IEC 61158-3-1

Edition 1.0 2007-12

INTERNATIONAL STANDARD

Industrial communication networks – Fieldbus specifications – Part 3-1: Data-link layer service definition – Type 1 elements

IEC 61158-3-1:2007(E)

IEC 61158-3-1

Edition 1.0 2007-12

INTERNATIONAL STANDARD

Industrial communication networks – Fieldbus specifications – Part 3-1: Data-link layer service definition – Type 1 elements

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE XF

ICS 35.100.20; 25.040.40

ISBN 2-8318-9410-7

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

____

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 3-1: Data-link layer service definition – Type 1 elements

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

NOTE Use of some of the associated protocol types is restricted by their intellectual-property-right holders. In all cases, the commitment to limited release of intellectual-property-rights made by the holders of those rights permits a particular data-link layer protocol type to be used with physical layer and application layer protocols in type combinations as specified explicitly in the IEC 61784 series. Use of the various protocol types in other combinations may require permission of their respective intellectual-property-right holders.

International Standard IEC 61158-3-1 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

This first edition and its companion parts of the IEC 61158-3 subseries cancel and replace IEC 61158-3:2003. This edition of this part constitutes an editorial revision.

This edition includes the following significant changes from the previous edition:

a) deletion of the former Type 6 fieldbus, and the placeholder for a Type 5 fieldbus data-link layer, for lack of market relevance;
b) addition of new types of fieldbuses;
c) division of this part into multiple parts numbered 3-1, 3-2, …, 3-19.

The text of this standard is based on the following documents:

FDIS	Report on voting
65C/473/FDIS	65C/484/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with ISO/IEC Directives, Part 2.

NOTE Slight variances from the directives have been allowed by the IEC Central Office to provide continuity of subclause numbering with prior editions.

The committee has decided that the contents of this publication will remain unchanged until the maintenance result date indicated on the IEC web site under http://webstore.iec.ch in the data related to the specific publication. At this date, the publication will be:

- reconfirmed;
- withdrawn;
- replaced by a revised edition, or
- amended.

NOTE The revision of this standard will be synchronized with the other parts of the IEC 61158 series.

The list of all the parts of the IEC 61158 series, under the general title *Industrial communication networks – Fieldbus specifications*, can be found on the IEC web site.

0 Introduction

0.1 General

This part of IEC 61158 is one of a series produced to facilitate the interconnection of automation system components. It is related to other standards in the set as defined by the "three-layer" fieldbus reference model described in IEC/TR 61158-1.

Throughout the set of fieldbus standards, the term "service" refers to the abstract capability provided by one layer of the OSI Basic Reference Model to the layer immediately above. Thus, the data-link layer service defined in this standard is a conceptual architectural service, independent of administrative and implementation divisions.

0.2 Nomenclature for references within this standard

Clauses, including annexes, can be referenced in their entirety, including any subordinate subclauses, as "Clause N" or "Annex N", where N is the number of the clause or letter of the annex.

Subclauses can be referenced in their entirety, including any subordinate subclauses, as "N.M" or "N.M.P" and so forth, depending on the level of the subclause, where N is the number of the subclause or letter of the annex, and M, P and so forth represent the successive levels of subclause up to and including the subclause of interest.

When a clause or subclause contains one or more subordinate subclauses, the text between the clause or subclause heading and its first subordinate subclause can be referenced in its entirety as "N.0" or "N.M.0" or "N.M.P.0" and so forth, where N, M and P are as above. Stated differently, a reference ending with ".0" designates the text and figures between a clause or subclause header and its first subordinate subclause.

NOTE This nomenclature provides a means of referencing text in hanging clauses. Such clauses existed in earlier editions of IEC 61158-3, Type 1 clauses. Those hanging clauses are maintained in this edition to minimize the disruption to existing national and multi-national standards and consortia documents which reference that prior subclause numbering.

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 3-1: Data-link layer service definition – Type 1 elements

1 Scope

1.1 Overview

This part of IEC 61158 provides common elements for basic time-critical messaging communications between devices in an automation environment. The term "time-critical" is used to represent the presence of a time-window, within which one or more specified actions are required to be completed with some defined level of certainty. Failure to complete specified actions within the time window risks failure of the applications requesting the actions, with attendant risk to equipment, plant and possibly human life.

This standard defines in an abstract way the externally visible service provided by the Type 1 fieldbus data-link layer in terms of

a) the primitive actions and events of the service;

b) the parameters associated with each primitive action and event, and the form which they take; and

c) the interrelationship between these actions and events, and their valid sequences.

The purpose of this standard is to define the services provided to

- the Type 1 fieldbus application layer at the boundary between the application and data-link layers of the fieldbus reference model;
- systems management at the boundary between the data-link layer and systems management of the fieldbus reference model.

1.2 Specifications

The principal objective of this standard is to specify the characteristics of conceptual data-link layer services suitable for time-critical communications, and thus supplement the OSI Basic Reference Model in guiding the development of data-link protocols for time-critical communications. A secondary objective is to provide migration paths from previously existing industrial communications protocols.

This specification may be used as the basis for formal DL-Programming-Interfaces. Nevertheless, it is not a formal programming interface, and any such interface will need to address implementation issues not covered by this specification, including

a) the sizes and octet ordering of various multi-octet service parameters;

b) the correlation of paired request and confirm, or indication and response, primitives.

1.3 Conformance

This standard does not specify individual implementations or products, nor does it constrain the implementations of data-link entities within industrial automation systems.

There is no conformance of equipment to this data-link layer service definition standard. Instead, conformance is achieved through implementation of the corresponding data-link protocol that fulfills the Type 7 data-link layer services defined in this standard.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 7498-1, *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model*

ISO/IEC 7498-3, *Information technology – Open Systems Interconnection – Basic Reference Model: Naming and addressing*

ISO/IEC 10731:1994, *Information technology – Open Systems Interconnection – Conventions for the definition of OSI services*

3 Terms, definitions, symbols, abbreviations and conventions

For the purposes of this document, the following terms, definitions, symbols, abbreviations and conventions apply.

3.1 Reference model terms and definitions

This standard is based in part on the concepts developed in ISO/IEC 7498-1 and ISO/IEC 7498-3, and makes use of the following terms defined therein.

3.1.1 DL-address	[7498-3]
3.1.2 DL-address-mapping	[7498-1]
3.1.3 called-DL-address	[7498-3]
3.1.4 calling-DL-address	[7498-3]
3.1.5 centralized multi-end-point-connection	[7498-1]
3.1.6 DL-connection	[7498-1]
3.1.7 DL-connection-end-point	[7498-1]
3.1.8 DL-connection-end-point-identifier	[7498-1]
3.1.9 DL-connection-mode transmission	[7498-1]
3.1.10 DL-connectionless-mode transmission	[7498-1]
3.1.11 correspondent (N)-entities **correspondent DL-entities (N=2)** **correspondent Ph-entities (N=1)**	[7498-1]
3.1.12 DL-duplex-transmission	[7498-1]
3.1.13 (N)-entity **DL-entity (N=2)** **Ph-entity (N=1)**	[7498-1]
3.1.14 DL-facility	[7498-1]
3.1.15 flow control	[7498-1]

3.1.16	**(N)-layer** **DL-layer (N=2)** **Ph-layer (N=1)**	[7498-1]
3.1.17	**layer-management**	[7498-1]
3.1.18	**DL-local-view**	[7498-3]
3.1.19	**DL-name**	[7498-3]
3.1.20	**naming-(addressing)-domain**	[7498-3]
3.1.21	**peer-entities**	[7498-1]
3.1.22	**primitive name**	[7498-3]
3.1.23	**DL-protocol**	[7498-1]
3.1.24	**DL-protocol-connection-identifier**	[7498-1]
3.1.25	**DL-protocol-data-unit**	[7498-1]
3.1.26	**DL-relay**	[7498-1]
3.1.27	**reset**	[7498-1]
3.1.28	**responding-DL-address**	[7498-3]
3.1.29	**routing**	[7498-1]
3.1.30	**segmenting**	[7498-1]
3.1.31	**(N)-service** **DL-service (N=2)** **Ph-service (N=1)**	[7498-1]
3.1.32	**(N)-service-access-point** **DL-service-access-point (N=2)** **Ph-service-access-point (N=1)**	[7498-1]
3.1.33	**DL-service-access-point-address**	[7498-3]
3.1.34	**DL-service-connection-identifier**	[7498-1]
3.1.35	**DL-service-data-unit**	[7498-1]
3.1.36	**DL-simplex-transmission**	[7498-1]
3.1.37	**DL-subsystem**	[7498-1]
3.1.38	**systems-management**	[7498-1]
3.1.39	**DLS-user-data**	[7498-1]

3.2 Service convention terms and definitions

This standard also makes use of the following terms defined in ISO/IEC 10731 as they apply to the data-link layer:

3.2.1 acceptor

3.2.2 asymmetrical service

3.2.3 confirm (primitive);
requestor.deliver (primitive)

3.2.4 deliver (primitive)

3.2.5 DL-confirmed-facility

3.2.6 DL-facility

3.2.7 DL-local-view

3.2.8 DL-mandatory-facility

3.2.9 DL-non-confirmed-facility

3.2.10 DL-provider-initiated-facility

3.2.11 DL-provider-optional-facility

3.2.12 DL-service-primitive;
primitive

3.2.13 DL-service-provider

3.2.14 DL-service-user

3.2.15 DLS-user-optional-facility

3.2.16 indication (primitive);
acceptor.deliver (primitive)

3.2.17 multi-peer

3.2.18 request (primitive);
requestor.submit (primitive)

3.2.19 requestor

3.2.20 response (primitive);
acceptor.submit (primitive)

3.2.21 submit (primitive)

3.2.22 symmetrical service

3.3 Data-link service terms and definitions

3.3.1
bridge, DL-router
DL-relay entity which performs selective store-and-forward and routing functions

a) to connect two or more separate DL-subnetworks (links) to form a unified DL-subnetwork (the extended link), and

b) to provide a means by which two end systems can communicate, when at least one of the end systems is periodically inattentive to the interconnecting DL-subnetwork;

and also provides time synchronization among the links to which it is forwarding

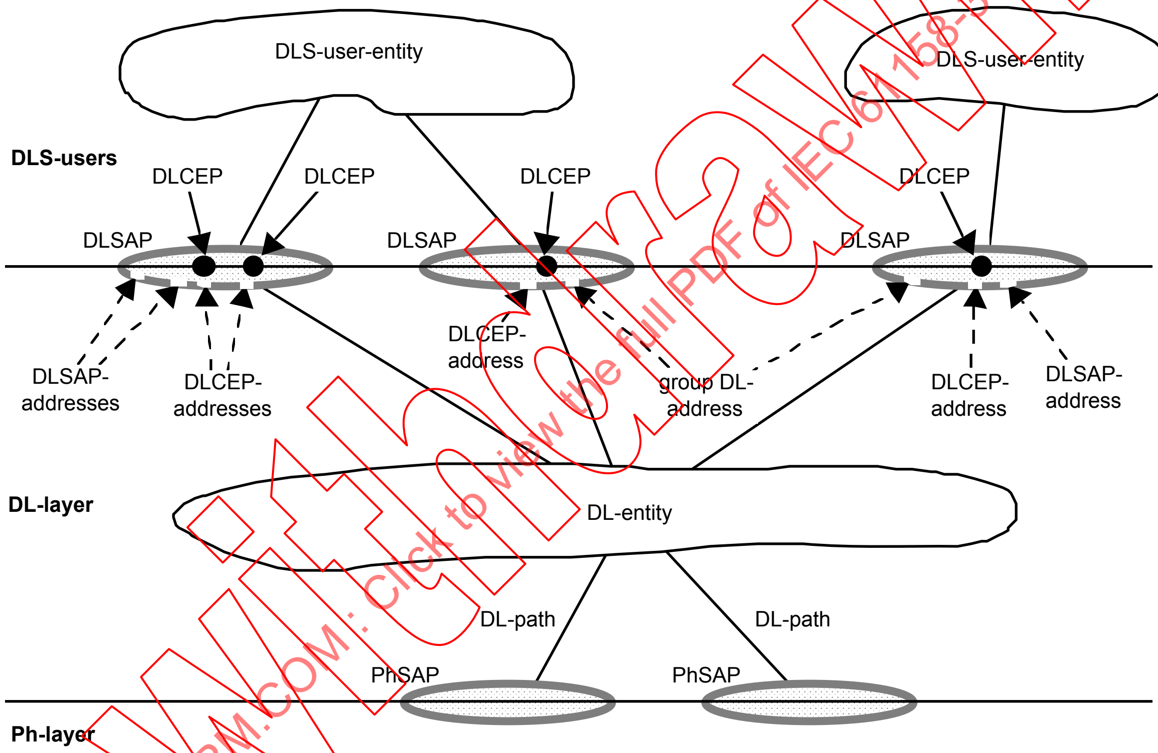
3.3.2
DLCEP-address
DL-address which designates either

a) one peer DL-connection-end-point; or

b) one multi-peer publisher DL-connection-end-point, and implicitly the corresponding set of subscriber DL-connection-end-points

where each DL-connection-end-point exists within a distinct DLSAP and is associated with a corresponding distinct DLSAP-address

NOTE This is an extension of the use of DL-addresses beyond that specified in ISO/IEC 7498-3 (see Figure 1).



NOTE 1 DLSAPs and PhSAPs are depicted as ovals spanning the boundary between two adjacent layers.

NOTE 2 DL-addresses are depicted as designating small gaps (points of access) in the DLL portion of a DLSAP. A DLCEP-address also designates a specific point of information flow (its DLCEP) within the DLSAP.

NOTE 3 A single DL-entity may have multiple DLSAP-addresses and group DL-addresses associated with a single DLSAP.

NOTE 4 Figure 1 also shows the relationships of DL-paths and PhSAPs.

Figure 1 – Relationships of DLSAPs, DLSAP-addresses, DLCEPs, DLCEP-addresses, DLSEP-addresses and group DL-addresses

3.3.3
DL-segment, link, local link
single DL-subnetwork in which any of the connected DLEs may communicate directly, without any intervening DL-relaying, whenever all of those DLEs that are participating in an instance of communication are simultaneously attentive to the DL-subnetwork during the period(s) of attempted communication

3.3.4
DLSAP
distinctive point at which DL-services are provided by a single DL-entity to a single higher-layer entity

NOTE This definition, derived from ISO/IEC 7498-1, is repeated here to facilitate understanding of the critical distinction between DLSAPs and their DL-addresses.

3.3.5
DL(SAP)-address
either an individual DLSAP-address, designating a single DLSAP of a single DLS-user, or a group DL-address potentially designating multiple DLSAPs, each of a single DLS-user

NOTE This terminology is chosen because ISO/IEC 7498-3 does not permit the use of the term DLSAP-address to designate more than a single DLSAP at a single DLS-user.

3.3.6
(individual) DLSAP-address
DL-address that designates only one DLSAP within the extended link

NOTE A single DL-entity may have multiple DLSAP-addresses associated with a single DLSAP.

3.3.7
DLSEP-address
DL-address which designates a DL-scheduling-end-point within a DLE

NOTE This is an extension of the use of DL-addresses beyond that specified in ISO/IEC 7498-3. (See Figure 1.)

3.3.8
extended link
DL-subnetwork, consisting of the maximal set of links interconnected by DL-relays, sharing a single DL-name (DL-address) space, in which any of the connected DL-entities may communicate, one with another, either directly or with the assistance of one or more of those intervening DL-relay entities

NOTE An extended link may be composed of just a single link.

3.3.9
frame
denigrated synonym for DLPDU

3.3.10
group DL-address
DL-address that potentially designates more than one DLSAP within the extended link. A single DL-entity may have multiple group DL-addresses associated with a single DLSAP. A single DL-entity also may have a single group DL-address associated with more than one DLSAP

3.3.11
initiator
DLE role in which a DLE sends a DLPDU to a peer responder DLE, which immediately sends a reply DLPDU back to the initiator DLE (and potentially to other DLEs) as part of the same transaction.

NOTE Some prior national standards have referred to this role as a “master” role.

3.3.12
multi-peer DLC
centralized multi-end-point DL-connection offering DL-duplex-transmission between a single distinguished DLS-user, known as the publisher or publishing DLS-user, and a set of peer but undistinguished DLS-users, known collectively as the subscribers or subscribing DLS-users, where the publishing DLS-user can send to the subscribing DLS-users as a group (but not individually), and the subscribing DLS-users can send to the publishing DLS-user (but not to each other)

NOTE 1 A multi-peer DLC always provides asymmetrical service. It may also be negotiated to provide only DL-simplex service, either from the publisher to the subscribers, or from the subscribers to the publisher. In this last case, the characterizations as publisher and subscriber are misnomers.

NOTE 2 The publishing DLS-user may need to employ control of its publishing rate, because a subscribing DLS-user cannot exert either flow or rate control on its publishing peer entity. Similar considerations apply to subscribing DLS-users with respect to their sending DLSDUs to the publishing DLS-user.

3.3.13
node
single DL-entity as it appears on one local link

3.3.14
peer DLC
point-to-point DL-connection offering DL-duplex-transmission between two peer DLS-users where each can be a sending DLS-user, and each as a receiving DLS-user may be able to exert flow control on its sending peer

NOTE A peer DLC is negotiated to provide either symmetrical service or asymmetrical service. A peer DLC may also be negotiated to provide only DL-simplex service.

3.3.15
receiving DLS-user
DL-service user that acts as a recipient of DLS-user-data

NOTE A DL-service user can be concurrently both a sending and receiving DLS-user.

3.3.16
responder
DLE role in which a DLE sends a DLPDU as an immediate reply to a DLPDU received from a peer initiator DLE, all as part of a single transaction

NOTE Some prior national standards have referred to this role as a “slave” role.

3.3.17
sending DLS-user
DL-service user that acts as a source of DLS-user-data

3.3.18
timeliness, DL-timeliness
attribute of a datum which provides an assessment of the temporal currency of that datum, of particular importance in sampled-data systems, which may need to make decisions based on the timeliness, or lack of timeliness, of current data samples

NOTE 1 As a general rule, timeliness is a user attribute which can be affected negatively by the various layers of the data transport system. That is, a datum which was timely when the requesting user presented it to a data communications subsystem for transmission may become untimely due to delays in the communications subsystem.

NOTE 2 DL-timeliness is an attribute of a DLS-user datum relating the timing of a DLS-user/DLE interaction which writes or reads that datum to one or more other (earlier) DLS-user/DLE interactions.

NOTE 3 These concepts also support migration from previous national standards.

3.3.19
transaction
single DLPDU, or a sequence of two immediately consecutive related DLPDUs, resulting from a single DLS-user request

NOTE 1 The DLE sending the first DLPDU of the transaction is known as the initiator; the DLE which sends the second DLPDU of the transaction, if any, is known as the responder.

NOTE 2 A DL-entity can be both an initiator and a responder in the same transaction.

3.4 Common symbols and abbreviations

NOTE Many symbols and abbreviations are common to more than one protocol Type; they are not necessarily used by all protocol Types.

3.4.1 DL-	Data-link layer (as a prefix)
3.4.2 DLC	DL-connection
3.4.3 DLCEP	DL-connection-end-point
3.4.4 DLE	DL-entity (the local active instance of the data-link layer)
3.4.5 DLL	DL-layer
3.4.6 DLPCI	DL-protocol-control-information
3.4.7 DLPDU	DL-protocol-data-unit
3.4.8 DLM	DL-management
3.4.9 DLME	DL-management Entity (the local active instance of DL-management)
3.4.10 DLMS	DL-management service
3.4.11 DLS	DL-service
3.4.12 DLSAP	DL-service-access-point
3.4.13 DLSEP	DL-schedule-end-point
3.4.14 DLSDU	DL-service-data-unit
3.4.15 FIFO	First-in first-out (queuing method)
3.4.16 OSI	Open systems interconnection
3.4.17 Ph-	Physical layer (as a prefix)
3.4.18 PhE	Ph-entity (the local active instance of the Physical layer)
3.4.19 PhL	Ph-layer
3.4.20 QoS	Quality-of-service

3.5 Common conventions

3.5.1 Basic conventions

This standard uses the descriptive conventions given in ISO/IEC 10731.

The service model, service primitives, and time-sequence diagrams used are entirely abstract descriptions; they do not represent a specification for implementation.

Service primitives, used to represent service user/service provider interactions (see ISO/IEC 10731), convey parameters that indicate information available in the user/provider interaction.

This standard uses a tabular format to describe the component parameters of the DLS primitives. The parameters that apply to each group of DLS primitives are set out in tables throughout the remainder of this standard. Each table consists of up to six columns, containing the name of the service parameter, and a column each for those primitives and parameter-transfer directions used by the DLS:

— the request primitive's input parameters;

— the request primitive's output parameters;

— the indication primitive's output parameters;

— the response primitive's input parameters; and

— the confirm primitive's output parameters.

NOTE The request, indication, response and confirm primitives are also known as requestor.submit, acceptor.deliver, acceptor.submit, and requestor.deliver primitives, respectively (see ISO/IEC 10731).

One parameter (or part of it) is listed in each row of each table. Under the appropriate service primitive columns, a code is used to specify the type of usage of the parameter on the primitive and parameter direction specified in the column:

M	—	parameter is mandatory for the primitive;
U	—	parameter is a user option and may or may not be provided depending on the dynamic usage of the DLS-user. When not provided, a default value for the parameter is assumed;
C	—	parameter is conditional upon other parameters or upon the environment of the DLS-user;
CU	—	parameter is a conditional user option, and may or may not be permitted depending upon other parameters or upon the environment of the DLS-user. When permitted, it may or may not be provided depending on the dynamic usage of the DLS-user. When permitted and not provided, a default value for the parameter is assumed.
(blank)	—	parameter is never present.

