

JIS

JAPANESE INDUSTRIAL STANDARD

**Open Systems Interconnection
—Basic Reference Model**

JIS X 5003—1987

Translated and Published

by

Japanese Standards Association

Printed in Japan

In the event of any doubt arising,
the original Standard in Japanese is to be final authority.

Table of Contents

	Page
1. Scope	1
2. Definitions	1
3. Notation	1
4. Open Systems Interconnection (OSI) Environment and Its Modelling.....	1
4.1 Definitions	2
4.2 Open Systems Interconnection Environment.....	2
4.3 Modelling the OSI Environment.....	4
5. Concepts of a Layered Architecture.....	5
5.1 Structure of This Clause and Basic Element.....	5
5.2 Principles of Layering	6
5.3 Communication between Peer-entities	9
5.4 Identifiers	11
5.5 Properties of Service-access-points	14
5.6 Data-units	15
5.7 Elements of Layer Operation	17
5.8 Routing	25
5.9 Management Aspects of OSI.....	26
6. Introduction to the Specific OSI Layers	28
6.1 Specific Layers	28
6.2 The Principles Used to Determine the Seven Layers in the Reference Model	30
6.3 Layer Descriptions.....	31
7. Detailed Description of the Resulting OSI Architecture	31
7.1 Application Layer	31
7.2 Presentation Layer	34
7.3 Session Layer	36
7.4 Transport Layer.....	41

X 5003-1987

	Page
7.5 Network Layer.....	46
7.6 Data Link Layer.....	52
7.7 Physical Layer	55
Appendix Open Systems Interconnection-Basic Reference Model Connectionless Transmission	59
1. Scope	59
2. Definitions	59
2.1 Existing Definitions	59
2.2 Definitions	60
3. Description.....	60
4. OSI Environment and Its Modelling.....	60
5. Concepts of a Layered Architecture.....	60
5.1 Structure of This Clause and Basic Element.....	60
5.2 Principles of Layering	60
5.3 Communication between Peer-entities	62
5.4 Identifiers.....	63
5.5 Properties of Service-access-points	63
5.6 Data-units	63
5.7 Elements of Layer Operation	64
5.8 Routing	67
5.9 Management Aspects of OSI.....	67
5.10 Relations of Services at Boundaries between Adjacent Layers	68
6. Introduction to the Specific OSI Layers.....	70
6.1 Outline.....	70
6.2 General Principles	70
6.3 Assembling of Connection and Connectionless Services	71
7. Detailed Description of the Resulting OSI Architecture	72

	Page
7.1 Application Layer.....	72
7.2 Presentation Layer	73
7.3 Session Layer	74
7.4 Transport Layer.....	74
7.5 Network Layer.....	75
7.6 Data Link Layer.....	75
7.7 Physical Layer	76
Appendix Reference 1. Standards Relating to the Connectionless Transmission	77
1. Related ISO Standards	77
2. Related CCITT Recommendations	78
Appendix Reference 2. Alphabetical Index to Definitions in Appendix.....	79
Reference 1. Location of Related Standards in the Layers.....	80
1. Related JIS	80
1.1 JIS Corresponding to the Layers of Basic Reference Model..	80
1.2 JIS Made on the Basis of the Basic Reference Model	81
2. Related ISO Standards.....	81
2.1 Standards Corresponding to the Layers of Basic Reference Model	81
2.2 Standards Developed on the Basis of the Basic Reference Model	82
3. Relating CCITT Recommendations	88
3.1 Recommendations Corresponding to the Layers of Basic Reference Model	88
3.2 Recommendations Developed on the Basis of Basic Reference Model	90
Reference 2. Situation of LAN in the Basic Reference Model.....	92
Reference 3. Alphabetical Index to Definitions	93

JAPANESE INDUSTRIAL STANDARD

J I S

Open Systems Interconnection
-Basic Reference Model

X 5003-1987

1. Scope

This Japanese Industrial Standard describes the Reference Model of Open Systems Interconnection, hereinafter referred to as the "OSI". It establishes a framework for coordinating the development of existing and future standards for the interconnection of systems and is provided for reference by those standards.

This Standard does not specify services and protocols of OSI. It is neither an implementation specification for systems, nor a basis for appraising the conformance of implementations.

2. Definitions

Definitions of terms are included at the beginning of individual clauses and sub-clauses.

Reference: An index of these terms is provided in Reference 3.

3. Notation

Layers are introduced in clause 5. The following notations are 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.

These notations are also used for other concepts in the model which are related to these layers.

Example: <N>-protocol, <N + 1>-service.

Clause 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.

Example: transport-protocol, session entity, and network-service.

4. Open Systems Interconnection (OSI) Environment and Its Modelling⁽¹⁾

Note ⁽¹⁾ The general principles described in this clause hold for all layers, unless layer specific statements to the contrary are made in clauses 6. and 7.

Corresponding International Standards:

ISO 7498-Open systems interconnection-Basic reference model

ISO 7498/AD 1-Open systems interconnection-Basic reference model

Addendum 1: Connectionless-Mode Transmission

4.1 Definitions

- (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.
- (2) real open system A real system which complies with the requirements of OSI standards in its communication with other real systems.
- (3) open system The representation within the Reference Model of those aspects of a real open system that are pertinent to OSI.
- (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:

- Examples 1: a person operating a banking terminal is a manual application-process;
- 2: 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
- 3: 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 Fig. 1, the physical media for Open Systems Interconnection⁽²⁾ provide the means for the transfer of information between open systems.

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⁽³⁾ between systems, which is implied by the expression "systems interconnection".

