-data-unit transfer-service provides the means by which transport-service-data-units of arbitrary length are delimited and transparently transferred in sequence from one sending transport-service-access-point to the receiving transport-service-access-point over a transport-connection. This service is subject to flow control.

b) The expedited transport-service-data-unit transfer service provides an additional means of information exchange on a transport-connection. The expedited transport-data-units are subject to their own set of transport-service and flow control characteristics. The maximum size of expedited transport-service data-units is limited.

7.4.3.3 *Transport-connection release*

This service provides the means by which either session-entity can release a transport-connection and have the correspondent session-entity informed of the release.

7.4.4 *Functions within the transport layer*

The transport layer functions may include:

a) mapping transport-address onto network-address;

b) multiplexing (end-to-end) transport-connections onto network-connections;

c) establishment and release of transport-connections;

d) end-to-end sequence control on individual connections;

e) end-to-end error detection and any necessary monitoring of the quality of service;

f) end-to-end error recovery;

g) end-to-end segmenting, blocking and concatenation;

h) end-to-end flow control on individual connections;

i) supervisory functions; and

j) expedited transport-service-data-unit transfer.

7.4.4.1 *Addressing*

When a session-entity requests the transport layer to establish a transport-connection with another session-entity identified by its transport-address, the transport layer determines the network-address identifying the transport-entity which serves the correspondent session-entity.

Because transport-entities support services on an end-to-end basis no intermediate transport-entity is involved as a relay between the end transport-entities. Therefore the transport layer maps transport-addresses to the network-addresses which identify the end transport-entities (see Figure 17/X.200).

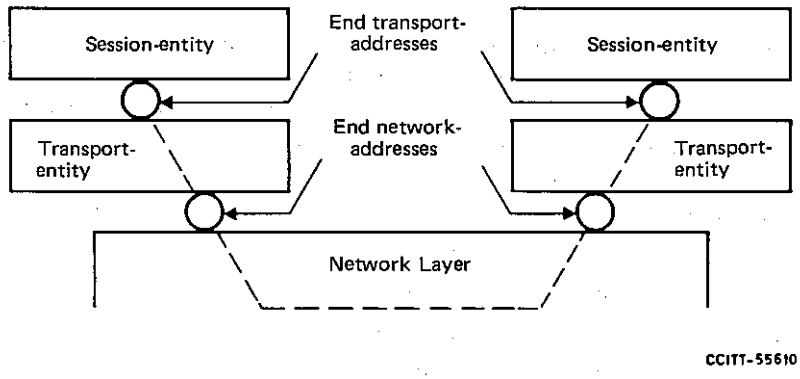


FIGURE 17/X.200

Association of transport-addresses and network-addresses

One transport-entity may serve more than one session-entity. Several transport-addresses may be associated with one network-address within the scope of the same transport-entity. Corresponding mapping functions are perfomed within the transport-entities to provide these facilities (see Figure 18/X.200).

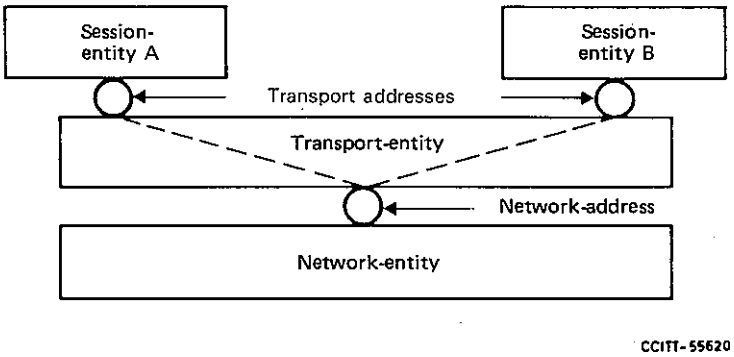


FIGURE 18/X.200

Association of one network-address with several transport-addresses

7.4.4.2 *Connection multiplexing and splitting*

In order to optimize the use of network-connections, the mapping of transport-connections onto network-connections need not be on a one-to-one basis. Both splitting and multiplexing may be performed, namely for optimizing cost of usage of the network-service.

7.4.4.3 *Phases of operation*

The phases of operation within the transport layer are:

a) establishment phase;

b) data transfer phase; and

c) release phase.

The transfer from one phase of operation to another will be specified in detail within the protocol for the transport layer.