Some entries are further qualified by items in brackets. These may be

a) a parameter-specific constraint

(**=**) indicates that the parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table;

(≤) indicates that the set of parameter values has an implicit order and that the parameter's value is less than or equal to that of the parameter in the service primitive to its immediate left in the table (that is, left by one or two columns);

(≥) indicates that the set of parameter values has an implicit order and that the parameter's value is greater than, or equal to, that of the parameter in the service primitive to its immediate left in the table (that is, left by one or two columns).

b) an indication that some note applies to the entry

(**n**) indicates that the following note n contains additional information pertaining to the parameter and its use.

In any particular interface, not all parameters need be explicitly stated. Some may be implicitly associated with the DLSAP at which the primitive is issued.

In the diagrams which illustrate these interfaces, dashed lines indicate cause-and-effect or time-sequence relationships, and wavy lines indicate that events are roughly contemporaneous.

3.5.2 Identifiers

Many of the DLS primitives specify one or more identifier parameters that are drawn from either a local DL-identifier space or a local DLS-user-identifier space. The existence and use of such identifiers in an implementation of the services specified in this standard is a purely local issue. Nevertheless, these identifiers are specified explicitly in these primitives to provide a descriptive means

a) of cancelling (aborting) an outstanding request primitive before receiving its corresponding confirm primitive;
b) for referring within a request or response primitive to persistent DL-objects, such as buffers and queues, which were created as the result of a previous DLS primitive; and
c) for referring within an indication or confirm primitive to persistent DL-objects which were created as the result of a previous DLS primitive.

Adherence to the OSI principle of architectural layering necessitates the presumption of distinct non-intersecting identifier spaces for the DLS-provider and each separate DLS-user, because they may have non-overlapping local views. Consequently, DLS-user identifiers are required for a) and b); while DL-identifiers are required for c).

4 Overview of the data-link layer service

4.1 General

The DLS provides for the transparent and reliable transfer of data between DLS-users. It makes the way that supporting communications resources are utilized invisible to these DLS-users.

In particular, the DLS provides for the following:

a) Independence from the underlying physical layer. The DLS relieves DLS-users from all direct concerns regarding which configuration is available (for example, direct connection, or indirect through one or more bridges) and which physical facilities are used (for example, which of a set of diverse physical paths).
b) Transparency of transferred information. The DLS provides for the transparent transfer of DLS-user-data. It does not restrict the content, format or coding of the information, nor does it ever need to interpret the structure or meaning of that information. It may, however, restrict the amount of information that can be transferred as an indivisible unit.

   NOTE A DLS-user may segment arbitrary-length data into limited-length DLSDUs before making DLS requests, and may reassemble received DLSDUs into those larger data units.

c) Reliable transfer of data. The DLS relieves the DLS-user from concerns regarding insertion or corruption of data, or, if requested, loss, duplication or misordering of data, which may occur. In some cases of unrecovered errors in the data-link Layer, duplication or loss of DLSDUs may occur. In cases where protection against misordering of data is not employed, misordering can occur.

   NOTE Detection of duplicate, lost or misordered DLSDUs may be performed by DLS-users.

d) Quality-of-service (QoS) selection. The DLS provides DLS-users with a means to request and to agree upon a quality of service for the transfer of data. QoS is specified by means of QoS parameters representing aspects such as mode of operation, transit delay, accuracy and reliability.
e) Addressing. The DLS allows the DLS-user to identify itself and to specify the DLSAPs between which a DLC is to be established. DL-addresses have only regional significance within a specific DL-segment. Extended DL-addresses have only regional significance within a specific DL-subsystem over a set of bridged DL-segments. Therefore, it is not appropriate to define a global addressing structure.

   NOTE The DLS is required to differentiate between the individual systems that are physically or logically connected to a multipoint data-link and to differentiate between connections. For commonality with other service definitions, this mechanism is referred to as addressing and the objects used to differentiate between systems are referred to as addresses. In a formal sense, this is an extension of the use of addresses beyond that specified in ISO/IEC 7498-3.

f) Scheduling. The DLS allows the set of DLS-users to provide some guidance on the internal scheduling of the distributed DLS-provider. This guidance supports the time-critical aspects of the DLS, by permitting the DLS-user some degree of management over when opportunities for communication will be granted to various DLEs for various DLSAP-addresses and DLCEPs.

g) Common time sense. The DLS can provide the DLS-user with a sense of time that is common to all DLS-users on the extended link.

h) Queues and buffers. The DLS can provide the sending or receiving DLS-user with either a FIFO queue or a retentive buffer or a non-retentive buffer (that is, one which becomes empty after being read), where each queue item or buffer can hold a single DLSDU.

4.1.1 Overview of DL-subnetwork structuring

A DL-subnetwork consists of a set of DL-segments (links) interconnected by DL-relay entities (bridges) which provide DL-layer-internal synchronization, coordination and routing services to the set of interconnected DLEs.

A DL-segment consists of a set of DLEs, all of which are connected directly (that is, without intervening DL-relay entities) to a single shared logical communications channel, known as a *link*.

A link (logical communications channel) consists of one or more physically independent, logically parallel, cooperatively scheduled, real communications channels, which are known as *paths*.

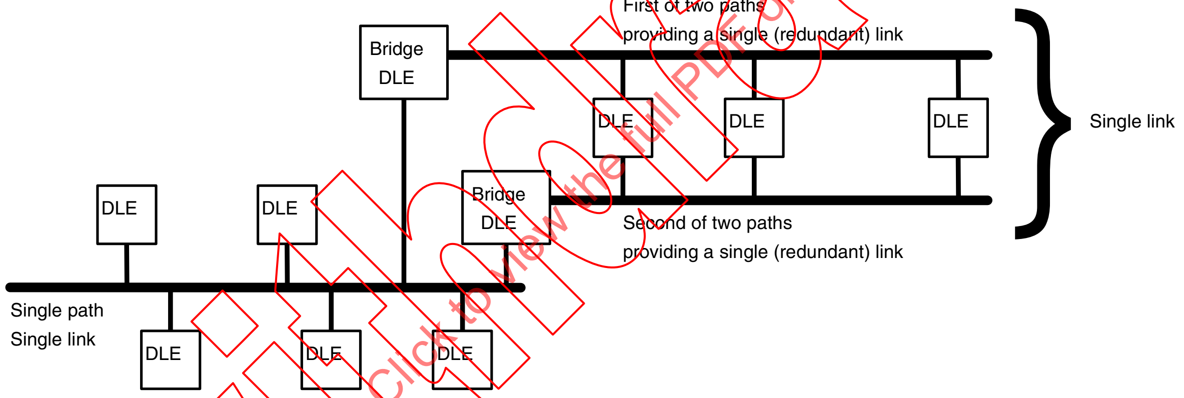


Figure 2 – Example of paths, links, bridges, and the extended link

A single shared PhL-provider enables communications among the DLEs on a given path. A link is made up of conceptually parallel paths. An example is shown in Figure 2.

NOTE 1 A link consisting of more than one path is an instance of DL-redundancy. This is distinct from Ph-redundancy, which is necessarily hidden from the DLL and all DLEs due to the principles of layering (ISO/IEC 7498-1).

NOTE 2 In a logical sense, DLEs are connected to links and bridges interconnect links. Yet in a physical sense, DLEs are connected to paths and bridges interconnect paths. DL-communication-services are independent of the specific path employed, and the DLS-user has no cognizance of any path multiplicity.

4.1.2 Overview of DL-naming (addressing)

DL-names, known conventionally as DL-addresses, are identifiers from a defined identifier space — the DL-address-space — which serve to name objects within the scope of the data-link layer. Examples of such objects are data-link layer-service-access-points (DLSAPs), data-link-connection-end-points (DLCEPs), and data-link-entities (DLEs).

The DL-address-space from which DL-addresses are drawn may be partitioned into subspaces of DL-addresses:

a) to cluster addresses of the same generic function, such as

   1) DLSAP-addresses naming specific DLSAPs;

2) DL-addresses naming groups of DLSAPs;

3) DL-addresses designating one or more DLCEPs; and

4) DL-addresses designating DLEs;

b) to cluster addresses for administrative purposes, such as addresses that are

1) known to be local to, and allocable by, a single DLE;

2) known to be local to a single link (DL-segment) but not to have a known specific DLE-locality; or

3) known to have no implicit DL-locality;

c) to cluster addresses for routing purposes, such as addresses that are known to be local to a single DL-segment or a single DLE.

4.1.2.1 Functional partitioning of the DL-address-space

The space of DL-addresses may be partitioned functionally as follows:

a) DLE-specific DLSAP-addresses;

b) other DLSAP-addresses;

NOTE These addresses can designate at most one DLSAP within a single DLE within the entire set of DL-interconnected DLEs.

c) group DL-addresses designating a group of DLSAPs;

NOTE These addresses are sometimes (incorrectly) referred to as group-DLSAP-addresses.

d) DLE-specific DLCEP-addresses;

e) other DLCEP-addresses;

f) specific aspects of a specific DLE; or

g) specific aspects of a group of DLEs.

4.1.2.2 Administrative partitioning of the DL-address-space

The space of DL-addresses may be partitioned administratively as follows:

a) DLE-specific DL-addresses, which are known to refer to objects within the scope of a specific DLE, and which are allocable by that DLE;

b) DL-segment (link) specific but DLE-independent DL-addresses, which are known to refer to objects within the scope of a specific DL-segment, and which are allocable locally by the DL-address-space administrator for that DL-segment; or

c) DL-segment-independent DL-addresses, which are known to refer to objects within the DL-connected set of DL-segments, and which are allocable by the DL-address-space administrator for the connected set of DL-segments.

NOTE A DL-address-space administrator can always allocate a set of addresses to a subordinate administrator for its sub-administration. For example, the DL-administrator for the entire set of DL-connected DL-segments can allocate a contiguous block of unassigned DL-addresses to the DL-administrator for a specific DL-segment, or to the local administrator within a single DLE.

4.1.2.3 Routing-related partitioning of the DL-address-space

The space of DL-addresses may be partitioned to assist DL-routing activities as follows:

a) DLE-specific DL-addresses;

b) DL-segment (link) specific but DLE-independent DL-addresses; or

c) DL-segment-independent DL-addresses.

4.2 Types and classes of data-link layer service

There are four types of DLS:

a) a DL(SAP)-address, queue and buffer management service (defined in Clause 5);

b) a connection-mode data transfer service with four classes of service (defined in Clause 6);

c) a connectionless-mode data transfer service with three classes of service (defined in Clause 7); and

d) a time and transaction scheduling service (defined in Clause 8).

All four types of DLS are always provided; the DLS-user may choose those most appropriate for use. Within the DLS types, the DLS-user is limited to those classes of service supported by the selected DL-protocol implementation.

NOTE Classes of service are defined in detail in the clause which describes a specific type of DLS.

A DL-management service (DLMS) is defined in Clause 9.

4.3 Quality-of-service (QoS) attributes common to multiple types of data-link layer service

A DLS-user may select, directly or indirectly, many aspects of the various data-link services. The term quality-of-service (QoS) refers to those aspects that are under the direct control of the DLS-provider. QoS can only be properly determined when DLS-user behavior does not constrain or impede the performance of the DLS.

Static QoS attributes are selected once for an entire type of DLS. *Dynamic* QoS attributes are selected independently at each DLS invocation. *Semi-static* QoS attributes are static attributes for one type of DLS, and serve as defaults for corresponding dynamic attributes in another type of DLS.

Most QoS attributes have default values which can be set by DL-management and then overridden on a per-DLSAP-address basis by the DLS-user.

NOTE The existence of multiple levels of default QoS attribute values and of means of setting those default values can simplify use of the DLS. Some implementations of this DLS may provide additional levels of default QoS beyond those specified in this standard.

Four QoS attributes of the DL-data transfer services apply conceptually to both connection-mode and connectionless operation. The DLS-user may specify values for these attributes when binding a DLSAP-address to the DLS-user's DLSAP; any unspecified attributes will assume the default values set by DL-management:

a) Two of these four attributes are considered dynamic; their DLSAP-address-related values serve merely as defaults for each appropriate DLS invocation and can be overridden on an instance by instance basis.

b) The third and fourth attributes are semi-static; they are static for connectionless DLS, but dynamic for all DLCEP-establishment requests and responses, where its value serves merely as a default for each appropriate DLS invocation and can be overridden on an instance by instance basis.

A fifth attribute applies only to connection-mode operation and is dynamic for each DLCEP.

4.3.1 DLL priority (dynamic QoS attribute)

All DLCEP establishment requests and responses, all connectionless data transfer requests, and many DL-scheduling requests, specify an associated DLL priority used in scheduling DLL data transfer services. This DLL priority also determines the maximum amount of DLS-user-data that can be conveyed in a single DLPDU. This maximum is determined by the DL-protocol specification.

The DL-protocol should support three DLL priority levels, each of which should be capable of conveying a specified amount of DLS-user data per appropriate DLPDU. The three DLL priorities with their corresponding ranges of conveyable DLS-user-data (per DLPDU) are, from highest priority to lowest priority:

a) *URGENT* — capability of conveying up to 64 DLS-user octets per DLPDU;

b) *NORMAL* — capability of conveying up to 128 DLS-user octets per DLPDU; and

c) *TIME-AVAILABLE* — capability of conveying up to 256 DLS-user octets per DLPDU.

NOTE 1 URGENT and NORMAL are considered *time-critical* priority levels; TIME-AVAILABLE is considered a *non-time-critical* priority level.

NOTE 2 DLCEP establishment may negotiate URGENT to NORMAL or TIME-AVAILABLE, or NORMAL to TIME-AVAILABLE.

The default QoS value can be set by DL-management; when not so set its value is TIME-AVAILABLE.

4.3.2 DLL maximum confirm delay (dynamic QoS attribute)

Each DLCEP establishment request, and each response, specifies upper bounds on the maximum time duration permitted for the completion of each related instance of a sequence of connection-oriented DLS primitives:

a) the common maximum time for completion of
   - a related sequence of DL-CONNECT primitives;
   - a related sequence of DL-RESET primitives;
   - a related sequence of DL-SUBSCRIBER-QUERY primitives; and

b) the maximum time for completion of a related sequence of DL-DATA primitives.

Each connectionless service request specifies an upper bound on the maximum time duration permitted for the completion of each related instance of a sequence of connectionless DLS primitives:

c) the maximum time for completion of a related sequence of locally-confirmed DL-UNITDATA primitives; and

d) the common maximum time for completion of
   - a related sequence of remotely-confirmed DL-UNITDATA primitives;
   - a related sequence of DL-LISTENER-QUERY primitives; or
   - an instance of the DL-UNITDATA-EXCHANGE service.

Each parameter either has the value *UNLIMITED* or specifies an upper bound, in units of 1 ms, from 1 ms to 60 s, inclusive. The value UNLIMITED provides compatibility with prior OSI protocols, and provides a means for DL-CONNECT requests to remain in a "listening" or "half-open" state. The completion status of "timeout" cannot occur on a DLS-user request which specifies UNLIMITED.

The parameters for the DL-DATA and locally-confirmed DL-UNITDATA primitives specify intervals less than or equal to that for the DL-CONNECT, DL-RESET, DL-SUBSCRIBER-QUERY, remotely-confirmed DL-UNITDATA, and DL-LISTENER-QUERY primitives.

The intervals specified are the maximum permissible delays

1) between the issuing of the specified request primitives and the issuing of the corresponding confirm primitives; and

2) between the initiation and completion of a single instance of the specified publishing or unitdata-exchange service.

NOTE For DLEs that do not support a time resolution of 1 ms, the requested time interval may be rounded up to the next-greatest multiple of that resolution that the DLE does support, or to approximately 60 s if the DLE has no sense of time.

The default QoS values can be set by DL-management; when not so set the value for each of these QoS parameters is UNLIMITED.

4.3.3 DLPDU authentication (semi-static QoS attribute)

Each DLCEP establishment negotiation, and each connectionless data transfer, uses this attribute to determine

a) a lower bound on the amount of DL-addressing information used in the DLPDUs that provide the associated DLL data transfer services;

   NOTE This has a slight impact on the residual rate of DLPDU misdelivery; more addressing information reduces the potential for misdelivery.

b) whether the current state of a sending peer or publisher DLCEP should be sent at low-frequency to the DLC's peer or subscriber DLCEP(s) even when there are no unconfirmed DLS-user requests outstanding at the sending DLCEP; and

   NOTE This continuing background transmission is known as *residual activity*.

c) whether all related scheduling actions should be executed locally.

NOTE These last two aspects are of particular importance in safety systems.

The three levels specifiable, with their amounts of DL-addressing information, are:

1) **ORDINARY** — each DLPDU should include the minimum permitted amount of addressing information;
2) **SOURCE** — each DLPDU should include a source DL-address where possible;
3) **MAXIMAL** — each DL-address should include the maximal amount of addressing information possible. Also, all related scheduling actions should be executed locally; and each sending peer or publisher DLCEP of the DLC should maintain a low-frequency report of state information when there is no DLS-user activity.

The default QoS value can be set by DL-management; when not so set its value is ORDINARY. DLCEP establishment may negotiate ORDINARY to SOURCE to MAXIMAL.

4.3.4 DL-scheduling-policy (semi-static QoS attribute)

This attribute is static for connectionless services, but is dynamic for connection-mode services.

For each DLSAP-address, and each DLCEP, the DLS-user can override the normal (implicitly-scheduled) DLL policy of providing the requested DLS as soon as possible, and instead can defer any inter-DLS-user communication required by a DL-DATA or DL-UNITDATA request DLS-primitive until that deferral is cancelled by an involved DLS-user. A DL-COMPEL-SERVICE request, specifying the affected DLSAP-address or DLCEP, permits the continued execution of just a single deferred in-process request or response DLS-primitive. Only DL-services that provide DLS-user intercommunication are affected by this attribute.

NOTE DLC support services such as DL-CONNECT, DL-RESET and DL-DISCONNECT, and intra-DLS-provider services such as DL-SUBSCRIBER-QUERY and DL-LISTENER-QUERY, are not affected by this attribute.

The two choices are:

a) **IMPLICIT** — any required communications with peer DLS-user(s) from this DLSAP-address, or from this DLCEP, will occur as soon as possible;

   NOTE The choice IMPLICIT is incompatible with a DLCEP which is bound as sender to a buffer, because writing to a buffer does not trigger transmission. Thus the only usable choice for a sending buffer is EXPLICIT.

b) **EXPLICIT** — any required data or unitdata communications with peer DLS-user(s) from this DLSAP-address, or from this DLCEP, will occur only when the deferral is explicitly cancelled by an involved DLS-user.

   NOTE Possible use of previously scheduled communications opportunities makes it possible for this deferral and subsequent release to result in earlier communications with the peer DLS-users than that provided by the IMPLICIT alternative.

The default QoS value cannot be set by DL-management; its value is always IMPLICIT.

4.3.5 DL-timeliness (dynamic DLCEP QoS attributes)

This attribute applies only to retentive DL-buffers, to DLCEPs at which DL-buffers are bound, and to those DLS-primitives which transfer DLS-user data to or from DL-buffers at such DLCEPs.