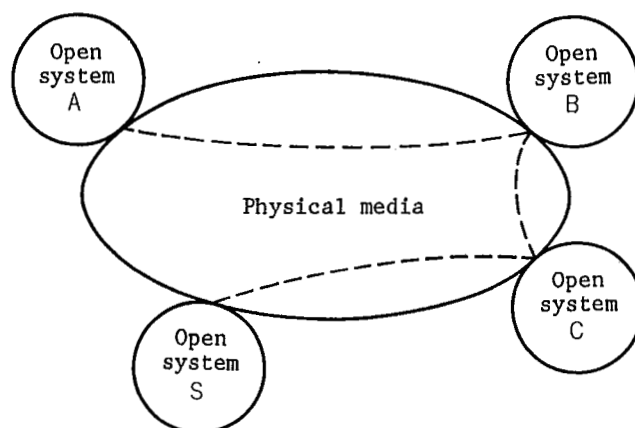
The objective of OSI is to define a set of standards to enable real open systems to cooperate. A system which complies with the requirements of applicable OSI standards in its cooperation with other systems is termed a real open system.

Notes ⁽²⁾ At this point, only telecommunications media have been considered. The use of other interconnection media is for further study.

⁽³⁾ 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 concern information processing constraints that have to 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.

Fig. 1. Open Systems Connected by Physical Media



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 Standard covers the elements of OSI aspects of these activities which are essential for early development of OSI standards.

4.3 Modelling the OSI Environment The development of OSI standards, i.e. standards 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 Standard.

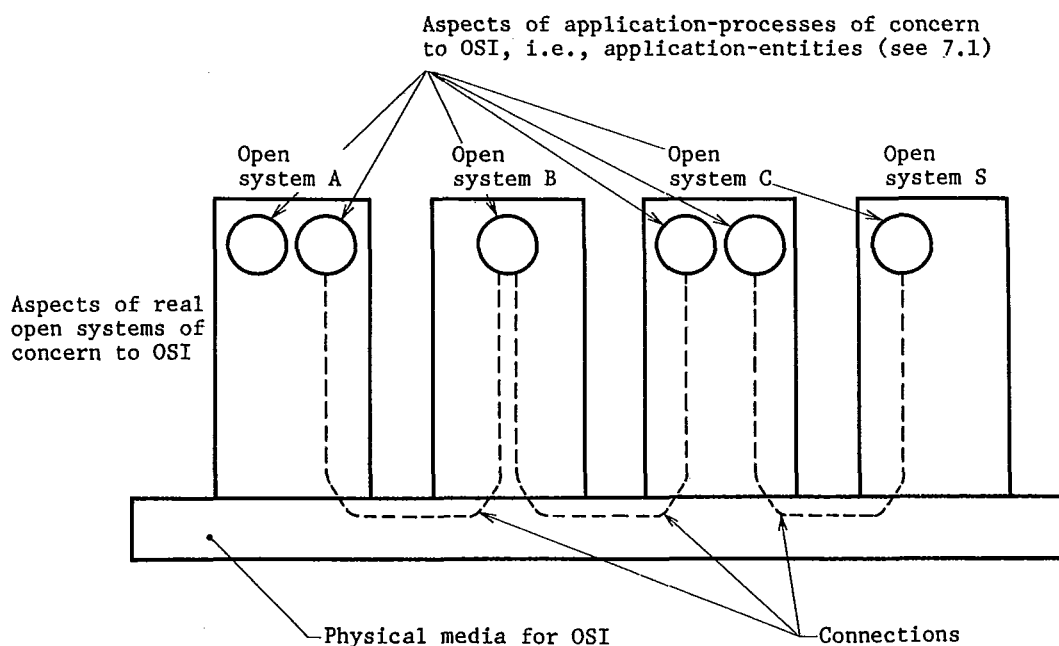
Then, the detailed and precise description of the functioning of the open system is developed in the framework formed by the Reference Model. This constitutes the services and protocols for OSI which are the subject of other Standards.

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.

It is needed to be 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 Standard, only the aspects of real systems and application-processes which lie within the OSI environment are considered. Their interconnection is illustrated throughout this Standard as depicted in Fig. 2.

Fig. 2. Basic Elements of OSI



5. Concepts of a Layered Architecture⁽⁴⁾

Note ⁽⁴⁾ The general principles described in this clause hold for all layers, unless layer specific statements to the contrary are made in clauses 6. and 7.

5.1 Structure of This Clause and Basic Element Clause 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 this clause are those required to describe in clause 6. and thereafter. 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 Fig. 2):

- (1) Open systems;
- (2) The application-entities which exist within the OSI environment;
- (3) The connections (see 5.3) which join the application-entities and permit them to exchange information⁽⁵⁾; and
- (4) The physical media for OSI.

Note (5) This Standard is based on the assumption that a connection is required for the transfer of data. An appendix to this Standard extends the description to cover the connectionless forms of data transmission which may be found in a wide variety of data communications techniques (for example local area networks, digital radio, etc.) and applications (for example remote sensing and banking).

Remark: Security aspects which are also general architectural elements of protocols are not discussed in this Standard.

5.2 Principles of Layering

5.2.1 Definitions

- (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.
- (2) <N>-layer A subdivision of the OSI architecture, constituted by subsystems of the same rank <N>.
- (3) <N>-entity An active element within an <N>-subsystem.
- (4) peer-entities Entities within the same layer.
- (5) sublayer A subdivision of a layer.
- (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.
- (7) <N>-facility A part of an <N>-service.
- (8) <N>-function A part of the activity of <N>-entities.
- (9) <N>-service-access-point The point at which <N>-services are provided by an <N>-entity to an <N + 1>-entity.
- (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 being logically composed of an ordered set of subsystems, represented for convenience in the vertical sequence shown in Fig. 3. Adjacent subsystems communicate through their common boundary. Subsystems of the same rank $\langle N \rangle$ collectively form the $\langle N \rangle$ -layer of the Reference Model of Open Systems Interconnection. An $\langle N \rangle$ -subsystem consists of one or several $\langle N \rangle$ -entities. Entities exist in each layer. Entities in the same layer are termed peer-entities. Note that the highest layer does not have an $\langle N + 1 \rangle$ -layer above it and the lowest layer does not have an $\langle N - 1 \rangle$ -layer below it. Not all peer $\langle N \rangle$ -entities need or even can communicate. There may be conditions which prevent this communication (for example: they are not in interconnected open systems, or they do not support the same protocol subsets).