7.4.4.4 *Establishment phase*

During the establishment phase, the transport layer establishes a transport-connection between two session-entities. The functions of the transport layer during this phase match the requested class of services with the services provided by the network layer. The following functions may be performed during this phase:

a) obtain a network-connection which best matches the requirements of the session-entity taking into account cost and quality of service;

b) decide whether multiplexing or splitting is needed to optimize the use of network connections;

c) establish the optimum transport-protocol-data-unit size;

d) select the functions that will be operational upon entering the data transfer phase;

e) map transport-addresses onto network-addresses;

f) provide identification of different transport-connections between the same pair of transport-service-access-points (connection identification function); and

g) transfer of data.

7.4.4.5 *Data transfer phase*

The purpose of the data transfer phase is to transfer transport-service-data-units between the two session-entities connected by the transport-connection. This is achieved by the transmission of transport-protocol-data-units and by the following functions, each of which is used or not used according to the class of service selected in the establishment phase:

a) sequencing;

b) blocking;

c) concatenation;

d) segmenting;

e) multiplexing or splitting;

f) flow control;

g) error detection;

h) error recovery;

i) expedited data transfer;

j) transport-service-data-unit delimiting; and

k) transport-connection identification.

7.4.4.6 *Release phase*

The purpose of the release phase is to release the transport-connection. It may include the following functions:

a) notification of reason for release;

b) identification of the transport-connection released; and

c) transfer of data.

7.4.4.7 *Transport layer management*

The transport layer protocols deal with some management activities of the layer (such as activation and error control). See sub-division 5.9 for the relationship with other management aspects.

7.5 *Network layer*

7.5.1 *Definitions*

7.5.1.1 **subnetwork**

A set of one or more intermediate open systems which provide relaying and through which end systems may establish network-connections.

Note – A subnetwork is a representation within the OSI Reference Model of a real network such as a carrier network, a private network or a local area network.

7.5.1.2 **subnetwork-connection**

A communication path through a subnetwork which is used by entities in the network layer in providing a network-connection.

7.5.2 *Purpose*

The network layer provides the means to establish, maintain and terminate network-connections between open systems containing communicating application-entities and the functional and procedural means to exchange network-service-data-units between transport-entities over network connections.

It provides to the transport-entities independence from routing and relay considerations associated with the establishment and operation of a given network-connection. This includes the case where several subnetworks are used in tandem (see § 7.5.4.2) or in parallel. It makes invisible to transport-entities how underlying resources such as data-link-connections are used to provide network-connections.

Any relay functions and hop-by-hop service enhancement protocols used to support the network-service between the OSI end open systems are operating below the transport layer, i.e., within the network layer or below.

7.5.3 *Services provided to the transport layer*

The basic service of the network layer is to provide the transparent transfer of data between transport-entities. This service allows the structure and detailed content of submitted data to be determined exclusively by layers above the network layer.

All services are provided to the transport layer at a known cost.

The network layer contains functions necessary to provide the transport layer with a firm network/transport layer boundary which is independent of the underlying communications media in all things other than quality of service. Thus the network layer contains functions necessary to mask the differences in the characteristics of different transmission and subnetwork technologies into a consistent network service.

The service provided at each end of a network-connection is the same even when a network-connection spans several subnetworks, each offering dissimilar services (see § 7.5.4.2).

Note – It is important to distinguish the specialized use of the term "service" within the OSI Reference Model from its common use by suppliers of private networks and carriers.

The quality of service is negotiated between the transport-entities and the network-service at the time of establishment of a network-connection. While this quality of service may vary from one network-connection to another it will be agreed for a given network-connection and be the same at both network-connection-endpoints.

The following services or elements of services provided by the network layer are described below:

a) network-addresses;

b) network-connections;

c) network-connection-endpoint-identifiers;

d) network-service-data-unit transfer;

e) quality of service parameters;

f) error notification;

g) sequencing;

h) flow control;

i) expedited network-service-data-unit transfer;

j) reset;

k) release; and

l) receipt of confirmation.

Some of the services described below are optional. This means that:

a) the user has to request the service; and

b) the network-service provider may honour the request or indicate that the service is not available.

7.5.3.1 *Network-addresses*

Transport-entities are known to the network layer by means of network-addresses. Network-addresses are provided by the network layer and can be used by transport-entities to identify uniquely other transport-entities, i.e., network addresses are necessary for transport-entities to communicate using the network-service. The network layer uniquely identifies each of the end open systems (represented by transport-entities) by their network-addresses. This may be independent of the addressing needed by the underlying layers.