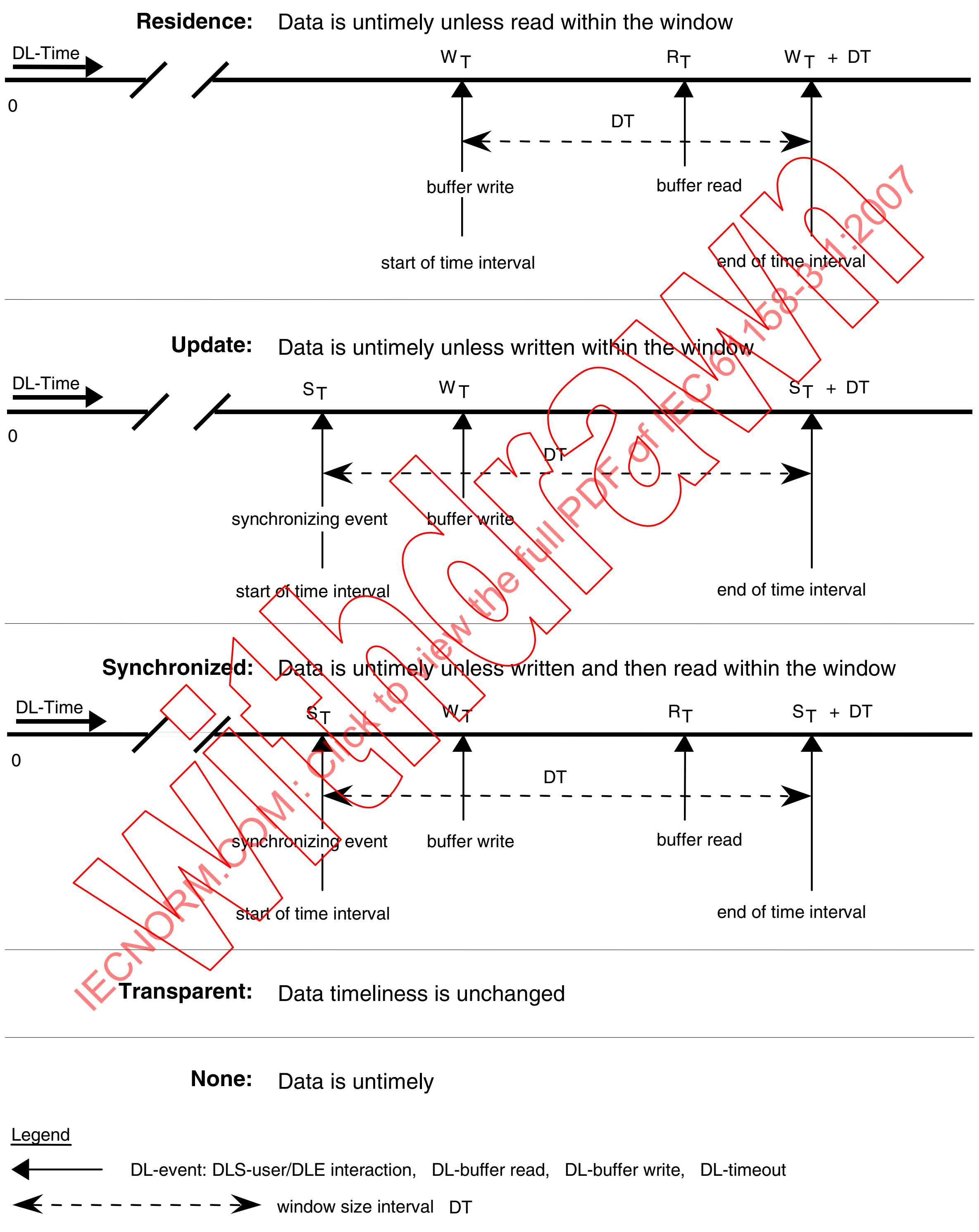


Figure 3 – Types of DL-timeliness
In terms of elapsed DL-time and events at the assessing DLCEP

Each DLCEP establishment request, and each response, can specify DL-timeliness criteria which are to apply to information sent from, or received into, retentive buffers at that DLCEP. Four types of DL-timeliness can be supported: RESIDENCE timeliness, UPDATE timeliness, SYNCHRONIZED timeliness, and TRANSPARENT timeliness. All four types of timeliness, and the case where there is no timeliness, are shown in Figure 3:

a) **RESIDENCE** timeliness is an assessment based upon the length of time that a DLS-user datum has been resident in a buffer, which is the time interval between

   1) the moment when the buffer is written (by a DL-PUT request primitive, or by reception into the buffer at a DLCEP); and

   2) the moment when the buffer is read (by a DL-GET request primitive, or by transmission from the buffer at a DLCEP);

$$\text{DL-timeliness} \equiv 0 \leq (R_T - W_T) < \Delta T \qquad \text{(Eq. 1)}$$

   NOTE This type of timeliness was called *asynchronous* in prior national standards.

b) **UPDATE** timeliness is an assessment based upon the time interval between

   1) the moment of occurrence of a multi-DLE synchronizing event (a DL-BUFFER-RECEIVED indication or DL-BUFFER-SENT indication); and

   2) the moment when the buffer is written (by a DL-PUT request primitive, or by reception into the buffer at a DLCEP);

$$\text{DL-timeliness} \equiv 0 \leq (W_T - S_T) < \Delta T \qquad \text{(Eq. 2)}$$

   NOTE A type of timeliness closely related to this one was called *punctual* in prior national standards.

c) **SYNCHRONIZED** timeliness is an assessment based upon the time intervals and timing relationships between

   1) the moment of occurrence of a multi-DLE synchronizing event (a DL-BUFFER-RECEIVED indication or DL-BUFFER-SENT indication);

   2) the moment when the buffer is written (by a DL-PUT request primitive, or by reception into the buffer at a DLCEP); and

   3) the moment when the buffer is read (by a DL-GET request primitive, or by transmission from the buffer at a DLCEP);

$$\text{DL-timeliness} \equiv 0 \leq (W_T - S_T) \leq (R_T - S_T) < \Delta T \qquad \text{(Eq. 3)}$$

   NOTE This type of timeliness was called *synchronous* in prior national standards.

d) **TRANSPARENT** timeliness occurs when timeliness is selected on a DLCEP but none of the above assessments are performed. In such a case the DLC preserves any prior buffer timeliness, but does not itself invalidate that timeliness. When no prior buffer timeliness exists, the default timeliness value is TRUE.

e) **NO** timeliness occurs when timeliness is not selected on a DLCEP. In such a case the DL-timeliness attribute of DLS-user data always is FALSE.

The DL-time when the original buffer is written by a DL-PUT request primitive also can be conveyed to DLS-users which read a copy of the buffer. This DL-time is not available when the buffer timeliness is FALSE.

NOTE DL-time is described in Clause 8.

5 DL(SAP)-address, queue and buffer management data-link layer service

5.1 Facilities of the DL(SAP)-address, queue and buffer management data-link layer service

The DLS provides the following facilities to the DLS-user:

a) a means for creating and deleting a retentive buffer, or a non-retentive buffer, or a FIFO queue of specified depth, for use
   1) in communicating DLS-user-data between a DLS-user and the DLS-provider;
   2) in redistributing received DLS-user data without continuing DLS-user intervention; and
   3) in facilitating DLS-user supervision of the timing of DLSDU-conveyance to peer DLS-users;

b) a means for associating an individual DLSAP-address or group DL-address, referred to collectively as a DL(SAP)-address, with, and disassociating a DL(SAP)-address from, the DLSAP at which the request is made.

Default values for some quality-of-service (QoS) attributes for connection-mode and connectionless data transfer services using the specified DL(SAP)-address can be specified when the association is made.

Additionally, the DLS-user may specify that previously created buffers or FIFO queues be used for each potential direction and priority of connectionless data transfer at the specified DL(SAP)-address.

c) A means by which DLSDUs of limited size are written to or read from a buffer, or read from a FIFO queue.

5.2 Model of the DL(SAP)-address, queue and buffer management data-link layer service

This standard uses the abstract model for a layer service defined in ISO/IEC 10731, Clause 5. The model defines interactions between the DLS-user and the DLS-provider that take place at a DLSAP. Information is passed between the DLS-user and the DLS-provider by DLS primitives that convey parameters.

The DL(SAP)-address, queue and buffer management primitives are used to provide a local service between a DLS-user and the local DLE. Remote DLEs and remote DLS-users are not directly involved, so there is no need for the other primitives of ISO/IEC 10731. Therefore the DL(SAP)-address, queue and buffer management services are provided by request (requestor.submit) primitives with input and output parameters.

5.3 Sequence of primitives at one DLSAP

5.3.1 Constraints on sequence of primitives

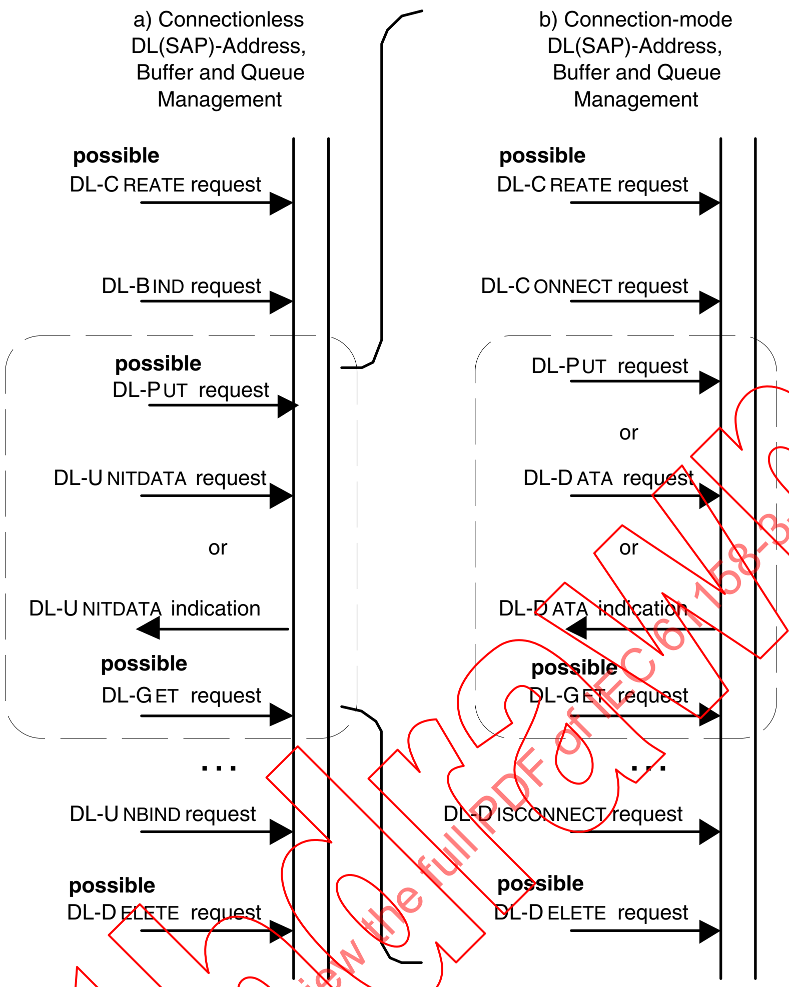
This subclause defines the constraints on the sequence in which the primitives defined in 5.4 may occur. The constraints determine the order in which primitives occur, but do not fully specify when they may occur.

The DL(SAP)-address, queue and buffer management primitives and their parameters are summarized in Table 1. The major relationships among the primitives at a single DLE are shown in Figure 4.

Table 1 – Summary of DL(SAP)-address, queue and buffer management primitives and parameters

Service	Primitive	Parameter
Buffer or queue creation	DL-CREATE request	(*in* Buffer-or-queue DLS-user-identifier, Queuing policy, Maximum DLSDU size, *out* Status, Buffer-or-queue DL-identifier)
Buffer or queue deletion	DL-DELETE request	(*in* Buffer-or-queue DL-identifier, *out* Status)
DL(SAP)-address activation	DL-BIND request	(*in* DL(SAP)-address DLS-user-identifier, DL(SAP)-address, DL(SAP)-role, Receiving-buffer-or-queue-bindings, Sending-buffer-or-queue-bindings, Default-QoS-as-sender, *out* Status, DL(SAP)-address DL-identifier)
DL(SAP)-address deactivation	DL-UNBIND request	(*in* DL(SAP)-address DL-identifier)
Update buffer	DL-PUT request	(*in* Buffer DL-identifier, DLS-user-data, DLS-user-data-timeliness, *out* Status)
Copy buffer or dequeue	DL-GET request	(*in* Buffer-or-queue DL-identifier, *out* Status, Reported-service-identification-class, Reported-service-identification, DLS-user-data, DLS-user-data-timeliness)

NOTE DL-identifiers in parameters are local and assigned by the DLS-provider and used by the DLS-user to designate a specific DL(SAP)-address, DLCEP, schedule, buffer-or-queue to the DLS-provider at the DLS interface. DLS-user-identifiers in parameters are local and assigned by the DLS-user and used by the DLS-provider to designate a specific DL(SAP)-address, DLCEP, schedule, buffer-or-queue to the DLS-user at the DLS interface.



NOTE 1 Primitives within the outlined areas can be repeated many times between instances of the primitives in the enclosing areas.

NOTE 2 The entire right-hand part of the figure is another alternative to the outlined area of the left-hand part of the figure, and also can be repeated many times between instances of the primitives in the left-hand enclosing area.

NOTE 3 DL-PUT and DL-GET request primitives both can be used earlier and later than shown.

Figure 4 – Sequence of primitives for the DL(SAP)-address, queue and buffer management DLS

5.4 DL(SAP)-address, queue and buffer management facilities

DL(SAP)-address management facilities bind a DL(SAP)-address to, and unbind a previously bound DL(SAP)-address from, the DLSAP at which the primitive is invoked. Such a binding is required while communicating using the specified DL(SAP)-address.

Queue and buffer management facilities permit, but do not require, a DLS-user to use retentive or non-retentive buffers or specified depth FIFO queues when employing the DLS-provider's data communications facilities. (See Figure 5.) Since these buffers and queues are managed by the DLS-provider, they support DLS-user interactions and data transfer and scheduling paradigms not available with DLS-user-based queuing.

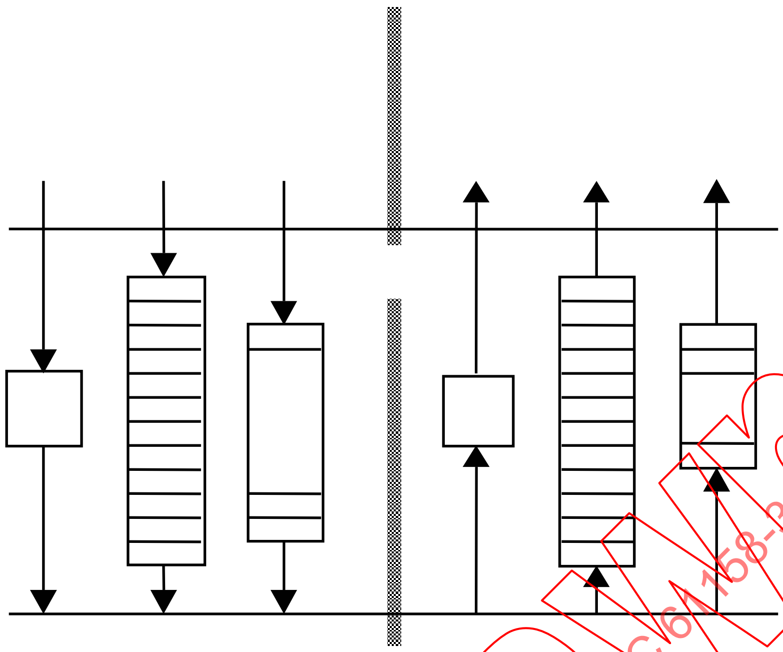


Figure 5 – Supported methods of data management for transmission and delivery

5.4.1 Create

5.4.1.1 Function

The create buffer or queue DLS primitive can be used to create a retentive buffer or non-retentive buffer or limited-depth FIFO queue for later constrained association with a DLSAP — either through a DL(SAP)-address or through a DLCEP. The resulting buffer or FIFO queue initially will be empty.

NOTE This facility may also be provided by local DL-management actions which are beyond the scope of this standard.

5.4.1.2 Types of parameters

Table 2 indicates the primitive and parameters of the create buffer or queue DLS.

Table 2 – DL-buffer-and-queue-management create primitive and parameters

DL-CREATE	Request	
Parameter name	input	output
Buffer-or-queue DLS-user-identifier	M	
Queuing policy	M	
Maximum queue depth	C	
Maximum DLSDU size	M	
Status		M
Buffer-or-queue DL-identifier		C

5.4.1.2.1 Buffer-or-queue DLS-user-identifier

This parameter specifies a means of referring to the buffer or queue in output parameters of other local DLS primitives which convey the name of the buffer or queue from the local DLE to the local DLS-user.

The naming-domain of the buffer-or-queue DLS-user-identifier is the DLS-user's local-view.

5.4.1.2.2 Queuing policy

This parameter specifies whether to create either

a) **BUFFER-R** — a retentive buffer, whose contents are not affected by being read, which can be overwritten (as a single atomic action) by either the DLS-provider or DLS-user, and which can be used only with the connectionless unitdata-exchange and connection-oriented data services; or

b) **BUFFER-NR** — a non-retentive buffer, which is set empty after being read, which can be overwritten (as a single atomic action) by either the DLS-provider or DLS-user, and which can be used only with the connectionless unitdata-exchange service; or

c) **QUEUE** — a FIFO queue of maximum depth *K* which contains between zero and *K* DLSDUs, which will reject attempts to remove DLSDUs when empty and to append DLSDUs when full, and which can be used with the connectionless unitdata-transfer and connection-oriented data services, and for DLSDU-receipt with the connectionless unitdata-exchange service.

The values of the Queuing Policy are BUFFER-R, BUFFER-NR and QUEUE.

NOTE 1 Buffer and queue bindings result from DL-BIND request, DL-CONNECT request and DL-CONNECT response primitives.

NOTE 2 An explicit queue needs to have one output binding and at least one input binding to be useful. Multiple input bindings provide a means of coalescing inputs from many sources into a single queue. Multiple output bindings are not permitted, because a queue-not-empty condition typically causes transmission to be attempted at the DLCEP, or from the DLSAP-address, to which the queue is bound.

NOTE 3 A retentive buffer (BUFFER-R) needs to have one input binding and at least one output binding to be useful. Multiple input bindings are not permitted, because they would permit any data source to overwrite the buffer at any time, with no indication to the buffer users of which source was involved. Multiple output bindings are useful, to share the contents of the buffer among multiple users or to support differing-rate redistribution of cached data within bridges.

NOTE 4 A non-retentive buffer (BUFFER-NR) needs to have one input binding and one output binding to be useful. Multiple input bindings are not useful for the reason stated in 2. Multiple output bindings are not useful; their existence would lead to non-intuitive semantics for the buffer.

NOTE 5 Buffers can be overwritten at any time, with complete loss of the prior contents except to those users which had begun to read the prior contents of the buffer (via DL-GET request primitives or sending DLCEPs) before the overwriting begins. Therefore buffers are well suited for caching of distributed data, where it is desired to retain the most recent value of received information, and are poorly suited for general messaging.

5.4.1.2.2.1 Maximum queue depth

This parameter is present when the Queuing Policy parameter has the value QUEUE. When present, this parameter specifies *K*, the maximum number of items in the associated queue. The supported values for this parameter should include the values one (1), two (2), three (3), and four (4).

NOTE Implementations of this DLS may extend the range of this parameter to include

a) values greater than four (4); and

b) the value zero (0), creating a null queue.

5.4.1.2.3 Maximum DLSDU size

This parameter specifies an upper bound on the size (in octets) of DLSDUs that can be put into the buffer or queue. The maximum size permitted for such DLSDUs may be constrained by a companion DL-protocol specification and by DL-management

NOTE This parameter does not preclude implementations from using a fixed, small set of record sizes when allocating buffers or entries in a queue.

5.4.1.2.4 Status

This parameter allows the DLS-user to determine whether the requested DLS was provided successfully, or failed for the reason specified. The value conveyed in this parameter is as follows:

a) “success”;

b) “failure — insufficient resources”;

c) "failure — parameter violates management constraint";

d) "failure — number of requests violates management constraint"; or

e) "failure — reason unspecified".

NOTE Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this standard.

If the status is "success", then the created buffer or queue is subject to the following constraints:

1) If a BUFFER-NR or BUFFER-R was created, then it can be written explicitly either

— by one priority of a receiving DLSAP-address whose DL(SAP)-role is INITIATOR, CONSTRAINED RESPONDER or UNCONSTRAINED RESPONDER;

— by a receiving DLCEP; or

— by a DLS-user through a DL-PUT request primitive, if no other binding exists to write the buffer.

It is not permitted to bind such a buffer so that it is written from two distinct sources.

2) If a BUFFER-NR or QUEUE was created, then it can be read explicitly either

— by one priority of a sending DLSAP-address whose DL(SAP)-role is BASIC;

— by a sending DLCEP; or

— by a DLS-user through a DL-GET request primitive, if no other binding exists to read the buffer.

It is not permitted to bind such a buffer or queue so that it is read by two distinct sinks.

5.4.1.2.5 Buffer-or-queue DL-identifier

This parameter is present when the Status parameter indicates that the DL-CREATE request primitive was successful. The buffer-or-queue DL-identifier parameter gives the local DLS-user a means of referring to the buffer or queue in input parameters of other local DLS primitives which convey the name of the buffer or queue from the local DLS-user to the local DLE.

The naming-domain of the buffer-or-queue DL-identifier is the DL-local-view.

5.4.1.3 Sequence of primitives

The sequence of primitives in creating, later using, and eventually deleting a buffer or queue is defined in the time sequence diagram of Figure 4.

5.4.2 Delete

5.4.2.1 Function

The delete buffer or queue DLS primitive can be used to delete a buffer or queue created by an earlier create buffer or queue DLS primitive.

NOTE 1 This primitive can only be used to delete a buffer or queue that was created by a prior DL-CREATE request primitive; it cannot be used to delete a buffer or queue that was created by prior local DL-management actions.

NOTE 2 This facility may also be provided by local DL-management actions, which are beyond the scope of this standard.

5.4.2.2 Types of parameters

Table 3 indicates the primitive and parameters of the delete buffer or queue DLS.

Table 3 – DL-buffer-and-queue-management delete primitive and parameters

DL-DELETE Parameter name	Request input	Request output
Buffer-or-queue DL-identifier	M	
Status		M

5.4.2.2.1 Buffer-or-queue DL-identifier

This parameter specifies the local DL-identifier returned by a successful prior DL-CREATE request primitive whose buffer or queue has not yet been deleted. The DLS-provider will release the local DL-identifier and associated DLS-provider resources.

The DLS-user may not delete a buffer or queue that is still associated with a DLSAP.

NOTE Such associations can occur only as a result of a DL-BIND request, or DL-CONNECT request or response, or of DL-management action.

5.4.2.2.2 Status

This parameter allows the DLS-user to determine whether the requested DLS was provided successfully, or failed for the reason specified. The value conveyed in this parameter is as follows:

a) “success”;

b) “failure — resource in use”;

c) “failure — management-controlled resource”; or

d) “failure — reason unspecified”.

NOTE Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this standard.

5.4.2.3 Sequence of primitives

The sequence of primitives in creating and later deleting a buffer or queue is defined in the time sequence diagram of Figure 4.