Remarks 1. 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 instances of this type can be referred to by an individual name. Each nameable instance and the type to which this instance belongs should 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 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 one displays an instance.

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

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

<N>-entity instance, it should be able to request another connection to that <N>-entity instance.

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 OSI. A sublayer is defined 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.

- Remarks 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.
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 co-operation of other <N>-entities to help complete the service request. In order to co-operate, <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 Fig. 4). The entities in the lowest layer are assumed to communicate directly via the physical media which connect them.

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 Fig. 7).

Co-operation between <N>-entities is governed by one or more <N>-protocols. The entities and protocols within a layer are illustrated in Fig. 5.

Fig. 3. Layering in Co-operating Open Systems

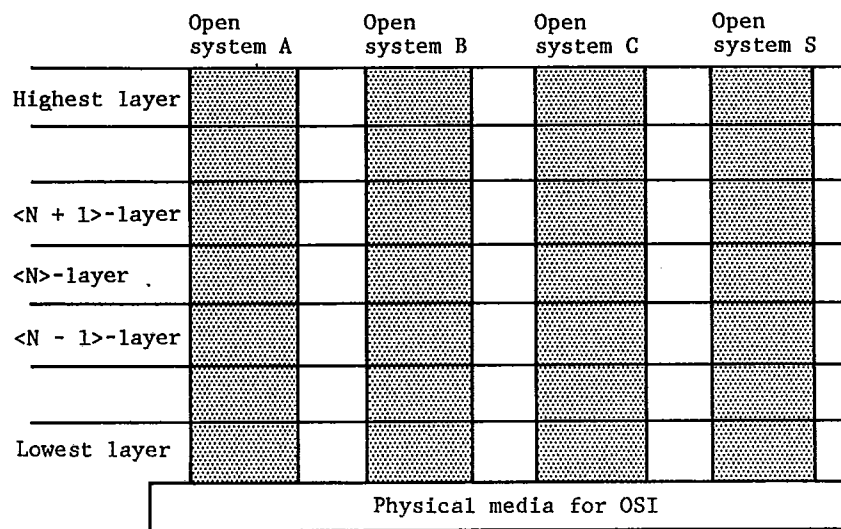


Fig. 4. <N>-entities in the <N>-layer Communicate through the <N - 1>-layer

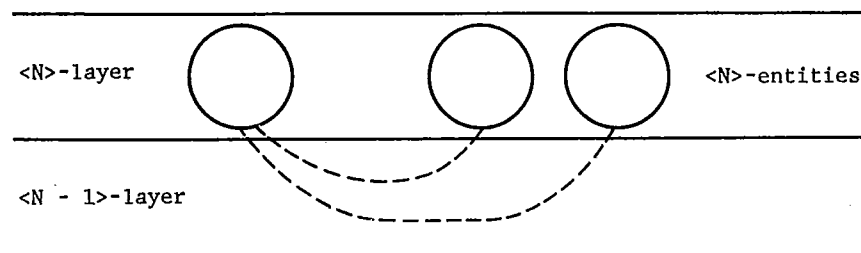
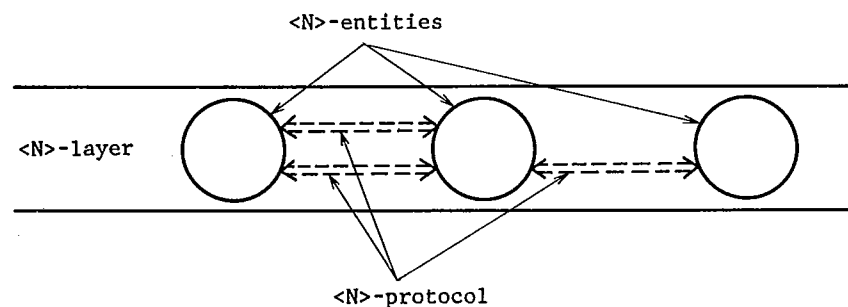


Fig. 5. <N>-protocols between <N>-entities



5.3 Communication between Peer-entities

5.3.1 Definitions

- (1) <N>-connection An association established by the <N>-layer between two or more <N + 1>-entities for the transfer of data.
- (2) <N>-connection-endpoint A terminator at one end of an <N>-connection within an <N>-service-access-point.

- (3) multi-endpoint-connection A connection with more than two connection-endpoints.
- (4) correspondent <N>-entities <N>-entities with an <N - 1>-connection between them.
- (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.
- (6) <N>-data-source An <N>-entity that sends <N - 1>-service-data-units [see 5.6.1 (7)] on an <N - 1>-connection.
- (7) <N>-data-sink An <N>-entity that receives <N - 1>-service-data-units on an <N - 1>-connection.
- (8) <N>-data-transmission An <N>-facility which conveys <N>-service-data-units from one <N + 1>-entity to one or more <N + 1>-entities.
- (9) <N>-duplex-transmission <N>-data transmission in both directions at the same time.
- (10) <N>-half-duplex-transmission <N>-data transmission in either direction, one direction at a time; the choice of direction is controlled by an <N + 1>-entity.
- (11) <N>-simplex-transmission <N>-data-transmission in one pre-assigned direction.
- (12) <N>-data-communication 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.
- (13) <N>-two-way-simultaneous communication <N>-data-communication in both directions at the same time.
- (14) <N>-two-way-alternate communication <N>-data-communication in both directions, one direction at a time.
- (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 shall be established between them in the <N>-layer using an <N>-protocol.