7.5.3.2 *Network-connections*

A network-connection provides the means of transferring data between transport-entities identified by network-addresses. The network layer provides the means to establish, maintain and release network-connections.

A network-connection is point-to-point.

More than one network-connection may exist between the same pair of network-addresses.

7.5.3.3 *Network-connection-endpoint-identifiers*

The network layer provides to the transport-entity a network-connection-endpoint-identifier which identifies the network-connection-end-point uniquely with the associated network-address.

7.5.3.4 *Network-service-data-unit transfer*

On a network-connection, the network layer provides for the transmission of network-service-data-units. These units have a distinct beginning and end and integrity of the unit's content is maintained by the network layer.

No limit is imposed on the maximum size of network-service-data-units.

The network-service-data-units are transferred transparently between transport-entities.

7.5.3.5 *Quality of service parameters*

The network layer establishes and maintains a selected quality of service for the duration of the network-connection.

The quality of service parameters include residual error rate, service availability, reliability, throughput, transit delay (including variations), and delay for network-connection establishment.

7.5.3.6 *Error notification*

Unrecoverable errors detected by the network layer are reported to the transport-entities.

Error notification may or may not lead to the release of the network-connection, according to the specification of a particular network-service.

7.5.3.7 *Sequencing*

The network layer may provide sequenced delivery of network-service-data-units over a given network-connection when requested by the transport-entities.

7.5.3.8 *Flow control*

A transport-entity which is receiving at one end of a network-connection can cause the network-service to stop transferring network-service-data-units across the service-access-point. This flow control condition may or may not be propagated to the other end of the network-connection and thus be reflected to the transmitting transport-entity, according to the specification of a particular network-service.

7.5.3.9 *Expedited network-service-data-unit transfer (optional)*

The expedited network-service-data-unit transfer is optional and provides an additional means of information exchange on a network-connection. The transfer of expedited network-service-data-units is subject to a different set of network-service characteristics and to separate flow control.

The maximum size of expedited network-service-data-units is limited.

7.5.3.10 *Reset (optional)*

The reset service is optional and when invoked causes the network layer to discard all network-service-data-units in transit on the network-connection and to notify the transport-entity at the other end of the network-connection that a reset has occurred.

7.5.3.11 *Release*

A transport-entity may request release of a network-connection. The network-service does not guarantee the delivery of data preceding the release request and still in transit. The network-connection is released regardless of the action taken by the correspondent transport-entity.

7.5.3.12 *Receipt of confirmation (optional)*

A transport-entity may confirm receipt of data over a network-connection. The use of receipt confirmation service is agreed by the two users of the network-connection during connection establishment.

This service is an optional service that may not always be available.

Note – This service is included in the network service only to support existing features of CCITT Recommendation X.25.

7.5.4 *Functions within the network layer*

Network layer functions provide for the wide variety of configurations supporting network-connections ranging from network-connections supported by point-to-point configurations, to network-connections supported by complex combinations of subnetworks with different characteristics.

Note – In order to cope with this wide variety of cases, network functions should be structured into sublayers. The subdivision of the network layer into sublayers need only be done when this is useful. In particular, sublayering need not be used when the access protocol of the subnetwork supports the complete functionality of the OSI network service.

The following are functions performed by the network layer:

a) routing and relaying;

b) network-connections;

c) network-connection multiplexing;

d) segmenting and blocking;

e) error detection;

f) error recovery;

g) sequencing;

h) flow control;

i) expedited data transfer;

j) reset;

k) service selection; and

l) network layer management.

7.5.4.1 *Routing and relaying*

Network-connections are provided by network-entities in end open systems but may involve intermediate open systems which provide relaying. These intermediate open systems may interconnect subnetwork-connections, data-link-connections, and data-circuits (see § 7.7). Routing functions determine an appropriate route between network-addresses. In order to set up the resulting communication, it may be necessary for the network layer to use the services of the data link layer to control the interconnection of data-circuits (see §§ 7.6.4.10 and 7.7.3.1).

The control of interconnection of data-circuits (which are in the physical layer) from the network layer requires interaction between a network-entity and a physical entity in the same open system. Since the Reference Model permits direct interaction only between adjacent layers, the network-entity cannot interact directly with the physical-entity. This interaction is thus described as through the data link layer which intervenes "transparently" to convey the interaction between the network layer and the physical layer.