5.4.3 Bind

5.4.3.1 Function

The bind DL(SAP)-address DLS primitive is used

a) to associate a DL(SAP)-address with the DLSAP at which the primitive is invoked;

b) to establish the DL(SAP)’s role, if any, when participating in the DL-Unitdata and DL-Unitdata-Exchange connectionless services at that DL(SAP)-address;

c) to associate up to six previously created retentive buffers or non-retentive buffers or limited-depth FIFO queues with the various priorities and directions of potential data transfer at the specified DL(SAP)-address; and

d) to specify default values for some quality-of-service (QoS) attributes for connection-mode and connectionless data transfer services using the specified DL(SAP)-address.

NOTE This facility may also be provided by local DL-management actions, which are beyond the scope of this standard.

5.4.3.2 Types of parameters

Table 4 indicates the primitive and parameters of the bind DL(SAP)-address DLS.

Table 4 – DL(SAP)-address-management bind primitive and parameters

DL-BIND Parameter name	Request input	Request output
DL(SAP)-address DLS-user-identifier	M	
DL(SAP)-address	M	
DL(SAP)-role	M	
Indicate-null-UNITDATA-EXCHANGE-transactions	C	
Remote-DLSAP-address	C	
Receiving-buffer-or-queue-bindings		
URGENT-buffer-or-queue DL-identifier	U	
NORMAL-buffer-or-queue DL-identifier	U	
TIME-AVAILABLE-buffer-or-queue DL-identifier	U	
Sending-buffer-or-queue-bindings		
URGENT-buffer-or-queue DL-identifier	CU	
NORMAL-buffer-or-queue DL-identifier	CU	
TIME-AVAILABLE-buffer-or-queue DL-identifier	CU	
Default QoS as sender		
DLL priority	CU	
DLL maximum confirm delay		
on DL-CONNECT, DL-RESET, DL-SUBSCRIBER-QUERY	CU	
on DL-DATA	CU	
on locally-confirmed DL-UNITDATA	CU	
on remotely-confirmed DL-UNITDATA, DL-LISTENERQUERY, DL-UNITDATA-EXCHANGE	CU	
DLPDU authentication	CU	
DL-scheduling-policy	CU	
Status		M
DL(SAP)-address DL-identifier		C

5.4.3.2.1 DL(SAP)-address DLS-user-identifier

This parameter specifies a means of referring to the DL(SAP)-address in output parameters of other local DLS primitives which convey the name of the DL(SAP)-address from the local DLE to the local DLS-user.

The naming-domain of the DL(SAP)-address DLS-user-identifier is the DLS-user's local-view.

5.4.3.2.2 DL(SAP)-address

This parameter specifies an individual local DLSAP-address or group DL-address to be associated with the invoking (necessarily local) DLSAP.

5.4.3.2.3 DL(SAP)-role

This parameter constrains, as specified in Table 5, the DL-connectionless DL-UNITDATA and DL-UNITDATA-EXCHANGE service primitives that can be issued with this DL(SAP)-address at the local DLSAP (to which this DL(SAP)-address is being bound). It also constrains whether the DL(SAP)-address may have an associated DLCEP and the permitted types of bindings (implicit queue, explicit queue or explicit buffer) to the DL(SAP)-address. Permitted values for this parameter are specified in Table 5.

Table 5 – DL(SAP)-role constraints on DLSAPs, DLCEPs and other DLS Primitives

DL(SAP)-role	Sending DLSAP-Address bindings	Receiving DL(SAP)-Address bindings	DLCEP permitted	DL-Unitdata Request permitted	DL-Unitdata Indication possible	Unitdata-Exchange Indication possible
BASIC	IQ, EQ	IQ, EQ	yes	yes	yes	no
GROUP	—	IQ, EQ	no	no	yes	No
INITIATOR	EB	EB, EQ	no	no	no	yes
CONSTRAINED RESPONDER	EB	EB, EQ	no	no	no	yes
UNCONSTRAINED RESPONDER	EB	EB, EQ	no	no	no	yes

Key

— : not applicable	IQ : implicit queue on transmit, immediate (as soon as possible) delivery on receipt
EQ : explicit queue	EB : explicit retentive or non-retentive buffer

The default value for this parameter is BASIC.

NOTE The roles of INITIATOR, CONSTRAINED RESPONDER, and UNCONSTRAINED RESPONDER support migration from previous national standards.

5.4.3.2.3.1 Indicate-null-unitdata-exchange-transactions

This Boolean parameter is present when the DL(SAP)-role parameter has the value INITIATOR, CONSTRAINED RESPONDER, or UNCONSTRAINED RESPONDER. When present, the indicate-null-UNITDATA-EXCHANGE-transactions parameter specifies whether an instance of a UNITDATA-EXCHANGE transaction which occurs at this DLSAP-address generates a DL-UNITDATA-EXCHANGE indication even when no DLS-user data transfer (in either direction) occurred.

NOTE 1 Such an indication would attest to the DLS-user that communication with the DLE of the addressed remote peer DLS-user was still possible. In other DL-protocols in which all DLS-users are on the same unbridged local link, this attestation is sometimes provided by a "live list".

NOTE 2 UNITDATA-EXCHANGE and the use of the indicate-null-UNITDATA-EXCHANGE-transactions parameter are covered in Clause 7.

5.4.3.2.3.2 Remote-DLSAP-address

This parameter is present when the DL(SAP)-role parameter has the value CONSTRAINED RESPONDER. When present, this parameter specifies an individual DLSAP-address, specifying that the DL-UNITDATA-EXCHANGE service may be initiated only from the specified DLSAP-address.

5.4.3.2.4 Receiving buffer-or-queue bindings

When present, each buffer-or-queue DL-identifier parameter specifies the local DL-identifier returned by a successful prior DL-CREATE request primitive (or DL-management action) that created a buffer or queue, which has not yet been deleted.

When the DL(SAP)-role parameter has the value BASIC or GROUP, then

a) explicit bindings to a buffer are not permitted;

b) explicit bindings to a queue are permitted; and

c) if no binding at a given DLL-priority exists, then the DLSAP-address is implicitly bound at that priority to the default OSI delivery service, which is immediate (as soon as possible) delivery.

When bound as in b), the maximum-DLSDU-size for each specified receive queue should accommodate the maximum amount of DLS-user data permitted within a single DLPDU of the priority corresponding to the binding, as specified in 4.3.1.

When the DL(SAP)-role parameter has the value INITIATOR, CONSTRAINED RESPONDER, or UNCONSTRAINED RESPONDER, then

1) explicit bindings to a buffer are permitted;
2) explicit bindings to a queue are permitted; and
3) if no binding at a given DLL-priority exists, then it is not possible to receive a DLSDU of that priority at that DLSAP-address.

NOTE If a queue is bound to the receiving (DLS-provider to DLS-user) direction of data transfer at a DL(SAP)-address at a specified priority, as in b) or e), then the DLS-user has specified the maximum number of queued received DLSDUs, and can choose when to process those DLSDUs.

5.4.3.2.4.1 Urgent buffer-or-queue DL-identifier

When permitted, specifying a buffer-or-queue DL-identifier results in the identified buffer or queue being bound to the DLSAP for use in reception at URGENT priority.

5.4.3.2.4.2 Normal buffer-or-queue DL-identifier

When permitted, specifying a buffer-or-queue DL-identifier results in the identified buffer or queue being bound to the DLSAP for use in reception at NORMAL priority.

5.4.3.2.4.3 Time-available buffer-or-queue DL-identifier

When permitted, specifying a buffer-or-queue DL-identifier results in the identified buffer or queue being bound to the DLSAP for use in reception at TIME-AVAILABLE priority.

5.4.3.2.5 Sending buffer-or-queue bindings

When present, each buffer-or-queue DL-identifier parameter specifies the local DL-identifier returned by a successful prior DL-CREATE request primitive (or DL-management action) that created a buffer or queue that has not yet been deleted.

When the DL(SAP)-role parameter has the value BASIC, then

a) explicit bindings to a buffer are not permitted;
b) explicit bindings to a queue are permitted; and
c) if no binding at a given DLL-priority exists, then the DLSAP-address is implicitly bound at that priority to a default OSI queue.

NOTE If a queue is bound to the sending (DLS-user to DLS-provider) direction of data transfer at a DLSAP-address at a specified priority, as in b) or c), then transmission of the queued DLSDUs in FIFO order will be attempted, as appropriate, when the queue is non-empty.

When the DL(SAP)-role parameter has the value INITIATOR, CONSTRAINED RESPONDER, or UNCONSTRAINED RESPONDER, then

1) explicit bindings to a buffer are permitted;
2) explicit or implicit bindings to a queue are not permitted; and
3) if no binding at a given DLL-priority exists, then it is not possible to source a DLSDU at that priority from that DLSAP-address.

When the DL(SAP)-role parameter has the value GROUP, then no bindings are permitted or implied; it is not possible to attribute a group DL-address as the source of a DLSDU.

5.4.3.2.5.1 Urgent buffer-or-queue DL-identifier

When permitted, specifying a buffer-or-queue DL-identifier results in the identified buffer or queue being bound to the DLSAP for use in transmission at URGENT priority.

5.4.3.2.5.2 Normal buffer-or-queue DL-identifier

When permitted, specifying a buffer-or-queue DL-identifier results in the identified buffer or queue being bound to the DLSAP for use in transmission at NORMAL priority.

5.4.3.2.5.3 Time-available buffer-or-queue DL-identifier

When permitted, specifying a buffer-or-queue DL-identifier results in the identified buffer or queue being bound to the DLSAP for use in transmission at TIME-AVAILABLE priority.

5.4.3.2.6 Default QoS as sender

The DLS-user may specify default values for some of the QoS parameters that apply to connection-mode and connectionless data transmission, as described in 4.3. These default values will be used whenever data or unitdata transmission services are initiated at this DLSAP-address, unless explicitly overridden during an actual service invocation.

When the DL(SAP)-role parameter has the value INITIATOR or CONSTRAINED RESPONDER or UNCONSTRAINED RESPONDER, some of these QoS attributes as sender are irrelevant and should be absent. When the DL(SAP)-role parameter has the value GROUP, all of these QoS attributes as sender are irrelevant and should be absent.

5.4.3.2.6.1 DLL priority

This QoS attribute is not relevant when the DL(SAP)-role parameter has the value INITIATOR, CONSTRAINED RESPONDER, UNCONSTRAINED RESPONDER, or GROUP, and thus should be absent.

5.4.3.2.6.2 DLL maximum confirm delay

This QoS attribute is not relevant when the DL(SAP)-role parameter has the value CONSTRAINED RESPONDER, UNCONSTRAINED RESPONDER, or GROUP, and thus should be absent.

5.4.3.2.6.3 DLPDU authentication

This QoS attribute is not relevant when the DL(SAP)-role parameter has the value GROUP, and thus should be absent.

5.4.3.2.6.4 DL-scheduling-policy

This QoS attribute is not relevant when the DL(SAP)-role parameter has the value INITIATOR, CONSTRAINED RESPONDER, UNCONSTRAINED RESPONDER, or GROUP, and thus should be absent.

5.4.3.2.7 Status

This parameter allows the DLS-user to determine whether the requested DLS was provided successfully, or failed for the reason specified. The value conveyed in this parameter is as follows:

a) “success”;

b) “failure — insufficient resources”;

c) “failure — DL(SAP)-address invalid or unavailable”;

d) “failure — DL(SAP)-role not supported”;

e) “failure — remote DL(SAP)-address invalid”;

f) “failure — invalid buffer or queue binding”;

g) “failure — parameter inconsistent with DL(SAP)-role”;

h) “failure — parameter violates management constraint”;

i) “failure — number of requests violates management constraint”; or

j) “failure — reason unspecified”.

NOTE Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this standard.

5.4.3.2.8 DL(SAP)-address DL-identifier

The DL(SAP)-address DL-identifier parameter is present when the status parameter indicates that the DL-BIND request primitive was successful. The DL(SAP)-address DL-identifier parameter gives the local DLS-user a means of referring to the DL(SAP)-address in input parameters of other local DLS primitives which convey the name of the DL(SAP)-address from the local DLS-user to the local DLE.

The naming-domain of the DL(SAP)-address DL-identifier is the DL-local-view.

5.4.3.3 Sequence of primitives

The sequence of primitives in

a) binding a DL(SAP)-address to the invoking DLSAP, and optionally binding one or more buffers or queues to the DL(SAP)-address; and
b) later unbinding the DL(SAP)-address and buffers or queues from the DLSAP

is defined in the time sequence diagram of Figure 4.

5.4.4 Unbind

5.4.4.1 Function

The unbind DL(SAP)-address DLS primitive is used to unbind a DL(SAP)-address from the invoking DLSAP. Any buffers or queues that were explicitly bound to the DL(SAP)-address are also unbound from that DL(SAP)-address at the same time.

NOTE 1 This primitive can only be used to unbind a DL(SAP)-address that was bound to the DLSAP by a prior DL-BIND request primitive. It cannot be used to unbind a DL(SAP)-address that was bound to the DLSAP by prior local DL-management actions.

NOTE 2 This facility may also be provided by local DL-management actions, which are beyond the scope of this standard.

5.4.4.2 Types of parameters

Table 6 indicates the primitive and parameters of the unbind DL(SAP)-address DLS.

Table 6 – DL(SAP)-address-management unbind primitive and parameters

DL-UNBIND	Request
Parameter name	input
DL(SAP)-address DL-identifier	M

5.4.4.2.1 DL(SAP)-address DL-identifier

This parameter specifies the local DL-identifier returned by a successful prior DL-BIND request primitive. The DLS-provider should unbind the local DL-identifier and its associated DL(SAP)-address, and any associated buffers or queues, from the invoking DLSAP, after first disconnecting all DLCEPs, unbinding all buffer and queues which were bound to those DLCEPs, and terminating all outstanding DL-UNITDATA requests associated with that DLSAP-address.

5.4.4.3 Sequence of primitives

The sequence of primitives in

a) binding a DL(SAP)-address, and possibly buffers or queues, with the invoking DLSAP; and

b) later unbinding the DL(SAP)-address and those buffers or queues from the DLSAP

is defined in the time sequence diagram of Figure 4.

5.4.5 Put

5.4.5.1 Function

The put buffer DLS primitive is used to copy a DLSDU to a buffer. In some cases it may also be used to set the buffer empty.

5.4.5.2 Types of parameters

Table 7 indicates the primitive and parameters of the put buffer DLS.

Table 7 – DL-buffer-management put primitive and parameters

DL-PUT Parameter name	Request input	 output
Buffer DL-identifier	M	
DLS-user-data	U	
DLS-user-data-timeliness	U	
Status		M

5.4.5.2.1 Buffer DL-identifier

This parameter specifies the local DL-identifier returned by a successful prior DL-CREATE request primitive which created a buffer (or by DL-management).

5.4.5.2.2 DLS-user-data

When present, this parameter specifies one or more octets of DLS-user-data, up to the maximum DLSDU size specified in the associated DL-CREATE request primitive, possibly further constrained by DLC negotiation. The DLE may also note the current DL-time for later reporting as the time-of-production.

When DLS-user-data is not present, then

a) If the buffer is bound to a DLCEP-address, then the DL-Put request fails and a status of “failure — invalid DLSDU size” is returned to the requesting DLS-user.
b) Otherwise, when a) does not apply, then the DL-Put request primitive resets the buffer to its initial empty state.

5.4.5.2.3 DLS-user-data-timeliness

This parameter specifies whether the associated DLS-user-data meets the requesting DLS-user’s timeliness criteria, or not. Its value is either TRUE (one or more criteria exist, and all were met) or FALSE (either no criteria exist, or one or more of the criteria were not met).

If the data in this buffer is then sent to other DLS-users through a DLC, then this timeliness attribute will be logically ANDed with the assessment(s) of any DLCEP-evaluated timeliness criteria, and the result associated with the receiving buffer, if any.

NOTE Buffer timeliness is presented to the DLS-user(s) by the DL-GET primitive.

The default value for this parameter is FALSE.

5.4.5.2.4 Status

This parameter allows the DLS-user to determine whether the requested DLS was provided successfully, or failed for the reason specified. The value conveyed in this parameter is as follows:

a) "success";

b) "failure — invalid DLSDU size"; or

c) "failure — reason unspecified".

NOTE Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this standard.

5.4.5.3 Sequence of primitives

The sequence of primitives in using a buffer is defined in the time sequence diagram of Figure 4.

5.4.6 Get

5.4.6.1 Function

The get buffer or queue DLS primitive is used to read a DLSDU from a buffer, or attempt to remove a DLSDU from a FIFO queue.

When the DL-GET request primitive specifies a buffer (see 5.4.5.2.1, 6.3.2.9.1a), and 7.5.2.2.3) and the buffer is empty, then the DL-GET request primitive returns an appropriate failure status. Otherwise, the DL-GET request primitive copies the DLSDU from the buffer and returns it.

When the DL-GET request primitive specifies an explicit queue and the queue is empty, then the DL-GET request primitive returns an appropriate failure status. Otherwise, the DL-GET request primitive dequeues the DLSDU from the FIFO queue and returns it, together with associated DL(SAP)-addresses or an associated DLCEP DL-identifier, if appropriate.

5.4.6.2 Types of parameters

Table 8 indicates the primitive and parameters of the get buffer or queue DLS.

Table 8 – DL-buffer-and-queue-management get primitive and parameters

DL-GET	Request	
Parameter name	**input**	**output**
Buffer-or-queue DL-identifier	M	
Status		M
Reported-service-identification-class		C
Reported-service-identification		
Receiving DLCEP DLS-user-identifier		C
Called DL(SAP)-address DLS-user-identifier		C
Calling DLSAP-address		C
DLL-priority		C
DLS-user-data		C
DLS-user-data-timeliness		
Local DLE timeliness		C
Sender and remote DLE timeliness		C
Time-of-production		C

5.4.6.2.1 Buffer-or-queue DL-identifier

This parameter specifies the local DL-identifier returned by a successful prior DL-CREATE request primitive (or by DL-management).

5.4.6.2.2 Status

This parameter allows the DLS-user to determine whether the requested DLS was provided successfully, or failed for the reason specified. Attempting to copy a DLSDU from a buffer that is empty, or to remove a DLSDU from a FIFO queue that is empty, will result in failure. The value conveyed in this parameter is as follows:

a) "success";
b) "possible failure — empty buffer";
c) "failure — empty queue"; or
d) "failure — reason unspecified".

NOTE 1 An empty buffer can be considered to contain a null DLSDU when used with the unitdata exchange service, and so may be treated as "success" by the DLS-user.

NOTE 2 Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this standard.

5.4.6.2.3 Reported-service-identification-class

This parameter is present when the status parameter indicates that the DL-GET request was successful. When present, this parameter specifies the structure of the associated reported-service-identification parameter. The values of this parameter are:

a) none — implying that a buffer is being retrieved and that source information is not present;
b) DLCEP — the receiving DLCEP DL-identifier is present; or
c) DL(SAP)-address — the receiving DL(SAP)-address DL-identifier, the sending DLSAP-address, and the priority of the DLSDU are present.

5.4.6.2.4 Reported-service-identification

This compound parameter is present when the status parameter indicates that the DL-GET request was successful. When present, this parameter identifies the service endpoint(s) and priority of the reported DLSDU.

5.4.6.2.4.1 Receiving-DLCEP DLS-user-identifier

This parameter is present when the reported-service-identification-class parameter specifies DLCEP. When present, this parameter identifies the DLCEP at which the associated DLSDU was received.

5.4.6.2.4.2 Called-DL(SAP)-address DLS-user-identifier

This parameter is present when the reported-service-identification-class parameter specifies DL(SAP)-ADDRESS. When present, this parameter identifies the destination DL(SAP)-address at which the associated DLSDU was received.

5.4.6.2.4.3 Calling-DLSAP-address

This parameter is present when the reported-service-identification-class parameter specifies DL(SAP)-ADDRESS. When present, this parameter identifies the source DLSAP-address from which the associated DLSDU was sent.

NOTE If the DLS-user has issued a DL-BIND request for the calling-DLSAP-address, then this parameter also can take the form of a DLSAP-address DLS-user-identifier.

5.4.6.2.4.4 DLL-priority

This parameter is present when the reported-service-identification-class parameter specifies DL(SAP)-ADDRESS. When present, this parameter identifies the sender's DLL priority of the received DLSDU.

5.4.6.2.5 DLS-user-data

This parameter is present when the status parameter indicates that the DL-GET request was successful. This parameter returns a single DLSDU consisting of one or more octets of DLS-user-data, up to the maximum DLSDU size specified in the associated DL-CREATE request primitive (or by DL-management).

5.4.6.2.6 DLS-user-data-timeliness

This parameter is present when the status parameter indicates that the DL-GET request was successful and the buffer-or-queue DL-identifier parameter specifies a buffer. When present, this parameter specifies up to three aspects of DLS-user-data timeliness.

5.4.6.2.6.1 Local-DLE-timeliness

This parameter specifies whether the associated DLS-user-data met the receiving DLCEP's timeliness criteria, or not. Its value is either TRUE (timeliness criteria exist, and all were met) or FALSE (either no timeliness criteria exist, or one or more of the criteria were not met). When the buffer is not bound as a sink to a DLCEP, but is instead written by a local DL-PUT request, then the value of this parameter is always TRUE. When the buffer is bound as a sink to a DLSAP-address whose DL(SAP)-role is INITIATOR or CONSTRAINED RESPONDER or UNCONSTRAINED RESPONDER, then the value of this parameter is always FALSE.