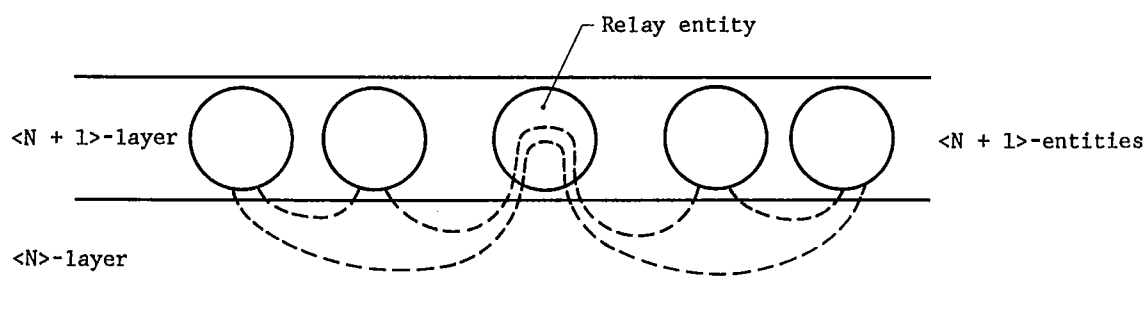
Remark: Classes of protocols may be defined within the <N>-protocols. The precise definition of the term "classes of protocols" is for further study.

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 Fig. 6).

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.

Fig. 6. Communication through a Relay



5.4 Identifiers

5.4.1 Definitions

- (1) title A permanent identifier for an entity.
- (2) title-domain A subset of the title space of the OSI environment.
- (3) title-domain-name An identifier which uniquely identifies a title-domain within the OSI environment.

Remark: Title-domains of primary importance are the layers. In this specific case, the title-domain-name identifies the <N>-layer.

- (4) local-title A title which is unique within a title-domain.
- (5) global-title A title which is unique within the OSI environment and comprises two parts, a title-domain-name and a local-title.
- (6) <N>-address or <N>-service-access-point-address An identifier which tells where an <N>-service-access-point may be found.
- (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.
- (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.

- (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.
- (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.
- (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.
- (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.
- (13) <N>-service-connection-identifier An identifier which uniquely specifies an <N>-connection within the environment of the correspondent <N + 1>-entities.
- (14) <N>-protocol-connection-identifier An identifier which uniquely specifies an individual <N>-connection within the environment of the multiplexed <N - 1>-connection.
- (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 Fig. 7). 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.

The use of an <N>-address to identify an <N + 1>-entity is the most efficient mechanism if the permanence of 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:

- (1) hierarchical <N>-address-mapping; and
- (2) <N>-address-mapping by tables.

If an $\langle N \rangle$ -address is always mapped into only one $\langle N - 1 \rangle$ -address then hierarchical construction of addresses can be used (see Fig. 8). The $\langle N \rangle$ -address-mapping function need only recognize the hierarchical structure of an $\langle N \rangle$ -address and extract the $\langle N - 1 \rangle$ -address it contains.

In this case, an $\langle N \rangle$ -address consists of two parts:

- (1) An $\langle N - 1 \rangle$ -address of the $\langle N \rangle$ -entity which is supporting the current $\langle N \rangle$ -service-access-point of the $\langle N + 1 \rangle$ -entity;
- (2) An $\langle N \rangle$ -suffix which makes the $\langle N \rangle$ -service-access-point uniquely identifiable within the scope of the $\langle N - 1 \rangle$ -address.

Within a given layer, a hierarchical structure of addresses simplifies $\langle N \rangle$ -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 $\langle N \rangle$ -address-mappings and to cover the case where an $\langle N \rangle$ -entity attached to more than one $\langle N - 1 \rangle$ -service-access-point supports only one $\langle N \rangle$ -service-access-point.

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

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

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

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

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

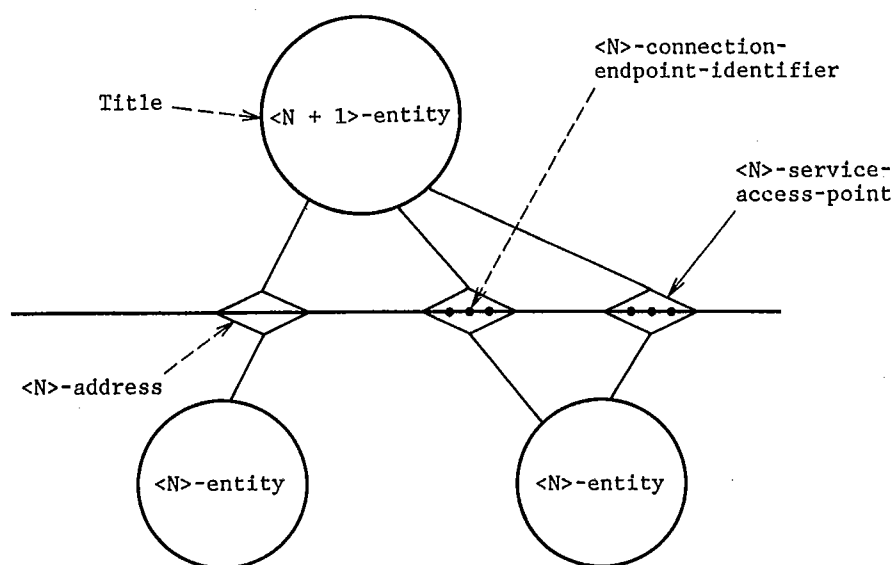
The $\langle N \rangle$ -connection-endpoint-identifier consists of two parts:

- (1) the $\langle N \rangle$ -address of the $\langle N \rangle$ -service-access-point which will be used in conjunction with the $\langle N \rangle$ -connection; and
- (2) an $\langle N \rangle$ -connection-endpoint-suffix which is unique within the scope of the $\langle N \rangle$ -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 shall be unique within the scope of the connection within which it is used.