This representation is an abstract representation of something happening inside an open system and which does not model the functioning of real open systems and as such has no impact on the standardization of OSI protocols.

Note – When network layer functions are performed by combinations of several individual subnetworks, the specification of routing and relaying functions could be facilitated by using sublayers, isolating individual subnetworks routing and relaying functions from internetwork routing and relaying functions. However, when subnetworks have access protocols supporting the complete functionality of the OSI network service, there need be no sublayering in the network layer.

7.5.4.2 *Network-connections*

This function provides network-connections between transport-entities, making use of data-link-connections provided by the data link layer.

A network-connection may also be provided as subnetwork-connections in tandem, i.e., using several individual subnetworks in series. The interconnected individual subnetworks may have the same or different service capabilities. Each end of a subnetwork-connection may operate with a different subnetwork protocol.

The interconnection of a pair of subnetworks of differing qualities may be achieved in two ways. To illustrate these, consider a pair of subnetworks, one of high quality and the other of low quality:

a) The two subnetworks are interconnected as they stand. The quality of the resulting network-connection is not higher than that of the lower quality subnetwork (see Figure 19/X.200).

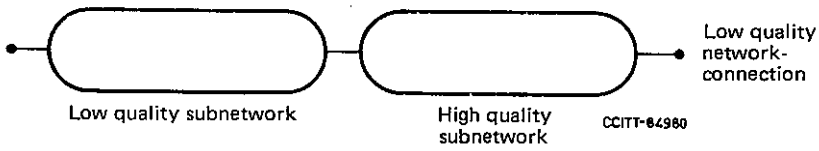


FIGURE 19/X.200

Interconnection of a low quality subnetwork and a high quality subnetwork

b) The lower quality subnetwork is enhanced equal to the higher quality subnetwork and the subnetworks are then interconnected. The quality of the resulting network-connection is aproximately that of the higher quality subnetwork (see Figure 20/X.200).

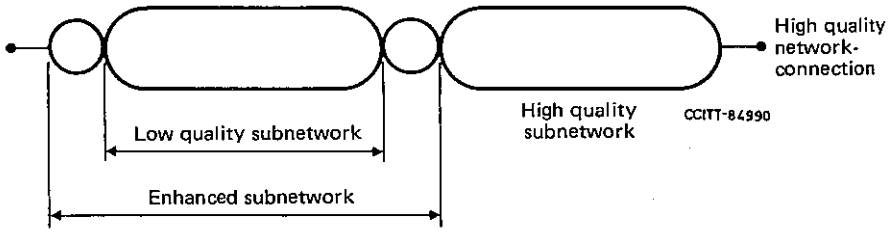


FIGURE 20/X.300

Interconnection of an enhanced low quality subnetwork and a hig quality subnetwork

The choice between these two alternatives depends on the degree of difference in quality, the cost of enhancement, and other economic factors.

7.5.4.3 *Network-connection multiplexing*

This function may be used to multiplex network-connections onto data-link-connections in order to optimize their use.

In the case of subnetwork-connections in tandem, multiplexing onto individual subnetwork-connections may also be performed in order to optimize their use.

7.5.4.4 *Segmenting and blocking*

The network layer may segment and/or block network-service-data-units for the purpose of facilitating the transfer. However, the network-service-data-unit delimiters are preserved over the network-connection.

7.5.4.5 *Error detection*

Error detection functions are used to check that the quality of service provided over a network-connection is maintained. Error detection in the network layer uses error notification from the data link layer. Additional error detection capabilities may be necessary to provide the required quality of service.

7.5.4.6 *Error recovery*

This function provides for recovering from detected errors. This function may vary depending on the quality of the network service provided.

7.5.4.7 *Sequencing*

This function provides for the sequenced delivery of network-service-data-units over a given network-connection when requested by transport-entities.

7.5.4.8 *Flow control*

If flow control service is required (see § 7.5.3.8), this function may need to be performed.

7.5.4.9 *Expedited data transfer*

This function provides for the expedited data transfer service.

7.5.4.10 *Reset*

This function provides for the reset service.

7.5.4.11 *Service selection*

This function allows service selection to be carried out to ensure that the service provided at each end of a network-connection is the same when a network-connection spans several subnetworks of dissimilar quality.

7.5.4.12 *Network layer management*

The network layer protocols deal with some management activities of the layer (such as activation and error control). See sub-division 5.9 for the relationship with other management aspects.