5.4.6.2.6.2 Sender-and-remote-DLE-timeliness

This parameter specifies whether the associated DLS-user-data met both the sending DLS-user's timeliness criteria, and any sending and intermediary receiving DLCEPs' timeliness criteria, or not. Its value is either TRUE (so specified by the sending DLS-user, and timeliness criteria exist for each of the transporting DLCEPs, and all were met) or FALSE (either so specified by the sending DLS-user, or timeliness was not selected for one or more of the transporting DLCEP(s), or one or more of the timeliness criteria were not met).

NOTE DL-timeliness is partitioned into separate receiver timeliness and sender timeliness:

a) to distinguish between those aspects of timeliness whose criteria are specified by the receiving DLS-user and those which are not;

b) to provide assistance in DLS-user diagnosis of the source of non-timeliness — either the DLS-user-data source and remote portions of the distributed DLS-provider, or the local DLE and the receiving DLS-user; and

c) to facilitate migration of existing national standards.

5.4.6.2.6.3 Time-of-production

This parameter is present only when

a) the sending DLCEP specified TRUE for its time-of-production QoS parameter (see 6.3.2.10.1.4); and

b) the value of the sender-and-remote-DLE-timeliness parameter (see 5.4.6.2.6.2) returned by this DL-GET primitive is TRUE.

This parameter specifies the DL-time at which a DLS-user transferred the associated DLS-user-data to the (distributed) DLS-provider.

NOTE 1 For a buffer bound as a source at a DLCEP, this is the DL-time at which the sending DLS-user issued the DL-PUT request which placed the associated DLS-user-data in the sending DL-buffer.

NOTE 2 If a DL-relay entity (a bridge) acts as a distributor by connecting a receiving DLCEP to a second explicitly-scheduled sending DLCEP through a shared intermediate buffer, then at the final receiver the time-of-production is still the time at which the sending DLS-user (at the first DLCEP) issued the DL-PUT request which placed the associated DLS-user-data in the original sending DL-buffer.

5.4.6.3 Sequence of primitives

The sequence of primitives in using a buffer or queue is defined in the time sequence diagram of Figure 4.

6 Connection-mode data-link layer service

6.1 Facilities of the connection-mode data-link layer service

The DLS provides the following facilities to the DLS-user:

a) A means to establish (see Figure 6) either

   1) a peer DLC between two DLS-users for exchanging DLSDUs between the two DLS-users, or

      NOTE It is also possible for a peer DLC to be negotiated to provide just DL-simplex transmission from one of the DLS-users to the other.

   2) a multi-peer DLC between a single publishing DLS-user and a set of subscribing DLS-users for sending DLSDUs

      i) from the publishing DLS-user to the set of subscribing DLS-users, and

      ii) optionally, from any of the subscribing DLS-users to the publishing DLS-user.

      NOTE It is also possible for a multi-peer DLC to be negotiated to provide just DL-simplex transmission from the publishing DLS-user to the set of subscribing DLS-users, or from the set of subscribing DLS-users to the publishing DLS-user.

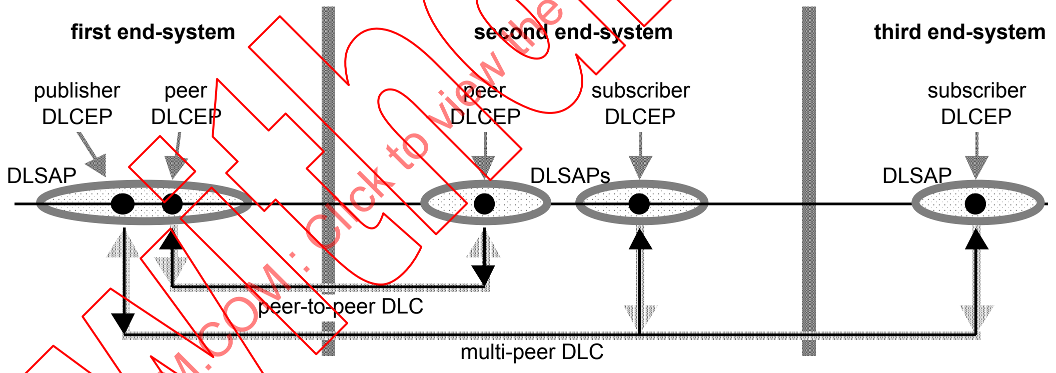


Figure 6 – Peer-to-peer and multi-peer DLCs and their DLCEPs

b) A means of establishing an agreement for a certain quality-of-service (QoS) associated with a DLC, between the initiating DLS-user, the responding DLS-user(s), and the distributed DLS-provider.

c) A means of transferring DLSDUs of limited length on a DLC. The transfer of DLSDUs is transparent, in that the boundaries of DLSDUs and the contents of DLSDUs are preserved unchanged by the DLS, and there are no constraints on the DLSDU content (other than limited length) imposed by the DLS.

   NOTE The length of a DLSDU is limited because of internal mechanisms employed by the DL-protocol (see ISO/IEC 7498-1).

d) A means of conveying timeliness information about those DLSDUs and their conveyance by the (distributed) DLS-provider in certain modes of DLC operation.

e) A means by which the receiving DLS-user at a peer DLCEP may flow control the rate at which the sending DLS-user may send DLSDUs, if supported by the DLC's QoS.

   NOTE A subscribing DLS-user of a multi-peer DLC may not flow control the rate at which the publishing DLS-user sends DLSDUs because this flow control would affect all the DLC's other DLS-users. Instead, the publishing DLS-user should employ some form of self-administered control of its publishing rate to minimize the probability of DLSDU loss due to congestion at receiving DLS-users. A sending DLS-user at a peer DLCEP whose QoS does not support flow control should adopt a similar policy of rate control.

f) A means by which a DLCEP, and possibly a DLC, can be returned to a defined state and the activities of the DLS-users resynchronized by use of a Reset DL-facility.

   NOTE Reset of a multi-peer DLCEP by a subscribing DLS-user or its local DLE does not cause the DLC to be reset at any of its other DLCEPs.

g) A means by which the publishing DLS-user of a multi-peer DLC can query whether there are any subscribing DLS-users of the DLC.

h) A means for the unconditional, and therefore possibly destructive, release of a DLCEP and possibly a DLC, by one of the DLC's DLS-users or by the DLS-provider.

   NOTE Release of a multi-peer DLCEP by a subscribing DLS-user or its local DLE does not cause the release of the DLC at any of its other DLCEPs.

There are four classes of connection-mode DLS: classes A (most comprehensive) to D (minimal). They differ in only a single QoS attribute — the available set of data delivery features (see 6.3.2.2). Classes A and B are capable of emulating OSI connection-mode and OSI connectionless services. Classes C and D are capable of emulating OSI connectionless services. All offer features that provide a basis for data distribution and distributed database cache-coherency protocols.

NOTE True connectionless services also are available, independent of these four classes. See Clause 7.

6.2 Model of the connection-mode data-link layer service

This clause uses the abstract model for a layer service defined in ISO/IEC 10731, Clause 5. The model defines interactions between the DLS-user and the DLS-provider that take place at the DLC's DLSAPs. Information is passed between the DLS-user and the DLS-provider by DLS primitives that convey parameters.

6.2.1 DLCEP-identification

A DLS-user may need to distinguish among several DLCEPs at the same DLSAP; thus a local DLCEP-identification mechanism is provided. All primitives issued at such a DLSAP within the context of a DLC use this mechanism to identify the local DLCEP. Other uses exist, such as the timeliness QoS parameters of 6.3.2.10 and the scheduling primitives of 8.5.2 and, 8.5.3. Thus this DLCEP-identification is explicitly included within the primitives that reference a local DLCEP. The naming-domain of this DLCEP-identification is the DL-local-view. (See also 3.5.2.)

6.2.2 Model of a peer DLC

Between the two end-points of a peer DLC there may exist a QoS-dependent flow control function that relates the behavior of the DLS-user receiving data to the ability of the other DLS-user to send data. As a means of specifying this flow control feature and its relationship with other capabilities provided by the connection-mode DLS, the OSI abstract queue model of a peer DLC, which is described in the following paragraphs, is used.

NOTE 1 The abstract queues referred to in this model are used solely for describing the interactions at the DLCEPs between the DLS-users, as mediated by the DLS-provider. They have no intended relationship to the actual queues managed by the primitives of Clause 5.

This abstract queue model of a peer DLC is discussed only to aid in the understanding of the end-to-end DLS features perceived by DLS-users. It is not intended to serve as a substitute for a precise, formal description of the DLS, nor as a complete specification of all allowable sequences of DLS-primitives. (Allowable primitive sequences are specified in 6.4. See also the note below.) In addition, this model does not attempt to describe all the functions or operations of DLEs that are used to provide the DLS. No attempt to specify or constrain DLE implementation is implied.

NOTE 2 The internal mechanisms that support the operation of the DLS are not visible to the DLS-user. Along with the interactions between service primitives described by this model (for example, the issue of a DL-RESET request at a DLSAP may prevent the receipt of a DL-DATA indication, corresponding to a previously issued DL-DATA request, by the peer DLS-user) there may also be

a) constraints applied locally on the ability to invoke DLS-primitives; and

b) service procedures defining particular sequencing constraints on some DLS-primitives.

6.2.2.1 OSI abstract queue model concepts

The OSI abstract queue model represents the operation of a peer DLC in the abstract by a pair of abstract queues linking the two DLCEPs. There is one abstract queue for each direction of information flow (see Figure 7).

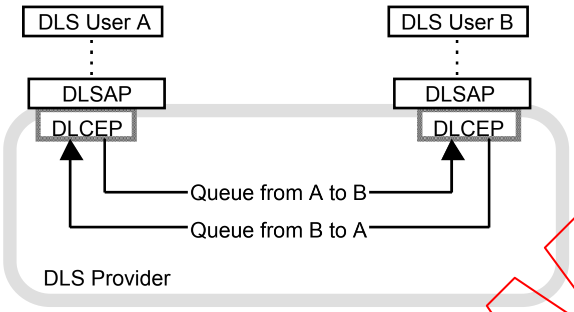


Figure 7 – OSI abstract queue model of a peer DLC between a pair of DLS-users

Each OSI abstract queue represents a flow control function in one direction of transfer. The ability of a DLS-user to add objects to an abstract queue will be determined by the behavior of the other DLS abstract queue. Objects are entered or removed from the abstract queue as a result of interactions at the two DLSAPs. Each peer DLC is considered to have one pair of abstract queues.

The following objects may be placed in an abstract queue by a DLS-user:

a) a connect object, representing a DL-Connect primitive and its parameters;
b) a data object, representing a DL-Data primitive and its parameters;
c) a reset object, representing a DL-Reset primitive and its parameters; and
d) a disconnect object, representing a DL-Disconnect primitive and its parameters.

The following objects may be placed in an abstract queue by the DLS-provider:

1) a reset object, representing a DL-Reset primitive and its parameters;
2) a synchronization mark object (6.2.2.4); and
3) a disconnect object, representing a DL-Disconnect primitive and its parameters.

The abstract queues are defined to have the following general properties:

i) An abstract queue is empty before a connect object has been entered, and can be returned to this state, with loss of its contents, by the DLS-provider.
ii) Objects are entered into an abstract queue by the sending DLS-user, subject to control by the DLS-provider. Objects may also be entered by the DLS-provider.
iii) Objects are removed from the abstract queue under the control of the receiving DLS-user.
iv) Objects are normally removed in the same order that they were entered (however, see 6.2.2.3).
v) An abstract queue has a limited capacity, but this capacity is not necessarily either fixed or determinable.
vi) Depending on the peer DLC's QoS, the abstract queue may occasionally duplicate or lose data objects.

6.2.2.2 Peer DLC / DLCEP establishment

When the DLS-provider receives a DL-CONNECT request primitive at one of the DLSAPs, it associates a pair of abstract queues and an abstract peer DLC between the two DLSAPs, and enters either a connect object or a disconnect object into the appropriate abstract queue. From the standpoint of the DLS-users of the peer DLC, the abstract queues remain associated with the peer DLC until a disconnect object representing a DL-DISCONNECT primitive is either entered into or removed from the abstract queue.

DLS-user A, which initiates a peer DLC establishment by entering a connect object representing a DL-CONNECT request primitive into the abstract queue from DLS-user A to DLS-user B, is not allowed to enter any other object, other than a disconnect object, into the abstract queue until after the connect object representing the DL-CONNECT confirm primitive has been removed from the DLS-user B to DLS-user A abstract queue.

In the abstract queue from DLS-user B to DLS-user A, objects other than a disconnect object can be entered only after DLS-user B has entered a connect object representing a DL-CONNECT response primitive, and has received a subsequent DL-CONNECTION-ESTABLISHED indication primitive. A disconnect object can be entered at any time.

While a peer DLC exists, the properties exhibited by the abstract queues represent the agreements concerning QoS reached among the DLS-users and the DLS-provider during this DLC establishment procedure.

6.2.2.3 Data transfer

Flow control on the peer DLC is represented in this queue model by the management of the queue capacity, allowing objects to be added to the queues. The addition of an object may prevent the addition of a further object.

Once objects are in the queue, the DLS-provider may manipulate pairs of adjacent objects, resulting in deletion. An object may be deleted if, and only if, the object that follows it is defined to be destructive with respect to the object. If necessary, the last object on the queue will be deleted to allow a destructive object to be entered — they may therefore always be added to the queue. Disconnect objects are defined to be destructive with respect to all other objects. Reset objects are defined to be destructive with respect to all other objects except connect and disconnect objects.

The relationships between objects that may be manipulated in the above fashion are summarized in Table 9.

Table 9 – Relationships between abstract queue model objects

The following (column) object is defined with respect to the preceding (row) object	Connect	Data	Reset	Synchronization mark	Disconnect
Connect	N/A	—	—	N/A	DES
Data	N/A	—	DES	N/A	DES
Reset	N/A	—	DES	—	DES
Synchronization Mark	N/A	—	DES	N/A	DES
Disconnect	N/A	N/A	N/A	N/A	DES

Where

N/A	: X will not precede Y in a valid state of a queue
—	: not to be destructive nor to be able to advance ahead
DES	: to be destructive to the preceding object.

Whether the DLS-provider performs actions resulting in deletion or not will depend upon the behavior of the peer DLC users and the agreed QoS for the peer DLC. With few exceptions, if a DLS-user does not remove objects from an abstract queue, the DLS-provider should, after some unspecified time, perform all the permitted deletions.

6.2.2.4 Peer DLC / DLCEP reset

To model the reset service accurately, a synchronization mark object is required. The synchronization mark object exhibits the following characteristics:

a) It cannot be removed from an abstract queue by a DLS-user.

b) An abstract queue appears empty to a DLS-user when a synchronization mark object is the next object in the queue.

c) A synchronization mark can be destroyed by a disconnect object (see Table 9).

d) When a reset object is immediately preceded by a synchronization mark object, both the reset object and the synchronization mark object are deleted from the abstract queue.

The initiation of a reset procedure is represented in the two abstract queues as follows:

1) Initiation of a reset procedure by the DLS-provider is represented by the introduction into each abstract queue of a reset object followed by a synchronization mark object.

2) A reset procedure initiated by the DLS-user is represented by the addition, by the DLS-provider, of objects into both abstract queues:

   i) from the reset requestor to the peer DLS-user of a reset object; and

   ii) from the peer DLS-user to the reset requestor of a reset object followed by a synchronization mark object.

NOTE 1 The DLS-provider is always able to add a synchronization mark object following a reset object.

Unless destroyed by a disconnect object, a synchronization mark object remains in the abstract queue until the next object following it in the abstract queue is a reset object. Then the DLS-provider deletes both the synchronization mark object and the following reset object.

NOTE 2 Restrictions on the issuance of certain other types of DL-primitives are associated with the initiation of a reset procedure. These restrictions result in constraints on the entry of certain object types into the abstract queue until the reset procedure is completed (see 6.7.3.3).

6.2.2.5 Peer DLC / DLCEP release

The insertion into an abstract queue of a disconnect object, which may occur at any time, represents the initiation of a peer DLC release procedure. The release procedure may be destructive with respect to other objects in the two queues. The release procedure eventually results in the emptying of the queues and the disassociation of the queues from the peer DLC.

The insertion of a disconnect object may also represent the rejection of a peer DLC establishment attempt or the failure to complete peer DLC establishment. In such cases, if a connect object representing a DL-CONNECT request primitive is deleted by a disconnect object, then the disconnect object is also deleted. The disconnect object is not deleted when it deletes any other object, including the case where it deletes a connect object representing a DL-CONNECT response.

6.2.3 Model of a multi-peer DLC

Between the publishing and subscribing end-points of a multi-peer DLC, during the process of DLC establishment, there exists a relationship between the behavior of the responding DLS-user(s) and that of the initiating DLS-user. After DLC establishment, there exists a relationship between the publishing DLS-user and the subscribing DLS-user(s). As a means of specifying these relationships, the OSI abstract queue model of a multi-peer DLC, which is described in the following paragraphs, is used.

NOTE 1 The abstract queues referred to in this model are used solely for describing the interactions at the DLCEPs between the DLC users and the DLC provider. They have no relationship to actual queues managed by the primitives of Clause 5.

This abstract queue model of a multi-peer DLC is discussed only to aid in the understanding of the end-to-end DLS features perceived by DLS-users. It is not intended to serve as a substitute for a precise, formal description of the DLS, nor as a complete specification of all allowable sequences of DLS primitives. (Allowable primitive sequences are specified in 6.4. See also the note below.) In addition, this model does not attempt to describe all the functions or operations of DLEs that are used to provide the DLS. No attempt to specify or constrain DLE implementation is implied.

NOTE 2 The internal mechanisms that support the operation of the DLS are not visible to the DLS-user. In addition to the interactions between service primitives described by this model (for example, the issue of a DL-DISCONNECT request at a receiving DLSAP will prevent the receipt of a DL-DATA indication, corresponding to a previously issued DL-DATA request, by the sending DLS-user) there may also be:

a) constraints applied locally on the ability to invoke DLS-primitives; and

b) service procedures defining particular sequencing constraints on some DLS-primitives.

6.2.3.1 OSI abstract queue model concepts

The OSI abstract queue model represents the operation of a multi-peer DLC in the abstract by a set of abstract queues linking the publishing DLCEP with the subscribing DLCEPs — one queue per receiving DLCEP (see Figure 8).

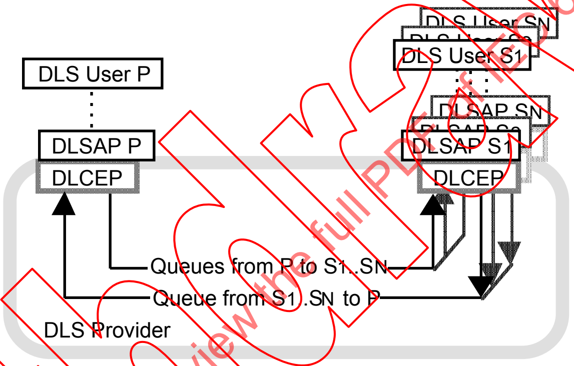


Figure 8 – OSI abstract queue model of a multi-peer DLC between a publishing DLS-user and a set of subscribing DLS-users

Each OSI abstract queue represents one direction of transfer. The ability of a subscribing DLS-user to remove objects from an abstract queue will be determined by the behavior of the publishing DLS-user. The ability of a publishing DLS-user to remove objects from an abstract queue will be determined by the aggregate behavior of the set of the publishing DLS-user and the subscribing DLS-users.

The following objects may be placed in an abstract queue by any sending DLS-user:

a) a connect object, representing a DL-CONNECT primitive and its parameters; and

b) a data object, representing a DL-DATA primitive and its parameters.

The following objects may be placed in an abstract queue by the publishing DLS-user:

c) a reset object, representing a DL-RESET primitive and its parameters; and

d) a disconnect object, representing a DL-DISCONNECT primitive and its parameters.

The publishing DLS-user may place a connect object in either a single abstract queue or simultaneously in the entire set of publisher-to-subscriber abstract queues. The publisher places all other objects simultaneously in the entire set of publisher-to-subscriber abstract queues. Each subscriber places all objects in the single subscribers-to-publisher abstract queue.

The following objects may be placed in an abstract queue by the DLS-provider:

1) a reset object, representing a DL-RESET primitive and its parameters; and
2) a disconnect object, representing a DL-DISCONNECT primitive and its parameters.

The abstract queues are defined to have the following general properties:

i) An abstract queue is empty before a connect object has been entered, and can be returned to this state, with loss of its contents, by the DLS-provider.
ii) Objects are entered into an abstract queue by the sending DLS-user, subject to control by the DLS-provider. Objects may also be entered by the DLS-provider.
iii) Objects are removed from the abstract queue under the control of the receiving DLS-user.
iv) Objects are normally removed in the same order that they were entered (however, see 6.2.3.3).
v) An abstract queue has a limited capacity, but this capacity is not necessarily either fixed or determinable. When the sending DLS-user places a new object in an abstract queue that has reached its capacity, the oldest object in the abstract queue is lost.
vi) Depending on the multi-peer DLC's QoS, the abstract queue may occasionally duplicate or lose data objects.

6.2.3.2 Multi-peer DLC / DLCEP establishment

When the DLS-provider receives a DL-CONNECT request primitive at one of the DLSAPs, it associates a pair of abstract queues and an abstract multi-peer DLC between the two DLSAPs, and enters either a connect object or a disconnect object into the appropriate abstract queue. From the standpoint of the DLS-users of the multi-peer DLC, the abstract queues remain associated with the multi-peer DLC until a disconnect object representing a DL-DISCONNECT primitive is either entered into or removed from the abstract queue.