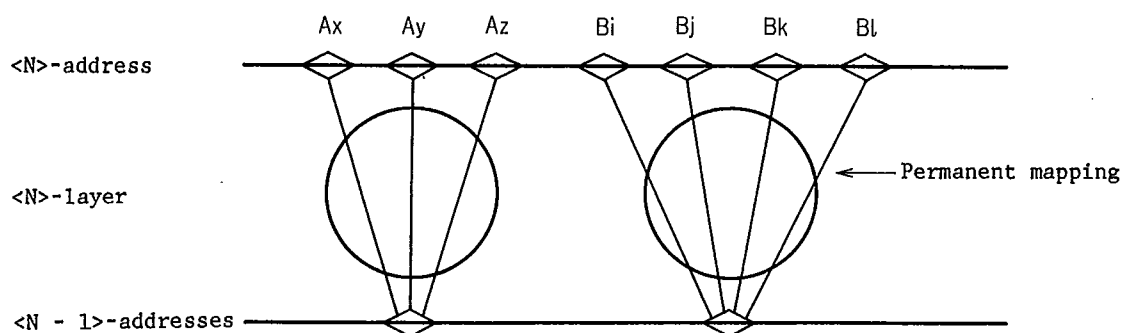
The $\langle N \rangle$ -layer may provide to the $\langle N + 1 \rangle$ -entities an $\langle N \rangle$ -service-connection-identifier which uniquely specifies the $\langle N \rangle$ -connection within the environment of the correspondent $\langle N + 1 \rangle$ -entities.

Fig. 7. Entities, Service-access-points, and Identifiers



Remark: Dashed arrows refer to identifiers.

Fig. 8. Hierarchical $\langle N \rangle$ -address-mapping



5.5 Properties of Service-access-points An $\langle N + 1 \rangle$ -entity requests $\langle N \rangle$ -services via an $\langle N \rangle$ -service-access-point which permits the $\langle N + 1 \rangle$ -entity to interact with an $\langle N \rangle$ -entity.

Both the $\langle N \rangle$ - and $\langle N + 1 \rangle$ -entities attached to an $\langle N \rangle$ -service-access-point are in the same system.

An $\langle N + 1 \rangle$ -entity may concurrently be attached to one or more $\langle N \rangle$ -service-access-points attached to the same or different $\langle N \rangle$ -entities.

An $\langle N \rangle$ -entity may concurrently be attached to one or more $\langle N + 1 \rangle$ -entities through $\langle N \rangle$ -service-access-points.

An $\langle N \rangle$ -service-access-point is attached to only one $\langle N \rangle$ -entity and to only one $\langle N + 1 \rangle$ -entity at a time.

An $\langle N \rangle$ -service-access-point may be detached from an $\langle N + 1 \rangle$ -entity and reattached to the same or another $\langle N + 1 \rangle$ -entity.

An $\langle N \rangle$ -service-access-point may be detached from an $\langle N \rangle$ -entity and reattached to the same or another $\langle N \rangle$ -entity.

An $\langle N \rangle$ -service-access-point is located by means of its $\langle N \rangle$ -address. An $\langle N \rangle$ -address is used by an $\langle N + 1 \rangle$ -entity to request an $\langle N \rangle$ -connection.

5.6 Data-units

5.6.1 Definitions

- (1) $\langle N \rangle$ -protocol-control-information Information exchanged between $\langle N \rangle$ -entities, using an $\langle N - 1 \rangle$ -connection, to co-ordinate their joint operation.
- (2) $\langle N \rangle$ -user-data The data transferred between $\langle N \rangle$ -entities on behalf of the $\langle N + 1 \rangle$ -entities for whom the $\langle N \rangle$ -entities are providing services.
- (3) $\langle N \rangle$ -protocol-data-unit A unit of data specified in an $\langle N \rangle$ -protocol and consisting of $\langle N \rangle$ -protocol-information and possibly $\langle N \rangle$ -user-data.
- (4) $\langle N \rangle$ -interface-control-information Information transferred between an $\langle N + 1 \rangle$ -entity and an $\langle N \rangle$ -entity to coordinate their joint operation.
- (5) $\langle N \rangle$ -interface-data Information transferred from an $\langle N + 1 \rangle$ -entity to an $\langle N \rangle$ -entity for transmission to a correspondent $\langle N + 1 \rangle$ -entity over an $\langle N \rangle$ -connection, or conversely, information transferred from an $\langle N \rangle$ -entity to an $\langle N + 1 \rangle$ -entity after being received over an $\langle N \rangle$ -connection from a correspondent $\langle N + 1 \rangle$ -entity.
- (6) $\langle N \rangle$ -interface-data-unit The unit of information transferred across the $\langle N \rangle$ -service-access-point between an $\langle N + 1 \rangle$ -entity and an $\langle N \rangle$ -entity in a single interaction. Each $\langle N \rangle$ -interface-data-unit contains $\langle N \rangle$ -interface-control-information and may also contain the whole or part of an $\langle N \rangle$ -service-data-unit.
- (7) $\langle N \rangle$ -service-data-unit An amount of $\langle N \rangle$ -interface-data whose identity is preserved from one end of an $\langle N \rangle$ -connection to the other.

- (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.