7.6 *Data link layer*

7.6.1 *Definitions*

No data link layer specific terms are identified.

7.6.2 *Purpose*

The data link layer provides functional and procedural means to establish, maintain and release data-link-connections among network-entities and to transfer data-link-service-data-units. A data-link-connection is built upon one or several physical-connections.

The data link layer detects and possibly corrects errors which may occur in the physical layer.

In addition, the data link layer enables the network layer to control the interconnection of data-circuits within the physical layer.

7.6.3 *Services provided to the network layer*

The following services or elements of services provided by the data link layer are described below:

a) data-link-connection;

b) data-link-service-data-units;

c) data-link-connection-endpoint-identifiers;

d) sequencing;

e) error notification;

f) flow control; and

g) quality of service parameters.

7.6.3.1 *Data-link-connection*

The data link layer provides one or more data-link-connections between two network-entities. A data-link-connection is always established and released dynamically.

7.6.3.2 *Data-link-service-data-units*

The data link layer allows exchange of data-link-service-data-units over a data-link-connection.

The size of the data-link-service-data-units may be limited by the relationship between the physical-connection error rate and the data link layer error detection capability.

7.6.3.3 *Data-link-connection-endpoint-identifiers*

If needed, the data link layer provides data-link-connection-endpoint-identifiers that can be used by a network-entity to identify a correspondent network-entity.

7.6.3.4 *Sequencing*

When required, the sequence integrity of data-link-service-data-units is maintained.

7.6.3.5 *Error notification*

Notification is provided to the network-entity when any unrecoverable error is detected by the data link layer.

7.6.3.6 *Flow control*

Each network-entity can dynamically control (up to the agreed maximum) the rate at which it receives data-link-service-data-units from a data-link-connection. This control may be reflected in the rate at which the data link layer accepts data-link-service-data-units at the correspondent data-link-connection-endpoint.

7.6.3.7 *Quality of service parameters*

Quality of service parameters may be optionally selectable. The data link layer establishes and maintains a selected quality of service for the duration of the data-link-connection. The quality of service parameters include mean time between detected but unrecoverable errors, residual error rate (where errors may arise from alteration, loss, duplication, disordering, misdelivery of data-link-service-data-unit, and other causes), service availability, transit delay and throughput.

7.6.4 *Functions within the data link layer*

The following functions performed by the data link layer are described below:

a) data-link-connection establishment and release;

b) data-link-service-data-unit mapping;

c) data-link-connection splitting;

d) delimiting and synchronization;

e) sequence control;

f) error detection;

g) error recovery;

h) flow control;

i) identification and parameter exchange;

j) control of data-circuit interconnection; and

k) data link layer management.

7.6.4.1 *Data-link-connection establishment and release*

This function establishes and releases data-link-connections on activated physical-connections. When a physical-connection has multiple endpoints (e.g., multipoint connection), a specific function is needed within the data link layer to identify the data-link-connections using such a physical-connection.

7.6.4.2 *Data-link-service-data-unit mapping*

This function maps data-link-service-data-units into data-link-protocol-data-units on a one-to-one basis.

Note – More general mappings are for further study.

7.6.4.3 *Data-link-connection splitting*

This function performs splitting of one data-link-connection onto several physical-connections.

7.6.4.4 *Delimiting and synchronization*

These functions provide recognition of a sequence of physical-service-data-units (i.e., bits, see § 7.7.3.2) transmitted over the physical-connection, as a data-link-protocol-data-unit.

Note – These functions are sometimes referred to as framing.

7.6.4.5 *Sequence control*

This function maintains the sequential order of data-link-service-data-units across a data-link-connection.

7.6.4.6 *Error detection*

This function detects transmission, format and operational errors occurring either on the physicalconnection, or as a result of a malfunction of the correspondent data-link-entity.

7.6.4.7 *Error recovery*

This function attempts to recover from detected transmission, format and operational errors and notifies the network-entities of errors which are unrecoverable.

7.6.4.8 *Flow control*

This function provides the flow control service as indicated in § 7.6.3.6.

7.6.4.9 *Identification and parameter exchange*

This function performs data-link-entity identification and parameter exchange.

7.6.4.10 *Control of data-circuit interconnection*

This function conveys to network-entities the capability of controlling the interconnection of data-circuits within the physical layer.