The publishing DLS-user, which initiates a multi-peer DLC establishment by entering a connect object representing a DL-CONNECT request primitive into each of the separate publisher-to-subscriber abstract queues, is not allowed to enter any other object, other than a disconnect object, into the abstract queues until after the connect object representing the DL-CONNECT confirm primitive has been removed from the merged subscribers-to-publisher abstract queue. A disconnect object can be entered at any time.

A subscribing DLS-user, which initiates a multi-peer DLCEP establishment by entering a connect object representing a DL-CONNECT request primitive into the subscribers-to-publisher abstract queue, is not allowed to enter any other object, other than a disconnect object, into the abstract queue until after the connect object representing the DL-CONNECT confirm primitive has been removed from its publisher-to-subscriber abstract queue. In the subscriber-specific abstract queue from the publisher to that subscriber, objects other than a disconnect object can be entered only after the publishing DLS-user has entered a connect object representing a DL-CONNECT response primitive. A disconnect object can be entered at any time.

While the multi-peer DLC exists between the publisher and a specific subscriber, the properties exhibited by the two abstract queues between them represent the agreements concerning QoS reached between the publishing DLS-user, that subscribing DLS-user, and the DLS-provider during the DLCEP establishment procedure.

6.2.3.3 Data transfer

Flow control on the multi-peer DLC is represented in this abstract queue model by the management of the abstract queue capacity, allowing objects to be added to the abstract queues. The addition of an object may prevent the addition of a further object, or may cause the loss of the first data object in the abstract queue. The addition of a disconnect object to the abstract queue may cause the loss of all prior objects still in the abstract queue. The ordering of objects in the abstract queue is identical to that implied by Table 9.

Whether the DLS-provider performs actions resulting in deletion or not will depend upon the behavior of the multi-peer DLC's users. With few exceptions, if a subscribing DLS-user does not remove objects from an abstract queue at the same long-term rate as the publishing DLS-user places them in the abstract queue, then some objects will be lost.

6.2.3.4 Multi-peer DLC/DLCEP reset

The initiation of a reset procedure is represented in the abstract queues as follows:

a) initiation of a reset procedure by the DLS-provider is represented by the introduction of a reset object into one or more abstract queues;
b) a reset procedure initiated by the publishing DLS-user is represented by the addition, by the DLS-provider, of a reset object into each publisher-to-subscriber abstract queue;
c) a reset procedure initiated by a subscribing DLS-user is represented by the addition, by the DLS-provider, of a reset object into the subscribers-to-publisher abstract queue at that subscriber.

NOTE Restrictions on the issuance of certain other types of DL-primitives are associated with the initiation of a reset procedure. These restrictions will result in constraints on the entry of certain object types into the abstract queue until the reset procedure is completed (see 6.7.3.3).

6.2.3.5 Multi-peer DLC subscriber query

At any time during the existence of a multi-peer DLC, its publishing DLS-user can query whether there are any subscribing DLS-users.

NOTE This query does not result in the identification of any of these receiving DLS-users.

This query occurs within the scope of the multi-peer DLC, and thus cannot precede DLC establishment, nor can either the query or its confirmation follow DLC release. In all other respects, this query is asynchronous to the other activities of the multi-peer DLC.

6.2.3.6 Multi-peer DLC/DLCEP release

The insertion into an abstract queue of a disconnect object, which may occur at any time, represents the initiation of a DLCEP, and possibly a DLC, release procedure. The release procedure may be destructive with respect to other objects in the queues. The release procedure eventually results in the emptying of the abstract queues and the disassociation of the abstract queues from the multi-peer DLC. For a release initiated by a subscribing DLS-user, only the publisher-to-subscriber abstract queue associated with that subscribing DLS-user is emptied and dissociated from the multi-peer DLC; the abstract queues associated with other DLS-users are unaffected.

The insertion of a disconnect object may also represent the rejection of a multi-peer DLC/DLCEP establishment attempt or the failure to complete multi-peer DLC/DLCEP establishment.

6.3 Quality of connection-mode service

The term quality-of-service (QoS) refers to certain characteristics of a DLC as observed between the connection end-points. QoS describes aspects of a DLC which are attributable solely to the DLS-provider. QoS can only be properly determined when DLS-user behavior does not constrain or impede the performance of the DLS.

Once a DLC is established, the DLS-users at the DLCEPs have the same knowledge and understanding of the actual QoS of the DLC.

6.3.1 Determination of QoS for connection-mode service

QoS is described in terms of QoS parameters. These parameters give DLS-users a method of specifying their needs, and give the DLS-provider a basis for protocol selection. All QoS parameters are selected on a per-connection basis during the establishment phase of a DLCEP, or of a DLC and DLCEP.

Some of the DLC's QoS parameters are defined in 4.3; others are defined in 6.3.2. The selection procedures for these parameters are described in detail in 6.5.2.3. Once the DLC is established, throughout the lifetime of the DLC, the agreed QoS values are not reselected at any point in time.

6.3.2 Definition of QoS parameters

QoS attributes are as follows.

6.3.2.1 DLCEP class

Each DLC/DLCEP establishment request specifies the class of the DLCEP. The three choices for DLCEP class are:

a) **PEER** — the DLS-user can exchange DLSDUs with one other peer DLS-user;

b) **PUBLISHER** — the DLS-user can send DLSDUs to a set of zero or more associated subscribing DLS-users, and may be able to receive DLSDUs from any of those subscribing DLS-users; or

c) **SUBSCRIBER** — the DLS-user can receive, and request, DLSDUs from the associated publishing DLS-user, and may be able to send DLSDUs to that publishing DLS-user.

The default QoS value for the DLCEP class is PEER.

6.3.2.2 DLCEP data delivery features

Both members of a peer DLC, or the publishing DLS-user of a multi-peer DLC, specify the data delivery features of the DLC's DLCEP(s). The available choices of DLCEP data delivery features depends on the class of DLS provided (see Table 10). The five choices for DLCEP data delivery features, and their effects, are:

a) **CLASSICAL** — the DLS-user can send data which will be delivered without loss, duplication or misordering to the receiving DLS-user(s). All relevant DLS-users will be notified of any loss of synchronization on the DLC.

b) **DISORDERED** — the DLS-user can send data which will be delivered immediately upon receipt to the receiving DLS-user(s), without duplication but potentially in a different order than the order in which it was originally sent. All relevant DLS-users will be notified of any unrecoverable loss of DLS-user-data or loss of synchronization on the DLC.

c) **ORDERED** — the DLS-user can send data which will be delivered immediately upon receipt to the receiving DLS-user(s), without duplication or misordering, but with potential loss of some DLS-user-data. Loss of DLS-user-data will not be reported, and recovery of DLS-user data lost prior to the last-reported DLS-user data will not be attempted.

NOTE Recovery of lost DLS-user data subsequent to that last reported is permitted but is not required.

d) **UNORDERED** — the DLS-user can send data which will be delivered immediately upon receipt to the receiving DLS-user(s). Loss, duplication and misordering of DLS-user-data will not be detected or reported. No attempt will be made by the DLS-provider to recover from such events.

e) **NONE** — the DLS-user cannot send data in this direction of data transfer.

There are four classes of connection-mode DLS as follows:

A CLASSICAL, ORDERED and UNORDERED peer and multi-peer DLCs are all supported; DISORDERED peer and multi-peer DLCs may be supported;

B only ORDERED and UNORDERED peer and multi-peer DLCs, and CLASSICAL peer DLCs, are supported; CLASSICAL and DISORDERED multi-peer DLCs are not supported; DISORDERED peer DLCs may be supported;

C only ORDERED and UNORDERED peer and multi-peer DLCs are supported; CLASSICAL and DISORDERED peer and multi-peer DLCs are not supported; and

D only UNORDERED peer and multi-peer DLCs are supported; ORDERED, CLASSICAL and DISORDERED peer and multi-peer DLCs are not supported.

Table 10 – Attributes and class requirements of DLCEP data delivery features

DLCEP data delivery features	Action on lost DLSDUs	Action on duplicate DLSDUs	Action on mis-ordered DLSDUs	Permitted forms of buffer and queue use	Support on a peer-to-peer DLC (note 4)	Support on a publisher-to-subscribers DLC (note 4)	Support on a subscribers-to-publisher DLC (note 4)
NONE	None	None	None	None	Mandatory in all classes	Mandatory in all classes	Mandatory in all classes
UNORDERED (default)	Not detected	Not detected	Not detected (note 2)	buffer ⇒ buffer buffer ⇒ queue queue ⇒ queue (note 3)	Mandatory in all classes	Mandatory in all classes	Optional in all classes
ORDERED	Detected	Discarded	Discarded	buffer ⇒ buffer buffer ⇒ queue queue ⇒ queue (note 3)	Mandatory in classes A,B,C	Mandatory in classes A,B,C	Not possible
CLASSICAL	Recovered (note 1)	Discarded	Recovered: delivered in order	queue ⇒ queue	Mandatory in classes A,B	Mandatory in class A	Not possible
DISORDERED	Recovered (note 1)	Discarded	Delivered as received	queue ⇒ queue	Optional in classes A,B	Optional in class A	Not possible

NOTE 1 An unrecoverable data loss causes the DLE to initiate a reset procedure.

NOTE 2 DL-communication topologies in which misordering is possible cause UNORDERED DLCs to be upgraded to ORDERED during DLC negotiation.

NOTE 3 Queue-to-queue use provides a QoS similar to connectionless service.

NOTE 4 The DLE should support all DLCEP data-delivery features required of its DLS class. (The DLE classes are introduced in 6.1.) If the DLE does not support DISORDERED but does support CLASSICAL, then the DLE upgrades the DLCEP data delivery features from DISORDERED to CLASSICAL. In all other cases, if the DLE does not support a requested DLCEP data delivery feature which is optional for its DLS class, then the DLE rejects the DLC/DLCEP establishment request.

On a peer DLC, the QoS value for the DLCEP data delivery features may be chosen independently for each direction of data transfer. The default QoS value in each direction for the DLCEP data delivery features is UNORDERED.

On a multi-peer DLC, the QoS value for the DLCEP data delivery features for the subscribers-to-publisher direction of data transfer is restricted to UNORDERED and NONE. The default QoS value for the DLCEP data delivery features in the publisher-to-subscribers direction is UNORDERED. The default QoS value for the DLCEP data delivery features in the subscribers-to-publisher direction is NONE.

Selection of any of the following QoS attribute values may cause the DLS-provider to upgrade DLCEP data delivery features from UNORDERED to ORDERED:

1) specification of a maximum DLSDU size (see 6.3.2.8) greater than the maximum number of DLS-user octets that can be conveyed by a DLPDU of the specified DLL priority (see 6.3.2.3); or

2) specification of DL-timeliness criteria (see 6.3.2.10) other than none for transmissions from a peer or publisher DLCEP; or

3) selection of a DLCEP data delivery feature other than unordered or none for the other (reverse) direction of a peer DLCEP.

This upgrade may also occur when multiple active paths can exist between the DLC's participating DLS-providers.

The DLS provider may also upgrade from DISORDERED to CLASSICAL during this negotiation process.

No other negotiation of either of the DLCEP's data delivery features is permitted.

Table 10 summarizes these DLCEP data delivery features, their attributes, and the DLS classes in which they are available.

6.3.2.3 DLL priority

This parameter is defined and its default value specified in 4.3.1 and 5.4.3.2.6.1.

NOTE DLC initiation and DLC termination DLPDUs are sent at TIME-AVAILABLE priority on the link, but are restricted to convey no more DLS-user-data than is permitted for NORMAL priority DLPDUs.

6.3.2.4 DLL maximum confirm delay

This parameter is defined and its default values specified in 4.3.2 and 5.4.3.2.6.2. Failure to complete a DL-CONNECT or DL-RESET request within the specified interval results in a DLS-provider initiated release of the DLCEP, and possibly of the DLC.

NOTE For DLEs which do not support a time resolution of 1 ms, the requested time interval may be rounded up to the next-greatest multiple of the resolution which the DLE does support.

This parameter provides user-specifiable bounds on the extent of DLC error recovery effort.

6.3.2.5 DLPDU authentication

This parameter is defined and its default value is specified in 4.3.3 and 5.4.3.2.6.3. The DLS-user may override that default value when establishing a DLCEP.

6.3.2.6 Residual activity

Each DLC/DLCEP establishment request, and each response, specifies whether the current state of a sending peer or publisher DLCEP should be sent at low-frequency to the DLC's peer or subscriber DLCEP(s) even when there are no unconfirmed DLS-user requests outstanding at the sending DLCEP.

NOTE This background transmission is known as DLC **residual activity**.

The default value for this parameter is FALSE, unless overridden by DL-management or by a DLPDU-authentication attribute of MAXIMAL.

6.3.2.7 DL-scheduling-policy

This parameter is defined and its default value is specified in 4.3.4 and 5.4.3.2.6.4. The DLS-user may override that default value when establishing a DLCEP. When the DLCEP is bound as sender to a buffer, then the DLS-user should ensure that this parameter has the value EXPLICIT.

6.3.2.8 Maximum DLSDU sizes

Each DLC/DLCEP establishment request, and each response, specifies an upper bound on the size (in octets) of DLSDUs which will be offered for transmission, and an upper bound on the size of DLSDUs which are acceptable for reception. For peer DLCs, the maximum DLSDU size permitted will be the smallest of that offered by the sender, that permitted by the receiver, and that permitted by DLL management. For multi-peer DLCs, the maximum DLSDU size permitted will be the smaller of that offered by the publisher and that permitted by the publisher's DLL management. For subscribers of multi-peer DLCs, the DLC will be refused by the DLS-provider if the maximum DLSDU size established by the publisher is larger than

a) that permitted by the subscriber; or

b) that permitted by the subscriber's DLL management.

The default value for both the sender's and receiver's maximum DLSDU size is the maximum number of DLS-user octets which can be carried by a single DLPDU of the specified DLL Priority (see 4.3.1). The DLS-provider should always support that DLSDU size.

NOTE These negotiation rules do not preclude either derivative protocol specifications or real implementations from using a fixed, small set of sizes when allocating buffers or entries in a queue.

6.3.2.9 DLCEP buffer-or-queue bindings

Each DLCEP establishment request, and each response, can bind one or two local retentive buffers or specified-depth FIFO queues, created by DL-CREATE buffer and queue management primitives (or by DL-management), to the DLCEP.

NOTE When these bindings are made for a DLS-user of a peer DLC, or a publishing DLS-user of a multi-peer DLC, they determine the maximum transmit window (that is, number of transmitted but unacknowledged DLSDUs) for that direction of DLC data transfer. Since the size of the transmit window can also be limited by DL-management, or by an implementation, the queue depth only imposes an upper limit on the window size:

a) One queue or retentive buffer can be bound to a DLCEP to convey DLSDUs from the DLS-user to the DLS-provider.

b) One queue or retentive buffer can be bound to a DLCEP to convey DLSDUs from the DLS-provider to the DLS-user.

c) It is also possible to bind a queue or retentive buffer to be written at one DLCEP and to source data at another DLCEP. Such an intermediate buffer or queue can serve to cross scheduling boundaries or redistribute received DLS-user data to a second set of DLS-users.

Such a binding is made by specifying, for the appropriate parameter, a buffer-or-queue DL-identifier which resulted from a prior DL-CREATE request (or by DL-management), and which has not yet been deleted.

When the DLCEP's sending data delivery features specify UNORDERED or ORDERED, both the sender and receiver(s) may specify a queuing policy of BUFFER-R or QUEUE. When the DLCEP's sending data delivery features specify DISORDERED or CLASSICAL, both the sender and receiver(s) may specify a queuing policy of QUEUE; a queuing policy of BUFFER-R is not permitted. A queuing policy of BUFFER-NR is never permitted.

6.3.2.9.1 Binding to a retentive buffer

When a sending retentive buffer is bound to a DLCEP by a DLS-user:

a) A DL-PUT request primitive overwrites the buffer with a DLSDU.

NOTE After creation, the buffer is empty. After it has once been written, a buffer bound to a DLCEP can never again be empty.

b) A DL-BUFFER-SENT indication primitive notifies the DLS-user of the specific DLCEP on which the buffer was transmitted, and to which the buffer is bound, that the buffer was just transmitted.

c) A DL-COMPEL request primitive causes the transmission, at the first opportunity, of the contents of the buffer at the moment of transmission; the primitive does not itself specify a DLSDU. As a consequence, the only meaningful scheduling policy for buffers is EXPLICIT.

When a receiving retentive buffer is bound to a DLCEP by a DLS-user:

d) A DL-BUFFER-RECEIVED indication primitive notifies the DLS-user of the overwriting of the buffer by a newly-received DLSDU; the primitive does not itself specify a DLSDU.

e) A DL-GET request primitive copies the DLSDU from the buffer.

6.3.2.9.2 Binding to a FIFO queue

When a sending FIFO queue of maximum depth *K* is bound to a DLCEP by a DLS-user:

a) A DL-PUT request primitive is not permitted.

b) A DL-DATA request primitive attempts to append a DLSDU to the queue, but fails if the queue already contains *K* DLSDUs. If the append operation was successful, then the DLSDU will be transmitted at the first opportunity, after all preceding DLSDUs in the queue.

When a receiving FIFO queue of maximum depth *K* is bound to a DLCEP by a DLS-user

c) A DL-GET request primitive attempts to remove a DLSDU from the queue, but fails if the queue is empty.

d) A DL-DATA indication primitive notifies the receiving DLS-user of the result of appending a newly-received DLSDU to the receive queue; the primitive does not itself specify a DLSDU.

6.3.2.9.3 Default bindings

When these binding options are not specified, the conventional implicitly-queued sending and direct receiving interfaces between DLS-user and DLS-provider are employed, and each associated DL-DATA primitive contains a DLSDU:

a) DL-Put and DL-Get request primitives are not permitted.

b) A DL-Data request primitive by the sending DLS-user attempts to append a DLSDU to the implicit queue, but fails if the queue is full. If the append operation was successful, then the DLSDU will be transmitted at the first opportunity, after all preceding DLSDUs in the queue.

c) A DL-Data indication primitive notifies a receiving DLS-user of a newly-received DLSDU, and conveys that DLSDU to the DLS-user. No apparent queuing is provided within the DLL.

6.3.2.10 DLCEP timeliness

Each DLCEP establishment request, and each response, can specify DL-timeliness QoS criteria which are to apply to information sent or received at that DLCEP. (See 4.3.5.)

6.3.2.10.1 Sender timeliness

This set of sub-parameters applies only when the buffer-and-queue-bindings-as-sender specify a retentive buffer.

6.3.2.10.1.1 DL-timeliness class

The DLS-user should specify one of the four types of DL-timeliness specified in 4.3.5 — RESIDENCE, UPDATE, SYNCHRONIZED, or TRANSPARENT — or should specify NONE, indicating that DL-timeliness is not to be provided.

The default value for this parameter is NONE.

6.3.2.10.1.2 Time window size (ΔT)

When the value for the DL-timeliness-class parameter is RESIDENCE, UPDATE or SYNCHRONIZED, then the DLS-user should specify the applicable time window size (ΔT), with a permitted maximum of 60 s.

6.3.2.10.1.3 Synchronizing DLCEP

When the value for the DL-timeliness-class parameter is UPDATE or SYNCHRONIZED, then the DLS-user should specify the DL-identifier (see 6.5.2.1.2) of the DLCEP whose activity (DL-BUFFER-SENT indication or DL-BUFFER-RECEIVED indication (see 6.7.2) is to be used as the synchronizing activity for the timeliness computation. If the synchronizing DLCEP disconnects before the referencing DLCEP, then after that disconnection the resulting sender timeliness is FALSE.

6.3.2.10.1.4 Time of production

When the value for the DL-timeliness-class parameter is RESIDENCE, UPDATE, SYNCHRONIZED or TRANSPARENT, then the DLS-user may specify that the DL-time of DLS-user-data production is to be distributed to the consuming peer DLS-users to facilitate their assessment of DLS-user-timeliness. Permitted values for this parameter are TRUE and FALSE.

The default value for this parameter is FALSE.

6.3.2.10.1.5 Receiver timeliness

This set of sub-parameters applies only when the buffer-and-queue-bindings-as-receiver specify a retentive buffer.

6.3.2.10.1.6 DL-timeliness class

The DLS-user should specify one of the four types of DL-timeliness specified in 4.3.5 — RESIDENCE, UPDATE, SYNCHRONIZED, or TRANSPARENT — or should specify NONE, indicating that DL-timeliness is not to be provided.

The default value for this parameter is NONE.

6.3.2.10.1.7 Time window size (ΔT)

When the value for the DL-timeliness-class parameter is RESIDENCE, UPDATE or SYNCHRONIZED, then the DLS-user should specify the applicable time window size (ΔT), with a permitted maximum of 60 s.

6.3.2.10.1.8 Synchronizing DLCEP

When the value for the DL-timeliness-class parameter is UPDATE or SYNCHRONIZED, then the DLS-user should specify the DL-identifier (see 6.5.2.1.2) of the DLCEP whose activity (DL-BUFFER-SENT indication or DL-BUFFER-RECEIVED indication (see 6.7.2) is to be used as the synchronizing activity for the timeliness computation. If the synchronizing DLCEP disconnects before the referencing DLCEP, then after that disconnection the resulting receiver timeliness is FALSE.