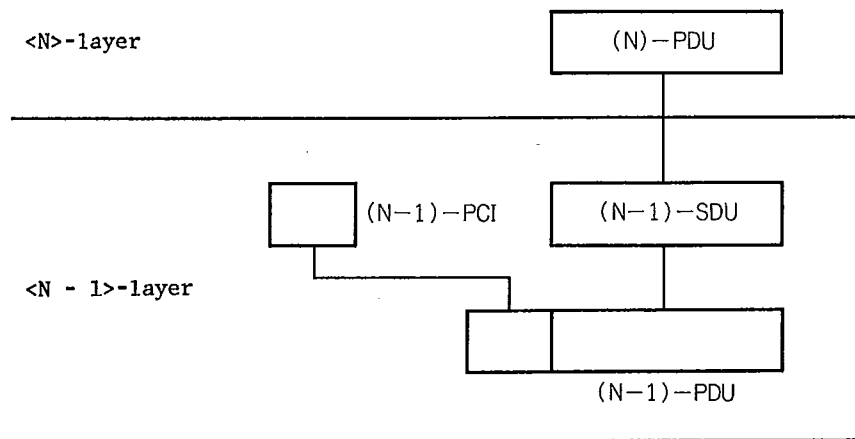
5.6.2 Description Information is transferred in various types of data-units between peer-entities and between entities attached to a specific service-access-point. The data-units are defined in 5.6.1 and the relationships among them are illustrated in Table and Fig. 9.

Except for the relationships defined in Table and Fig. 9, there is no overall architectural limit to the size of data-units. There may be other size limitations at specific layers.

Table Relationships among Data-units

Division	Control-information	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

Fig. 9. An Illustration of Mapping between Data-units in Adjacent Layers



Remarks 1. PCI = protocol-control-information

PDU = protocol-data-unit

SDU = service-data-unit

- This figure assumes that neither segmenting nor blocking of <N>-service-data-units is performed (see 5.7.6.5).
- This figure does not imply any positional relationship between protocol-control-information and user-data in protocol-data-units.

4. An $\langle N \rangle$ -protocol-data-unit may be mapped one-to-one into an $\langle N - 1 \rangle$ -service-data-unit, but other relationships are possible (see Fig. 10).

Date 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

- (1) $\langle N \rangle$ -protocol-identifier An identifier used between correspondent $\langle N \rangle$ -entities to select a specific $\langle N \rangle$ -protocol to be used on a particular $\langle N - 1 \rangle$ -connection.
- (2) centralized multi-endpoint-connection A multi-endpoint-connection where data sent by an $\langle N + 1 \rangle$ -entity associated with central $\langle N \rangle$ -connection-endpoint is received by all other $\langle N + 1 \rangle$ -entities, while data sent by one of the other $\langle N + 1 \rangle$ -entities is only received by a central $\langle N + 1 \rangle$ -entity.
- (3) decentralized multi-endpoint-connection A multi-endpoint-connection such that data sent by an $\langle N + 1 \rangle$ -entity associated with an $\langle N \rangle$ -connection-endpoint is received by all other $\langle N + 1 \rangle$ -entities.
- (4) multiplexing A function within the $\langle N \rangle$ -layer by which one $\langle N - 1 \rangle$ -connection is used to support more than one $\langle N \rangle$ -connection.

Remark: The term multiplexing is also used in a more restricted sense to refer to the function performed by the sending $\langle N \rangle$ -entity while the term demultiplexing is used to refer to the function performed by the receiving $\langle N \rangle$ -entity.

- (5) demultiplexing The function performed by an $\langle N \rangle$ -entity which identifies $\langle N \rangle$ -protocol-data-units for more than one $\langle N \rangle$ -connection within $\langle N - 1 \rangle$ -service-data-units received on a single $\langle N - 1 \rangle$ -connection. It is the reverse function of the multiplexing function performed by the $\langle N \rangle$ -entity sending the $\langle N - 1 \rangle$ -service-data-units.
- (6) splitting A function within the $\langle N \rangle$ -layer by which more than one $\langle N - 1 \rangle$ -connection is used to support one $\langle N \rangle$ -connection.

Remark: The term splitting is also used in a more restrictive sense to refer to the function performed by the sending $\langle N \rangle$ -entity while the term recombining is used to refer to the function performed by the receiving $\langle N \rangle$ -entity.

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

- (8) flow control A function which controls the flow of data within a layer or between adjacent layers.
- (9) segmenting A function performed by an $\langle N \rangle$ -entity to map one $\langle N \rangle$ -service-data-unit into multiple $\langle N \rangle$ -protocol-data-units.
- (10) reassembling A function performed by an $\langle N \rangle$ -entity to map multiple $\langle N \rangle$ -protocol-data-units into one $\langle N \rangle$ -service-data-unit. It is the reverse function of segmenting.
- (11) blocking A function performed by an $\langle N \rangle$ -entity to map multiple $\langle N \rangle$ -service-data-units into one $\langle N \rangle$ -protocol-data-unit.
- (12) deblocking A function performed by an $\langle N \rangle$ -entity to identify multiple $\langle N \rangle$ -service-data-units which are contained in one $\langle N \rangle$ -protocol-data-unit. It is the reverse function of blocking.
- (13) concatenation A function performed by an $\langle N \rangle$ -entity to map multiple $\langle N \rangle$ -protocol-data-units into one $\langle N - 1 \rangle$ -service-data-unit.
- (14) separation A function performed by an $\langle N \rangle$ -entity to identify multiple $\langle N \rangle$ -protocol-data-units which are contained in one $\langle N - 1 \rangle$ -service-data-unit. It is the reverse function of concatenation.
- (15) sequencing A function performed by the $\langle N \rangle$ -layer to preserve the order of $\langle N \rangle$ -service-data-units that were submitted to the $\langle N \rangle$ -layer from the $\langle N + 1 \rangle$ -layer.
- (16) acknowledgement A function of the $\langle N \rangle$ -layer which allows a receiving $\langle N \rangle$ -entity to inform a sending $\langle N \rangle$ -entity of the receipt of an $\langle N \rangle$ -protocol-data-unit.
- (17) reset A function which sets the correspondent $\langle N \rangle$ -entities to a predefined state with a possible loss or duplication of data.