7.6.4.11 *Data link layer management*

The data link layer protocols deal with some management activities of the layer (such as activation and error control). See § 5.9 for the relationship with other management aspects.

7.7 *Physical layer*

7.7.1 *Definitions*

7.7.1.1 **data-circuit**

A communication path in the physical media for OSI between two physical-entities, together with the facilities necessary in the physical layer for the transmission of bits on it.

7.7.2 *Purpose*

The physical layer provides mechanical, electrical, functional and procedural means to activate, maintain and deactivate physical-connections for bit transmission between data-link-entities. A physical-connection may involve intermediate open systems, each relaying bit transmission within the physical layer. Physical layer entities are interconnected by means of a physical medium.

7.7.3 *Services provided to the data link layer*

The following services or elements of services provided by the physical layer are described below:

a) physical-connections;

b) physical-service-data-units;

c) physical-connection-endpoints;

d) data-circuit identification;

e) sequencing;

f) fault condition notification; and

g) quality of service parameters.

7.7.3.1 *Physical-connections*

The physical layer provide for the transparent transmission of bit streams between data-link-entities across physical-connections.

A data-circuit is a communication path in the physical media for OSI between two physical-entities, together with the facilities necessary in the physical layer for the transmission of bits on it.

A physical-connection may be provided by the interconnection of data-circuits using relaying functions in the physical layer. The provision of a physical-connection by such an assembly of data-circuits is illustrated in Figure 21/X.200.

The control of the interconnection of data-circuits is offered as a service to data-link-entities.

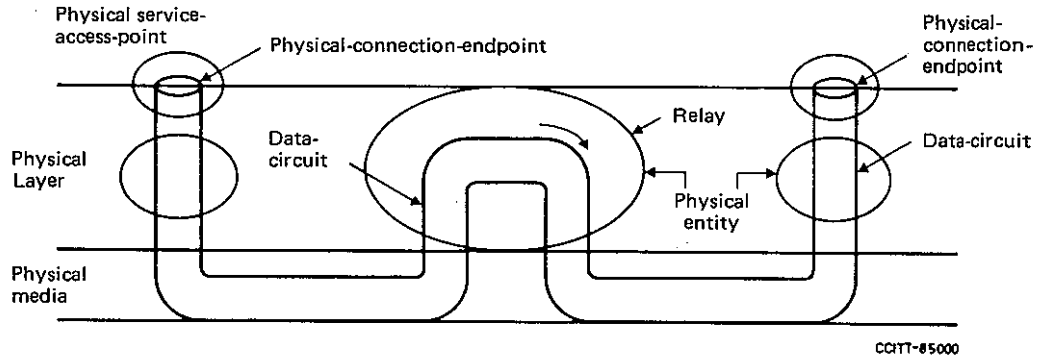


FIGURE 21/X.200

Interconnection of data-circuits within the physical layer

7.7.3.2 *Physical-service-data-units*

A physical-service-data-unit consists of one bit in serial transmission and of "n" bits in parallel transmission.

A physical-connection may allow duplex or half-duplex transmission of bit streams.

7.7.3.3 *Physical-connection-endpoints*

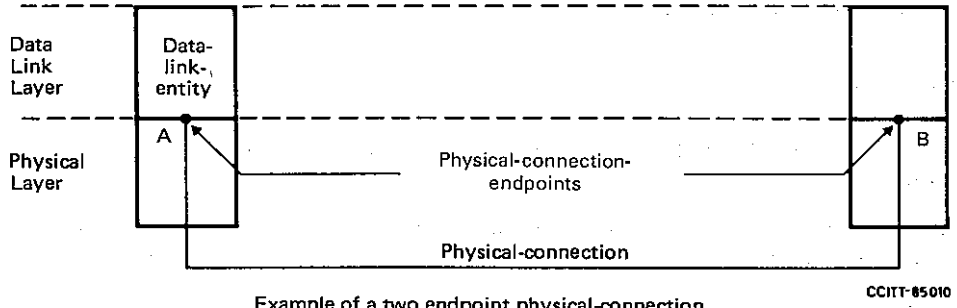
The physical layer provides physical-connection-endpoint-identifiers which may be used by a data-link-entity to identify physical-connection-endpoints.

A physical-connection will have two (point-to-point) or more (multi-endpoint) physical-connection-endpoints (see Figure 22/X.200).