6.4 Sequence of primitives

6.4.1 Concepts used to define the connection-mode DLS

The service definition uses the following concepts:

a) A DLC can be dynamically established (or terminated) between the DLS-users for the purpose of exchanging data. It is also possible statically to pre-establish an unordered or ordered DLC.

b) Associated with each DLC, certain measures of QoS are agreed between the DLS-provider and the DLS-users when the DLC is established.

c) The DLC allows transmission of DLS-user-data and preserves the data's division into DLSDUs.

  — The transmission of this data may be subject to flow control, depending on the DLCEP class and data delivery features.

  — The transmission and reception of this data may also be subject to timeliness assessments, which are combined with any previously-determined data timeliness to determine the overall timeliness of the DLS-user-data.

d) The DLC can be returned to a defined state, and the activities of either or both of the two DLS-users synchronized, by use of a reset-service.

e) Failure to provide the requested service may be signaled to the DLS-user. There are three classes of failure:

   1) failures involving termination of the DLC;

   2) failures involving duplication, loss or disordering of user data, as permitted by the DLC's QoS, but without loss of the DLC; and

      NOTE This includes resetting of the DLC.

   3) failures to provide the requested QoS, but without loss of user data or loss of the DLC.

6.4.2 Constraints on sequence of primitives

This subclause defines the constraints on the sequence in which the primitives defined in 6.5 through 6.7 may occur. The constraints determine the order in which primitives occur, but do not fully specify when they may occur. Other constraints, such as flow control of data, will affect the ability of a DLS-user or a DLS-provider to issue a primitive at any particular time.

The connection-mode primitives and their parameters are summarized in Table 11 and Table 12.

Table 11 – Summary of DL-connection-mode primitives and parameters (portion 1)

Phase	Service	Primitive		Parameter	
DLC / DLCEP Establishment	DLC / DLCEP Establishment	DL-CONNECT request	(*in*	DLCEP DLS-user-identifier, Called DL(SAP)-address or Called DLCEP-address or UNKNOWN, Calling DLSAP-address DL-identifier or Calling DLCEP DL-identifier, optional Calling DLCEP-address, QoS parameter set, limited DLS-user-data,	(2)
			out	DLCEP DL-identifier)	
		DL-CONNECT indication	(*out*	DLCEP DL-identifier, Called DL(SAP)-address DLS-user-identifier, Calling DLSAP-address, QoS parameter set, limited DLS-user-data)	(2)
		DL-CONNECT response	(*in*	DLCEP DLS-user-identifier, Responding DL(SAP)-address DL-identifier or Responding DLCEP DL-identifier, optional Responding DLCEP-address, QoS parameter set, limited DLS-user-data)	(2)
		DL-CONNECT confirm	(*out*	Responding DL(SAP)-address or UNKNOWN, QoS parameter set, limited DLS-user-data)	(2)
		DL-CONNECTION-ESTABLISHED indication	(*out*	DLCEP DLS-user-identifier)	
DLC / DLCEP Release	DLC / DLCEP Release	DL-DISCONNECT request	(*in*	DLCEP DL-identifier, Reason, limited DLS-user-data)	
		DL-DISCONNECT indication	(*out*	DLCEP-identifier-type, DLCEP DLS-user-identifier, DLCEP DL-identifier, Originator, Reason, limited DLS-user-data)	

Table 12 – Summary of DL-connection-mode primitives and parameters (portion 2)

Phase	Service	Primitive	Parameter
Data Transfer	Normal Data Transfer	DL-DATA request	(*in* DLCEP DL-identifier, DLS-user-data)
		DL-DATA indication	(*out* DLCEP DLS-user-identifier, Queue DLS-user-identifier, DLS-user-data)
		DL-DATA confirm	(*out* Status)
		DL-BUFFER-SENT indication	(*out* DLCEP DLS-user-identifier, Buffer DLS-user-identifier)
		DL-BUFFER-RECEIVED indication	(*out* DLCEP DLS-user-identifier, Buffer DLS-user-identifier, Duplicated DLSDU)
		DL-GET-NEXT-SEQUENCE-NUMBER request	(*in* DLCEP DL-identifier *out* Status, Sequence-number)
	DLC / DLCEP Reset / Resynchronize	DL-RESET request	(*in* DLCEP DL-identifier, Reason, limited DLS-user-data)
		DL-RESET indication	(*out* DLCEP DLS-user-identifier, Originator, Reason, limited DLS-user-data)
		DL-RESET response	(*in* DLCEP DL-identifier, limited DLS-user-data)
		DL-RESET confirm	(*out* limited DLS-user-data, Status)
		DL-RESET-COMPLETED indication	(*out* DLCEP DLS-user-identifier)
	Subscriber Query	DL-SUBSCRIBER-QUERY request	(*in* DLCEP DL-identifier)
		DL-SUBSCRIBER-QUERY confirm	(*out* Status)

NOTE 1 DL-identifiers in parameters are local and assigned by the DLS-provider and used by the DLS-user to designate a specific DL(SAP)-address, DLCEP, DL-buffer or DL-queue to the DLS-provider at the DLS interface. DLS-user-identifiers in parameters are local and assigned by the DLS-user and used by the DLS-provider to designate a specific DL(SAP)-address, DLCEP, DL-buffer or DL-queue to the DLS-user at the DLS interface.

NOTE 2 If the DLS-user has issued a DL-Bind request for a DL(SAP)-address, then a parameter referencing that DL(SAP)-address can also take the form of a DL(SAP)-address DL-identifier.

NOTE 3 The method by which a confirm primitive is correlated with its corresponding preceding request primitive, or a response primitive is correlated with its corresponding preceding indication primitive, is a local matter.

6.4.2.1 Relation of primitives at the two DLC end-points

With few exceptions, a primitive issued at one DLCEP will have consequences at another DLCEP. The relations of primitives at one DLCEP to primitives at another DLCEP of the same DLC are defined in the appropriate subclause in 6.5 through 6.7, and are summarized in the diagrams of Figure 9 through Figure 14.

However, a DL-DISCONNECT request or indication primitive may terminate any of the other sequences before completion. A DL-RESET request or indication primitive may terminate a data transfer sequence before completion.

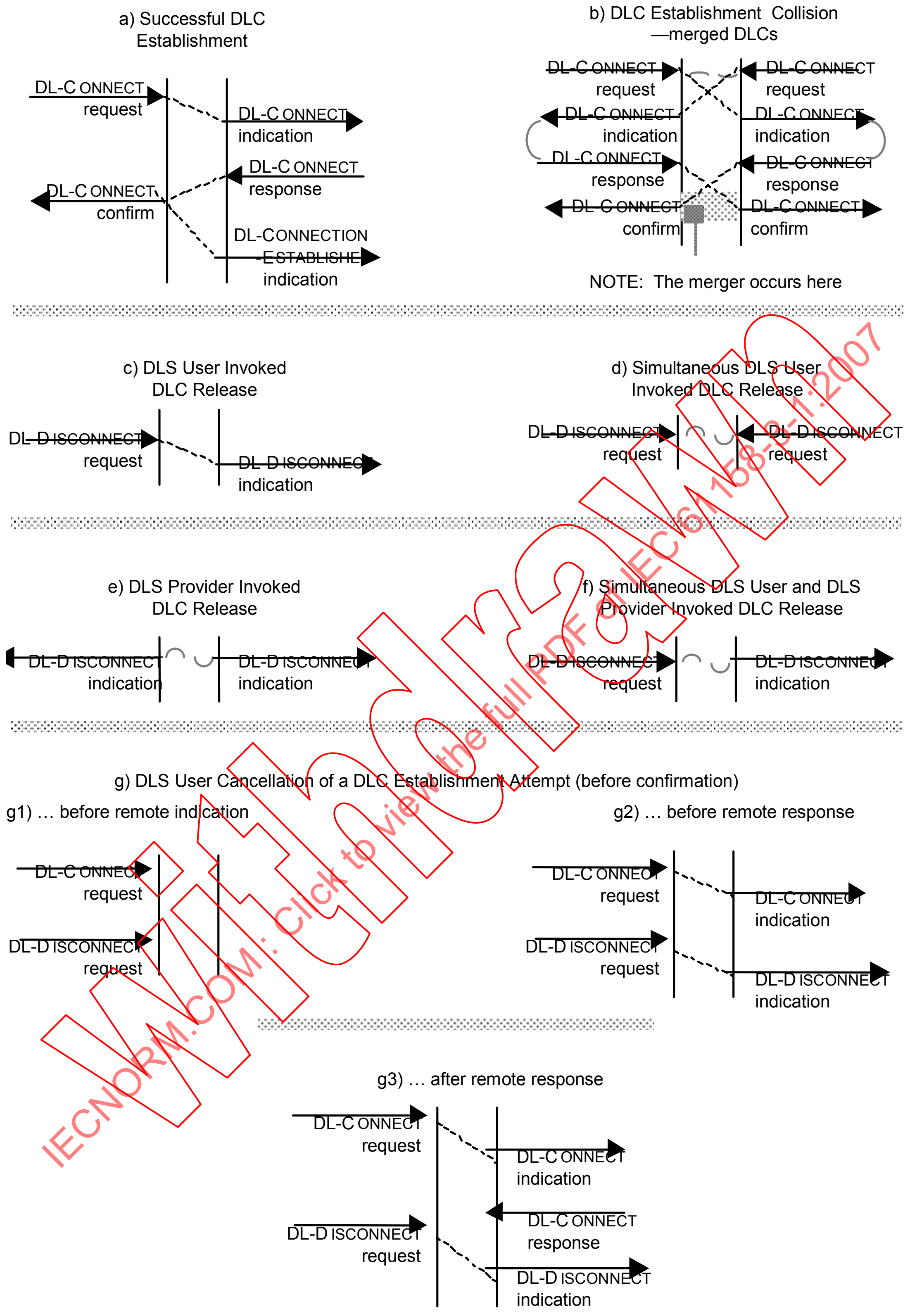


Figure 9 – Summary of DL-connection-mode service primitive time-sequence diagrams for peer DLCs (portion 1)

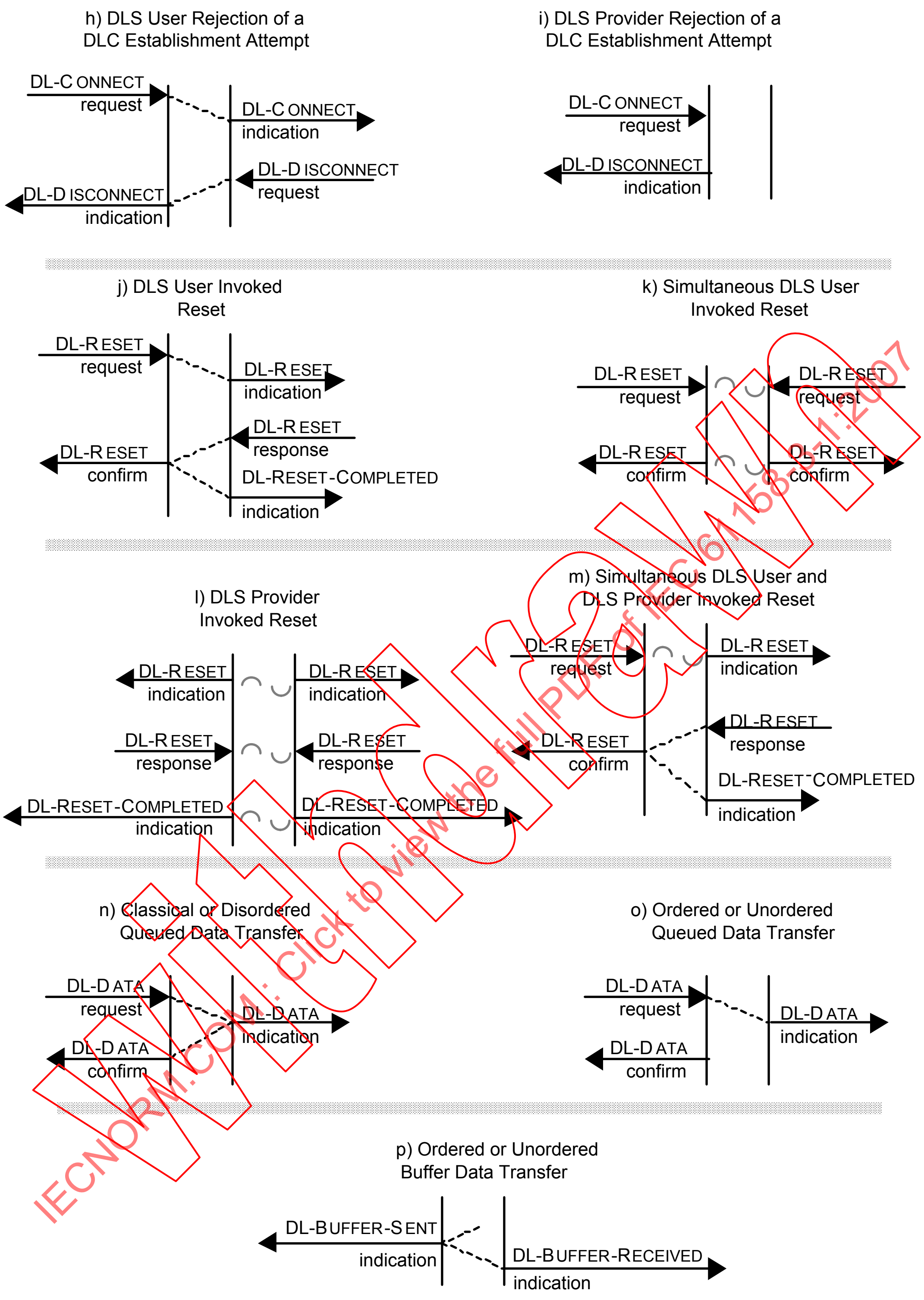

h) DLS User Rejection of a DLC Establishment Attempt
DL-CONNECT request
DL-CONNECT indication
DL-DISCONNECT request
DL-DISCONNECT indication
i) DLS Provider Rejection of a DLC Establishment Attempt
DL-CONNECT request
DL-DISCONNECT indication
j) DLS User Invoked Reset
DL-RESET request
DL-RESET indication
DL-RESET response
DL-RESET confirm
DL-RESET-COMPLETED indication
k) Simultaneous DLS User Invoked Reset
DL-RESET request
DL-RESET request
DL-RESET confirm
DL-RESET confirm
l) DLS Provider Invoked Reset
DL-RESET indication
DL-RESET indication
DL-RESET response
DL-RESET response
DL-RESET-COMPLETED indication
DL-RESET-COMPLETED indication
m) Simultaneous DLS User and DLS Provider Invoked Reset
DL-RESET request
DL-RESET indication
DL-RESET response
DL-RESET confirm
DL-RESET-COMPLETED indication
n) Classical or Disordered Queued Data Transfer
DL-DATA request
DL-DATA indication
DL-DATA confirm
o) Ordered or Unordered Queued Data Transfer
DL-DATA request
DL-DATA indication
DL-DATA confirm
p) Ordered or Unordered Buffer Data Transfer
DL-BUFFER-SENT indication
DL-BUFFER-RECEIVED indication

Figure 10 – Summary of DL-connection-mode service primitive time-sequence diagrams for peer DLCs (portion 2)

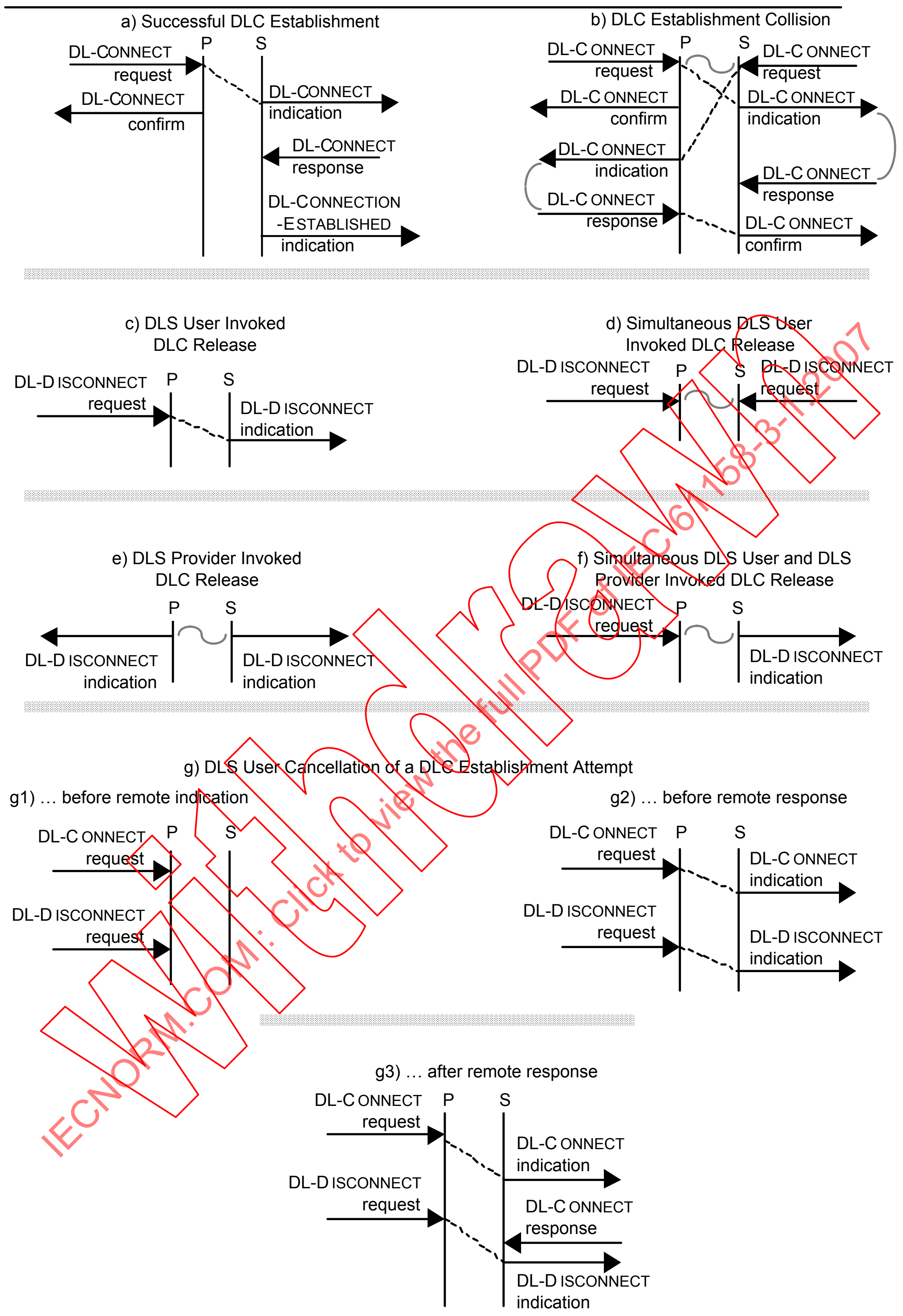


Figure 11 – Summary of DL-connection-mode service primitive time-sequence diagrams for publishers of a multi-peer DLC (portion 1)