5.7.2 Protocol Selection One or more $\langle N \rangle$ -protocols may be defined for the $\langle N \rangle$ -layer. An $\langle N \rangle$ -entity may employ one or more $\langle N \rangle$ -protocols.

Meaningful communication between $\langle N \rangle$ -entities over an $\langle N - 1 \rangle$ -connection requires the agreed selection of one $\langle N \rangle$ -protocol.

$\langle N \rangle$ -protocol-identifiers name the specific protocols defined.

5.7.3 Properties of Connections An $\langle N \rangle$ -connection is an association established for communication between two or more $\langle N + 1 \rangle$ -entities, identified by their $\langle N \rangle$ -addresses. An $\langle N \rangle$ -connection is offered as a service by the $\langle N \rangle$ -layer, so that information may be exchanged between the $\langle N + 1 \rangle$ -entities.

An $\langle N + 1 \rangle$ -entity may have, simultaneously, one or more $\langle N \rangle$ -connections with other $\langle N + 1 \rangle$ -entities, with any given $\langle N + 1 \rangle$ -entity, and with itself.

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

The source $\langle N \rangle$ -address and one or more of the destination $\langle N \rangle$ -addresses may be the same. One or more of the destination $\langle N \rangle$ -addresses may be the same while the source $\langle N \rangle$ -address is different. All may be different.

One $\langle N \rangle$ -connection-endpoint is constructed for each $\langle N \rangle$ -address referenced explicitly or implicitly when an $\langle N \rangle$ -connection is established.

An $\langle N + 1 \rangle$ -entity accesses an $\langle N \rangle$ -connection via an $\langle N \rangle$ -service-access-point.

An $\langle N \rangle$ -connection has two or more $\langle N \rangle$ -connection-endpoints.

An $\langle N \rangle$ -connection-endpoint is not shared by $\langle N + 1 \rangle$ -entities or $\langle N \rangle$ -connections.

An $\langle N \rangle$ -connection-endpoint relates three elements:

- (1) an $\langle N + 1 \rangle$ -entity;
- (2) an $\langle N \rangle$ -entity; and
- (3) an $\langle N \rangle$ -connection.

The $\langle N \rangle$ -entity and the $\langle N + 1 \rangle$ -entity related by an $\langle N \rangle$ -connection-endpoint are those implied by the $\langle N \rangle$ -address referenced when the $\langle N \rangle$ -connection is established.

An $\langle N \rangle$ -connection-endpoint has an identifier, called an $\langle N \rangle$ -connection-endpoint-identifier, which is unique within the scope of the $\langle N + 1 \rangle$ -entity which is bound to the $\langle N \rangle$ -connection-endpoint.

An $\langle N \rangle$ -connection-endpoint-identifier is not the same as an $\langle N \rangle$ -address.

An $\langle N + 1 \rangle$ -entity references an $\langle N \rangle$ -connection using its $\langle N \rangle$ -connection-endpoint-identifier.

Multi-endpoint-connections are connections which have three or more connection-endpoints. Two types of multi-endpoint-connection are defined:

- (1) centralized; and
- (2) decentralized.

Remark: 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:

- (1) the availability of an <N - 1>-connection between the supporting <N>-entities; and
- (2) 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 OSI 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 taken 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 [for example time of failure of the <N - 1>-connection and common time-out]. In addition, both <N>-entities shall be 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 reestablished, 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:

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

- (2) 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:

- (1) one-to-one;
- (2) many <N>-connections to one <N - 1>-connection (multiplexing); and
- (3) one <N>-connection to many <N - 1>-connections (splitting).

Remarks 1. Multiplexing may be needed in order to:

- (1) make more efficient or more economic use of the <N - 1>-service; and
- (2) provide several <N>-connections in an environment where only a single <N - 1>-connection exists.

2. Splitting may be needed in order to:

- (1) improve reliability since more than one <N - 1>-connection is available;
- (2) provide the required grade of performance, through the utilization of multiple <N - 1>-connections; and
- (3) 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:

- (1) 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;
- (2) flow control on each <N>-connection in order to share the capacity of the <N - 1>-connection; and

- (3) scheduling the next $\langle N \rangle$ -connection to be serviced over the $\langle N - 1 \rangle$ -connection when more than one $\langle N \rangle$ -connection is prepared to send data.

The functions associated with splitting are:

- (1) scheduling the utilization of multiple $\langle N - 1 \rangle$ -connections used in splitting a single $\langle N \rangle$ -connection; and
- (2) resequencing of $\langle N \rangle$ -protocol-data-units associated with an $\langle N \rangle$ -connection since they may arrive out of sequence even when each $\langle N - 1 \rangle$ -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 $\langle N \rangle$ -entities in $\langle N \rangle$ -protocol-data-units. An $\langle N \rangle$ -protocol-data-unit is a unit of data specified in an $\langle N \rangle$ -protocol and contains $\langle N \rangle$ -protocol-control-information and possibly $\langle N \rangle$ -user-data.

$\langle N \rangle$ -protocol-control-information is transferred between $\langle N \rangle$ -entities using the $\langle N - 1 \rangle$ -connection. $\langle N \rangle$ -protocol-control-information is any information that supports the joint operation of $\langle N \rangle$ -entities. $\langle N \rangle$ -user-data is passed transparently between $\langle N \rangle$ -entities over an $\langle N - 1 \rangle$ -connection.