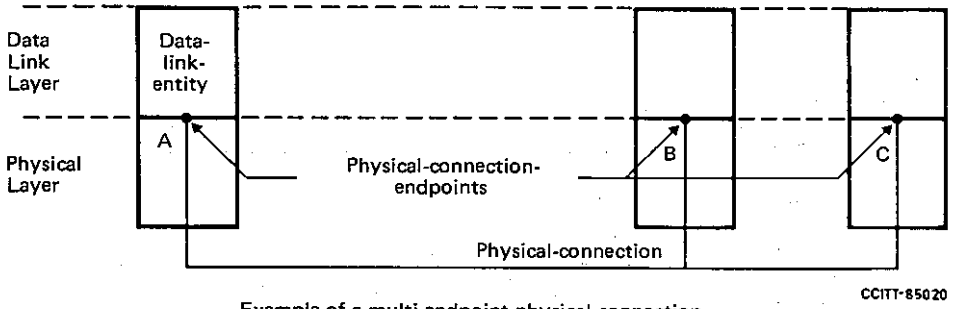
7.7.3.4 *Data-circuit identification*

The physical layer provides identifiers which uniquely specify the data-circuits between two adjacent open systems.

Note – This identifier is used by network-entities in adjacent open systems to refer to data-circuits in their dialogue.



Example of a two endpoint physical-connection
(connection exists between A and B)



Example of a multi-endpoint physical-connection
(connection exists between A, B and C)

FIGURE 22/X.200

Exemples of physical connections

7.7.3.5 *Sequencing*

The physical layer delivers bits in the same order in which they were submitted.

7.7.3.6 *Fault condition notification*

Data-link-entities are notified or fault conditions detected within the physical layer.

7.7.3.7 *Quality of service parameters*

The quality of service of a physical-connection is derived from the data-circuits forming it. The quality of service can be characterized by:

a) error rate, where errors may arise from alteration, loss, creation, and other causes;

b) service availability;

c) transmission rate; and

d) transit delay.

7.7.4 *Functions within the physical layer*

The following functions performed by the physical layer are described below:

a) physical-connection activation and deactivation;

b) physical-service-data-unit transmission; and

c) physical layer management.

7.7.4.1 *Physical-connection activation and deactivation*

These functions provide for the activation and deactivation of physical-connections between two data-link-entities upon request from the data link layer. These include a relay function which provides for interconnection of data-circuits.

7.7.4.2 *Physical-service-data-unit transmission*

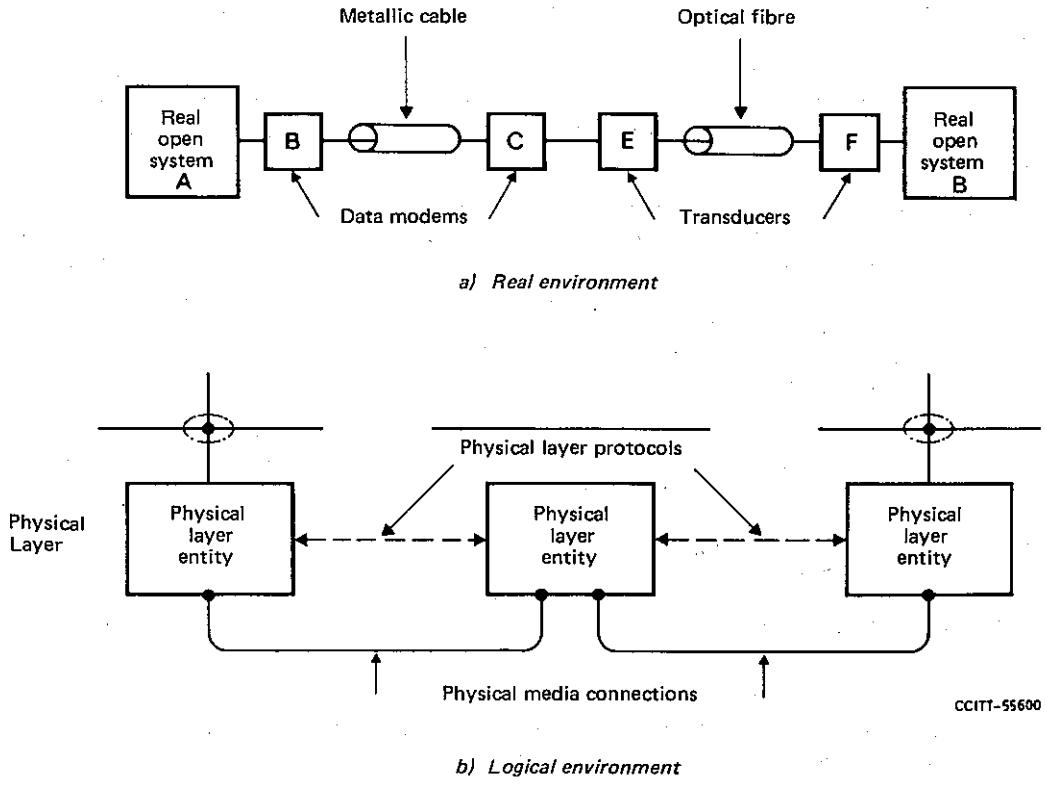
The transmission of physical-service-data-units (i.e., bits) may be synchronous or asynchronous.

7.7.4.3 *Physical layer management*

The physical layer protocols deal with some management activities of the layer (such as activation and error control). See sub-division 5.9 for the relationship with other management aspects.

Note – Relationship of the physical layer with the real environment

The above text deals with interconnection between open systems as illustrated in Figure 12/X.200. For open-systems to communicate in the real environment, real physical connections should be made, for example as in Figure 23a/X.200. Their logical representation is as shown in Figure 23b/X.200 and is called the physical media connection.



Note – The area of physical media connections in OSI requires further study.

FIGURE 23/X.200

Examples of interconnection

The mechanical, electromagnetic and other media dependant characteristics of physical media connections are defined at the boundary between the physical layer and the physical media. Definitions of such characteristics may be found in other Recommendations.

ANNEX A

(to Recommendation X.200)

Brief explanation of how the layers were chosen

This Annex provides elements giving additional information to this Recommendation, which are not an integral part of it.

The following is a brief explanation of how the layers were chosen:

a) It is essential that the architecture permit usage of a realistic variety of physical media for interconnection with different control procedures (e.g., V.24, V.25, etc.). Application of principles P3, P5 and P8 leads to identification of a Physical Layer as the lowest layer in the architecture.

b) Some physical communication media (e.g. telephone line) require specific techniques to be used in order to transmit data between systems despite a relatively high error rate (i.e., an error rate not acceptable for the great majority of applications). These specific techniques are used in data-link control procedures which have been studied and standardized for a number of years. It must also be recognized that new physical communication media (e.g., fibre optics) will require different data link control procedures. Application of principles P3, P5 and P8 leads to identification of a Data Link Layer on top of the Physical Layer in the architecture.

c) In the open system architecture, some open systems will act as the final destination of data, see division 4. Some open systems may act only as intermediate nodes (forwarding data to other open systems), see Figure 13/X.200. Application of principles P3, P5 and P7 leads to identification of a Network *Layer* on top of the Data Link Layer. Network oriented protocols such as routing, for example, will be grouped in this layer. Thus, the Network Layer will provide a connection path (network-connection) between a pair of transport-entities, including the case where intermediate nodes are involved, see Figure 13/X.200 (see also § 7.5.4.1).

d) Control data transportation from source end open system to destination end open system (which is not performed in intermediate nodes) is the last function to be performed in order to provide the totality of the transport-service. Thus, the upper layer in the transport-service part of the architecture is the Transport Layer, on top of the Network Layer. This Transport Layer relieves higher layer entities from any concern with the transportation of data between them.

e) There is a need to organize and synchronize dialogue, and to manage the exchange of data. Application of principles P3 and P4 leads to the identification of a Session Layer on top of the Transport Layer.

f) The remaining set of general interest functions are those related to representation and manipulation of structured data for the benefit of application programs. Application of principles P3 and P4 leads to identification of a Presentation Layer on top of the Session Layer.

g) Finally, there are applications consisting of application processes which perform information processing. An aspect of these application processes and the protocols by which they communicate comprise the Application Layer as the highest layer of the architecture.

The resulting architecture with seven layers, illustrated in Figure 12/X.200 obeys principles P1 and P2.

A more detailed definition of each of the seven layers identified above is given in division 7 of this Recommendation, starting from the top with the Application Layer described in § 7.1 down to the Physical Layer described in § 7.7.

ANNEX B

(to Recommendation X.200)

Alphabetical index to definitions

ITU-T RECOMMENDATIONS SERIES

Series A	Organization of the work of the ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	**Data networks and open system communications**
Series Y	Global information infrastructure and Internet protocol aspects
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland

Geneva, 2008