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

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

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

5.7.6.2 Date Transfer during Connection Establishment and Release $\langle N \rangle$ -user-data may be transferred by an $\langle N \rangle$ -entity in the $\langle N \rangle$ -connection establishment protocol exchange and in the $\langle N \rangle$ -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 $\langle N \rangle$ -user-data between correspondent $\langle N + 1 \rangle$ -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 $\langle N \rangle$ -service-data-unit is intended to be processed by the receiving $\langle N + 1 \rangle$ -entity with priority over normal $\langle N \rangle$ -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:

- (1) peer flow control which regulates the rate at which $\langle N \rangle$ -protocol-data-units are sent to the peer $\langle N \rangle$ -entity on the $\langle N \rangle$ -connection. Peer flow control requires protocol definitions and is based on protocol-data-unit size; and
- (2) $\langle N \rangle$ -interface flow control which regulates the rate at which $\langle N \rangle$ -interface-data are passed between an $\langle N + 1 \rangle$ -entity and an $\langle N \rangle$ -entity. $\langle N \rangle$ -interface flow control is based on the $\langle N \rangle$ -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 flow control information to be included in the $\langle N \rangle$ -protocol-control-information of an $\langle N \rangle$ -protocol-data-unit.

If the size of service-data-units exceeds the maximum size of the $\langle N \rangle$ -user-data portion of an $\langle N \rangle$ -protocol-data-unit, then first segmentation shall be performed on the $\langle N \rangle$ -service-data-unit to make it fit within the $\langle N \rangle$ -protocol-data-units. Peer flow control can then be applied on the $\langle N \rangle$ -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 $\langle N \rangle$ -service-data-unit into more than one $\langle N \rangle$ -protocol-data-units. Similarly, segmenting may occur when $\langle N \rangle$ -protocol-data-units are mapped into $\langle N - 1 \rangle$ -interface-data-units. Since it is necessary to preserve the identity of $\langle N \rangle$ -service-data-units on an $\langle N \rangle$ -connection, functions shall be available to identify the segments of an $\langle N \rangle$ -service-data-unit, and to allow the correspondent $\langle N \rangle$ -entities to reassemble the $\langle N \rangle$ -service-data-unit.

Segmenting may require that information be included in the $\langle N \rangle$ -protocol-control-information of an $\langle N \rangle$ -protocol-data-unit. Within a layer, $\langle N \rangle$ -protocol-control-information is added to an $\langle N \rangle$ -service-data-unit to form an $\langle N \rangle$ -protocol-data-unit when no segmenting or blocking is performed [see Fig. 10 (a)]. If segmenting is performed, an $\langle N \rangle$ -service-data-unit is mapped into several $\langle N \rangle$ -protocol-data-units with added $\langle N \rangle$ -protocol-control-information [see Fig. 10 (b)].

Conversely, it may be necessary to perform blocking whereby several $\langle N \rangle$ -service-data-units with added $\langle N \rangle$ -protocol-control-information form an $\langle N \rangle$ -protocol-data-unit [see Fig. 10 (c)].

The Reference Model also permits concatenation whereby several $\langle N \rangle$ -protocol-data-units are concatenated into a single $\langle N - 1 \rangle$ -service-data-unit [see Fig. 10 (d)].

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

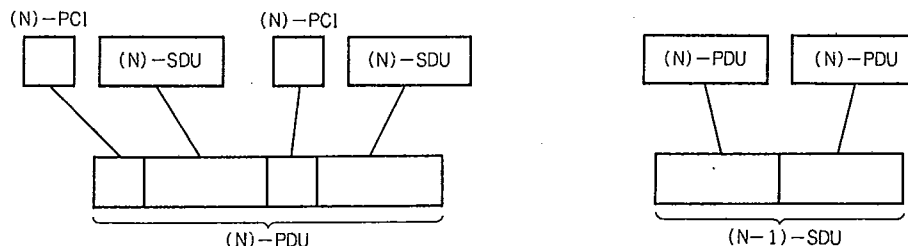
Fig. 10. Relationship between $\langle N \rangle$ -service-data-unit, $\langle N \rangle$ -protocol-data-unit and $\langle N - 1 \rangle$ -service-data-unit within a Layer

(a) Neither Segmenting nor Blocking (b) Segmenting/Reassembling



(c) Blocking/Deblocking

(d) Concatenation/Separation



Remarks 1. SDU = service-data-unit
PCI = protocol-control-information
PDU = protocol-data-unit

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

3. In the case of concatenation, $\langle N \rangle$ -protocol-data-unit does not necessarily include an $\langle N \rangle$ -service-data-unit.

5.7.6.6 Sequencing The <N - 1>-services provided by the <N - 1>-layer of the OSI architecture may not guarantee delivery of data in the same order as it was submitted by the <N>-layer. If the <N>-layer needs to preserve the order of data transferred through the <N - 1>-layer, sequencing mechanisms shall 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.

An acknowledgement function may require that information be included in the <N>-protocol-information 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.

Remark: 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.

Remark: 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

- (1) application-management Functions in the application layer (see 6.1) related to the management of OSI application-processes.
- (2) application-management-application-entity An application-entity which executes application-management functions.
- (3) OSI resources Data processing and data communication resources which are of concern to OSI.
- (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) systems-management-application-entity An application-entity which executes systems-management functions.
- (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 Concepts of OSI Management 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 in clause 7. of this Standard.

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 clause 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 Standard 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:

- (1) application-management;
- (2) systems-management; and

(3) 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:

- (1) initialization of parameters representing application-processes;
- (2) initiation, maintenance and termination of application-processes;
- (3) allocation and de-allocation of OSI resources to application-processes;
- (4) detection and prevention of OSI resource interference and deadlock;
- (5) integrity and commitment control;
- (6) security control; and
- (7) 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:

- (1) activation/deactivation management which includes:
 - (a) activation, maintenance and termination of OSI resources distributed in open systems, including physical media for OSI;
 - (b) some program loading functions;
 - (c) establishment/maintenance/release of connections between management entities; and
 - (d) open systems parameter initialization/modification;
- (2) monitoring which includes:
 - (a) reporting status or status changes; and
 - (b) reporting statistics; and
- (3) error control which includes:
 - (a) error detection and some of the diagnostic functions; and
 - (b) reconfiguration and restart.

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