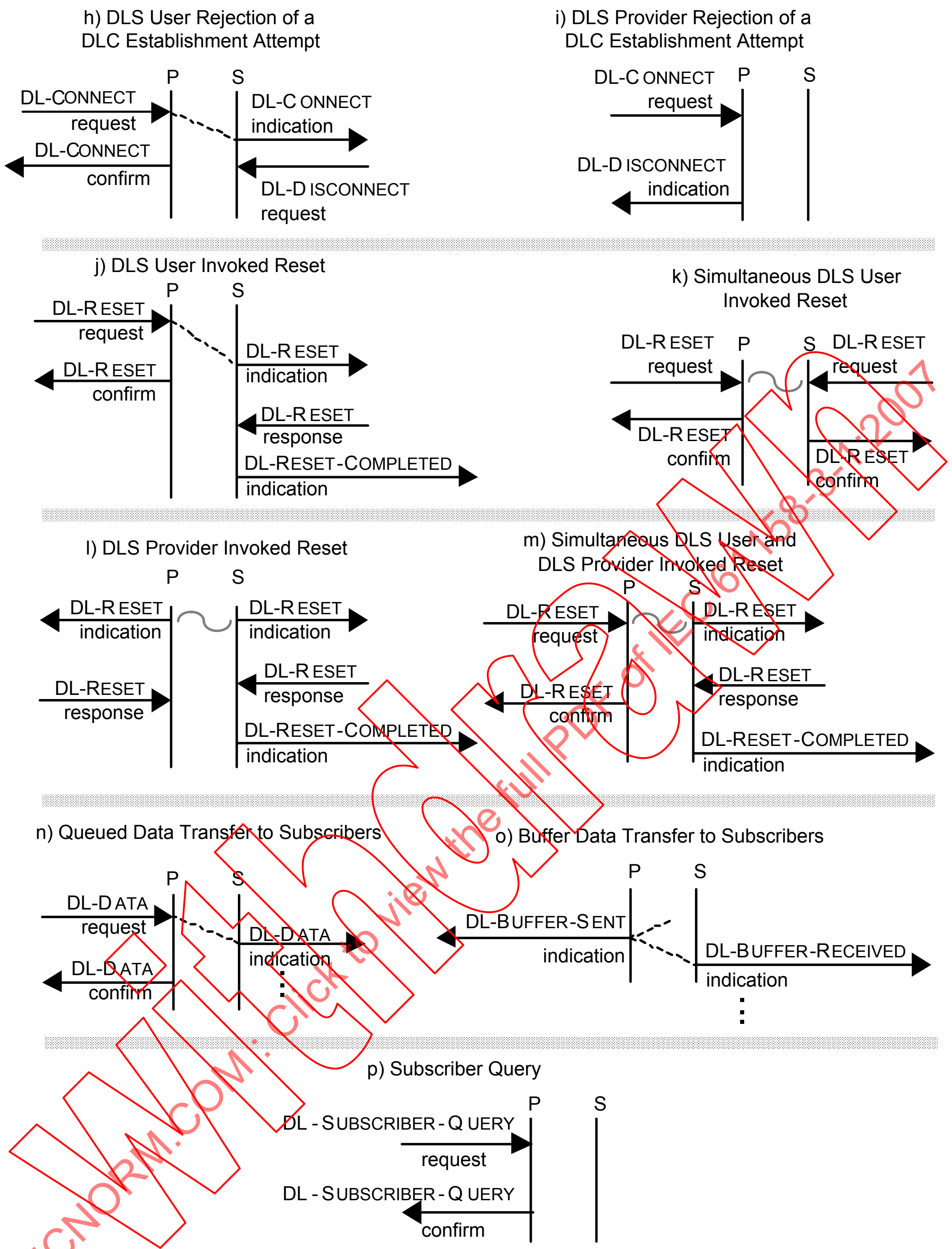

h) DLS User Rejection of a DLC Establishment Attempt
P
S
DL-CONNECT request
DL-CONNECT indication
DL-CONNECT confirm
DL-DISCONNECT request
i) DLS Provider Rejection of a DLC Establishment Attempt
DL-CONNECT request
P
S
DL-DISCONNECT indication
j) DLS User Invoked Reset
DL-RESET request
DL-RESET indication
DL-RESET confirm
DL-RESET response
DL-RESET-COMPLETED indication
k) Simultaneous DLS User Invoked Reset
DL-RESET request
DL-RESET request
DL-RESET confirm
DL-RESET confirm
l) DLS Provider Invoked Reset
DL-RESET indication
DL-RESET indication
DL-RESET response
DL-RESET response
DL-RESET-COMPLETED indication
m) Simultaneous DLS User and DLS Provider Invoked Reset
DL-RESET request
DL-RESET indication
DL-RESET response
DL-RESET confirm
DL-RESET-COMPLETED indication
n) Queued Data Transfer to Subscribers
DL-DATA request
DL-DATA indication
DL-DATA confirm
o) Buffer Data Transfer to Subscribers
DL-BUFFER-SENT indication
DL-BUFFER-RECEIVED indication
p) Subscriber Query
DL-SUBSCRIBER-QUERY request
DL-SUBSCRIBER-QUERY confirm

Figure 12 – Summary of DL-connection-mode service primitive time-sequence diagrams for publishers of a multi-peer DLC (portion 2)

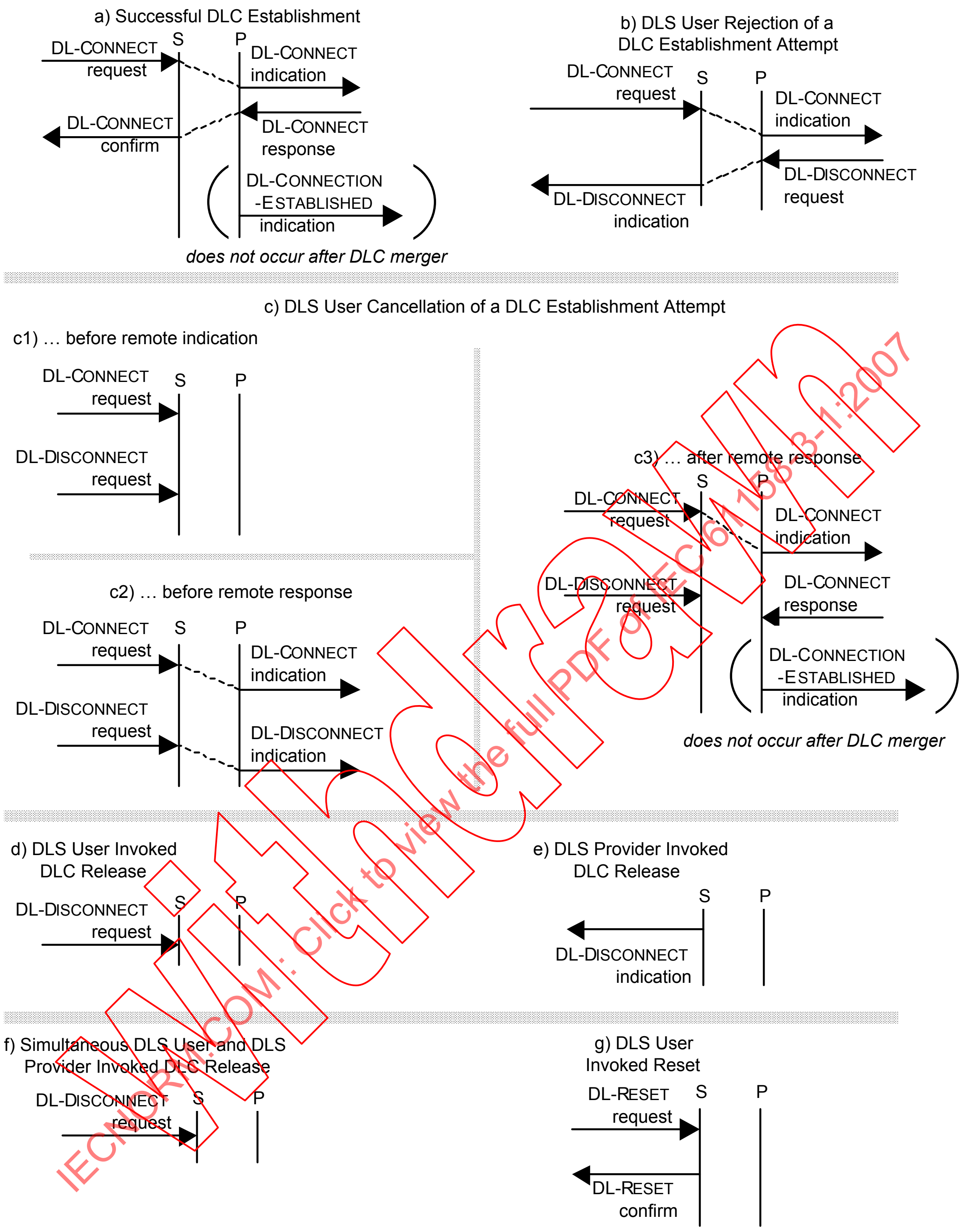

a) Successful DLC Establishment
S
P
DL-CONNECT request
DL-CONNECT indication
DL-CONNECT response
DL-CONNECT confirm
DL-CONNECTION -ESTABLISHED indication
does not occur after DLC merger
b) DLS User Rejection of a DLC Establishment Attempt
DL-CONNECT request
DL-CONNECT indication
DL-DISCONNECT request
DL-DISCONNECT indication
c) DLS User Cancellation of a DLC Establishment Attempt
c1) … before remote indication
DL-CONNECT request
DL-DISCONNECT request
c2) … before remote response
DL-CONNECT request
DL-CONNECT indication
DL-DISCONNECT request
DL-DISCONNECT indication
c3) … after remote response
DL-CONNECT request
DL-CONNECT indication
DL-DISCONNECT request
DL-CONNECT response
DL-CONNECTION -ESTABLISHED indication
does not occur after DLC merger
d) DLS User Invoked DLC Release
DL-DISCONNECT request
e) DLS Provider Invoked DLC Release
DL-DISCONNECT indication
f) Simultaneous DLS User and DLS Provider Invoked DLC Release
DL-DISCONNECT request
g) DLS User Invoked Reset
DL-RESET request
DL-RESET confirm

Figure 13 – Summary of additional DL-connection-mode service primitive time-sequence diagrams for a multi-peer DLC subscriber where the diagrams differ from the corresponding ones for a publisher (portion 1)

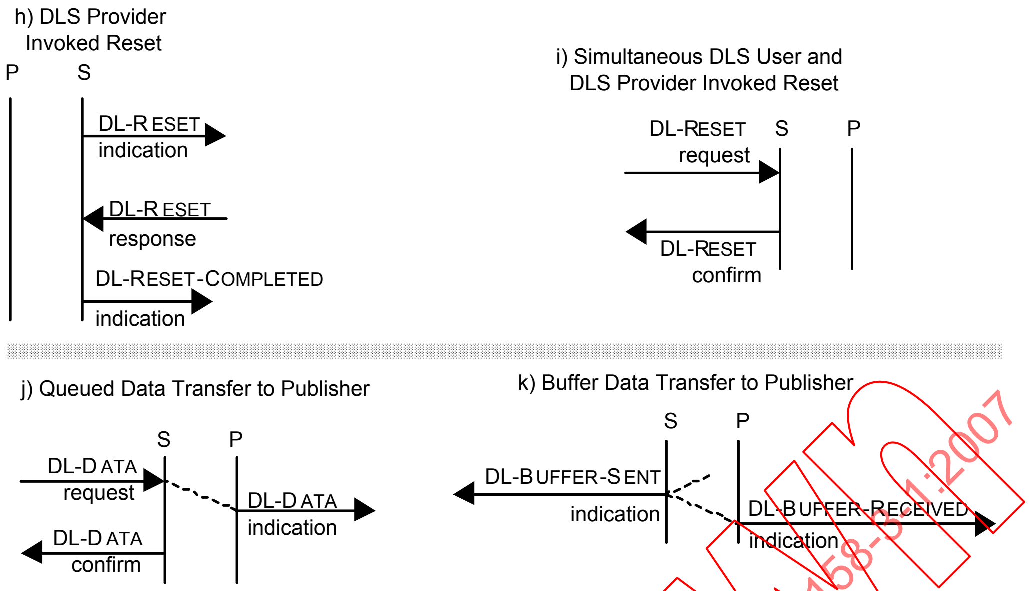

h) DLS Provider Invoked Reset
P
S
DL-RESET indication
DL-RESET response
DL-RESET-COMPLETED indication
i) Simultaneous DLS User and DLS Provider Invoked Reset
DL-RESET request
S
P
DL-RESET confirm
j) Queued Data Transfer to Publisher
S
P
DL-DATA request
DL-DATA confirm
DL-DATA indication
k) Buffer Data Transfer to Publisher
S
P
DL-BUFFER-SENT indication
DL-BUFFER-RECEIVED indication

Figure 14 – Summary of additional DL-connection-mode service primitive time-sequence diagrams for a multi-peer DLC subscriber where the diagrams differ from the corresponding ones for a publisher (portion 2)

6.4.2.2 Sequence of primitives at one DLC end-point

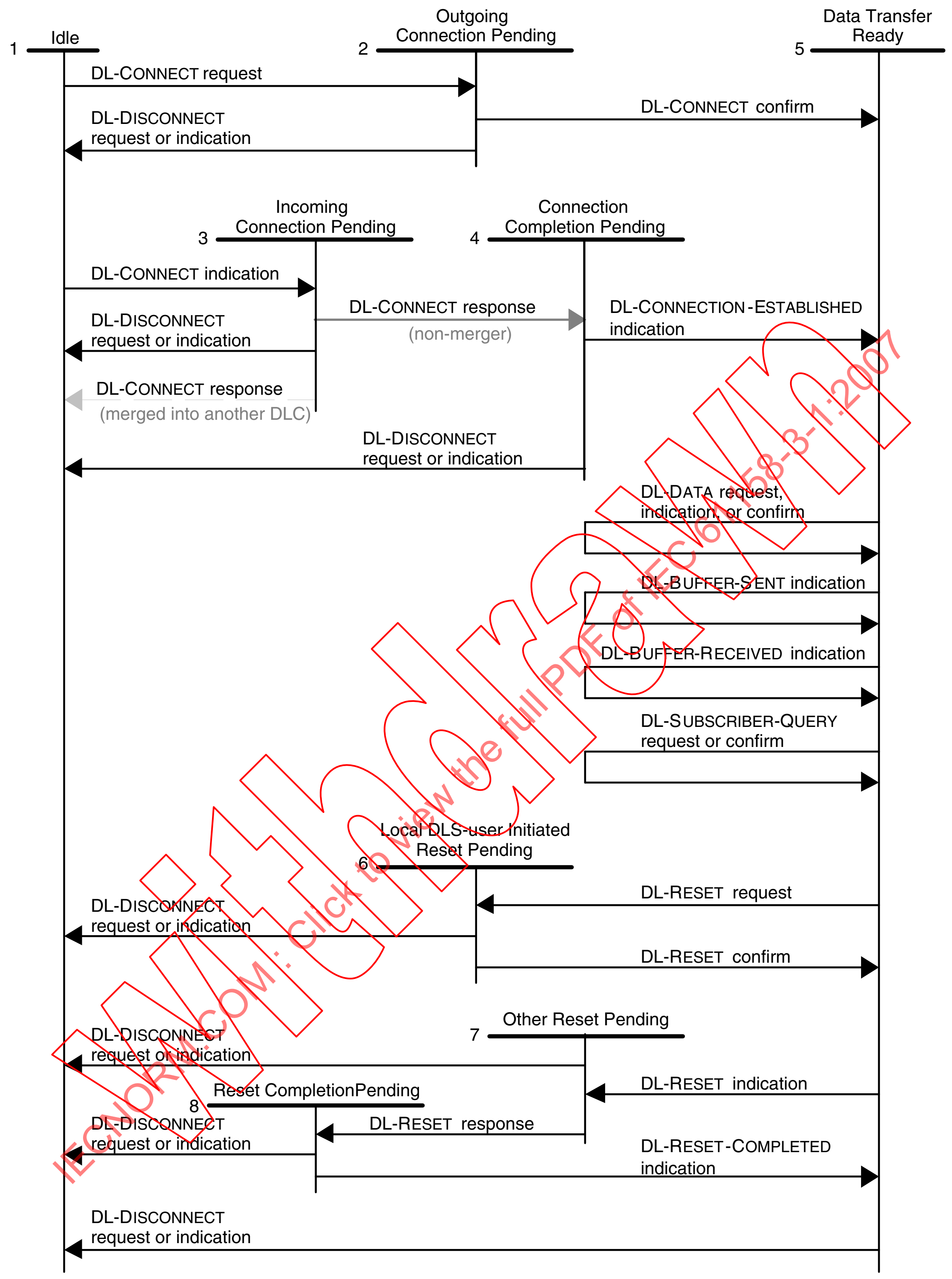
The possible overall sequences of primitives at a DLCEP are defined in the state transition diagram, Figure 15. In the diagram:

a) DL-Disconnect stands for either the request or the indication form of the primitive in all cases;

b) the labeling of the states “local DLS-user initiated reset pending” (6) and “other reset pending” (7) indicate the party that started the local interaction, and does not necessarily reflect the value of the originator parameter;

c) the “idle state” (1) reflects the absence of a DLCEP. It is the initial and final state of any sequence, and once it has been re-entered the DLCEP is released;

d) the use of a state transition diagram to describe the allowable sequences of service primitives does not impose any requirements or constraints on the internal organization of any implementation of the service.



1 Idle
2 Outgoing Connection Pending
5 Data Transfer Ready
DL-CONNECT request
DL-DISCONNECT request or indication
DL-CONNECT confirm
3 Incoming Connection Pending
4 Connection Completion Pending
DL-CONNECT indication
DL-DISCONNECT request or indication
DL-CONNECT response (non-merger)
DL-CONNECTION-ESTABLISHED indication
DL-CONNECT response (merged into another DLC)
DL-DISCONNECT request or indication
DL-DATA request, indication, or confirm
DL-BUFFER-SENT indication
DL-BUFFER-RECEIVED indication
DL-SUBSCRIBER-QUERY request or confirm
6 Local DLS-user Initiated Reset Pending
DL-DISCONNECT request or indication
DL-RESET request
DL-RESET confirm
7 Other Reset Pending
DL-DISCONNECT request or indication
DL-RESET indication
8 Reset CompletionPending
DL-DISCONNECT request or indication
DL-RESET response
DL-RESET-COMPLETED indication
DL-DISCONNECT request or indication

Figure 15 – State transition diagram for sequences of DL-connection-mode service primitives at a DLCEP

6.5 Connection establishment phase

6.5.1 Function

The DLC / DLCEP establishment service primitives can be used to establish a DLCEP, and possibly a DLC.

NOTE This function may also be provided by local DL-management actions, which are beyond the scope of this standard.

Simultaneous DL-CONNECT request primitives at the two DLSAPs may be merged into one DLC by the concurrently requesting-and-responding DLS-users as indicated in Figure 21 and Figure 22.

6.5.2 Types of primitives and parameters

Table 13 and Table 14 indicate the types of primitives and the parameters needed for DLC/DLCEP establishment.

Table 13 – DLC / DLCEP establishment primitives and parameters (portion 1)

DL-CONNECT Parameter name	Request input	 output	Indication output	Response input	Confirm output
DLCEP DLS-user-identifier	M			M	
DLCEP DL-identifier		M	M		
Called address	M		M (=)		
Calling address	M		M (=)		
Responding address				M	M (=)
Calling DLCEP-address	CU			CU	
QoS parameter set					
DLCEP class	U		M (=)	U (1)	M (=)
DLCEP data delivery features					
from requestor to responder(s)	U		M (=, 2)	U (=, 2)	M (=)
from responder(s) to requestor	U		M (=, 2)	U (=, 2)	M (=)
DLL priority	U		M (=)	U (≤)	M (=)
Maximum confirm delay					
on DL-CONNECT, DL-RESET and DL-SUBSCRIBER-QUERY	U		M (=)	U	M (=)
on DL-DATA	U		M (=)	U	M (=)
DLPDU-authentication	U		M (≥, 3)	U	M (≥, 3)
Residual activity					
as sender	U		M (=)	U (≤, 4)	M (=)
as receiver	U		M (=)	U (≤, 4)	M (=)
DL-scheduling-policy	U			U	
Maximum DLSDU sizes (5)					
from requestor	CU		M (≤)	CU (≤)	M (=)
from responder	CU		M (≤)	CU (≤)	M (=)
Buffer-and-queue bindings (5)					
as sender	CU			CU	
as receiver	CU			CU	
Sender timeliness (5)					
DL-timeliness-class	CU		M (=)	CU	M (=)
Time window size (ΔT)	CU			CU	
Synchronizing DLCEP	CU			CU	
Time-of-production	CU		C(=)	CU	C (=)
Receiver timeliness (5)					
DL-timeliness-class	CU		M (=)	CU	M (=)
Time window size (ΔT)	CU			CU	
Synchronizing DLCEP	CU			CU	
DLS-user-data	U		M (=)	U	M (=)

NOTE 1 The DLCEP classes should match, peer with peer, and publisher with subscriber.

NOTE 2 The DLCEP data delivery feature unordered may be upgraded to ordered, and disordered may be upgraded to classical.

NOTE 3 6.3.2.5 specifies that DLPDU-authentication may be upgraded from ordinary to source to maximal.

NOTE 4 6.3.2.6 specifies that a residual activity value of false may be upgraded to true.

NOTE 5 These parameters are conditional on the negotiated directions of data delivery, and in the case of timeliness, on the specified timeliness class.

NOTE 6 The method by which a confirm primitive is correlated with its corresponding preceding request primitive, or a response primitive is correlated with its corresponding preceding indication primitive, is a local matter.

Table 14 – DLC / DLCEP establishment primitives and parameters (portion 2)

DL-CONNECTION-ESTABLISHED Parameter name	Indication output
DLCEP DLS-user-identifier	M (1)
(1) The DLCEP DLS-user-identifier equals the DLS-user-identifier specified in the corresponding DL-CONNECT response primitive.	

6.5.2.1 Local-view identifiers

6.5.2.1.1 DLCEP DLS-user-identifier

This parameter specifies a means of referring to the DLCEP in output parameters of other local DLS primitives which convey the name of the DLCEP from the local DLE to the local DLS-user. It is specified by the DLS-user on DL-CONNECT request and response primitives, and is used by the DLE to refer to the DLC-end-point in DLS output parameters of other DLC primitives.

The naming-domain of the DLCEP DLS-user-identifier is the DLS-user's local-view.

6.5.2.1.2 DLCEP DL-identifier

This parameter, which is returned by the DLS-provider on DL-CONNECT request and indication primitives, provides a local means of identifying a specific DLC-end-point in input parameters of other local DLS primitives which convey the name of the DLC-end-point from the local DLS-user to the local DLE.

The naming-domain of the DLCEP DL-identifier is the DL-local-view.

6.5.2.2 Addresses

The parameters that take addresses as values refer to either DL(SAP)-addresses or DLCEP-addresses. A DLCEP-address, which is used within a DL-protocol to identify a DLCEP, is structurally similar to a DLSAP-address, and is drawn from the same overall address space.

6.5.2.2.1 Called address

This parameter conveys an address identifying the remote DLSAP(s) at which the DLC is to be established. It should be either

a) a DLSAP-address;

b) a group DL-address;

NOTE If the called address for a peer or subscriber DLC is a group DL-address, then the DLC will be established with the DLSAP whose response is first received, and the others will be disconnected when and if they respond.

c) a DLCEP-address; or

NOTE Direct knowledge of a DLCEP-address can be obtained only through extra-protocol means. The ability to specify such an address supports migration from previous national standards and facilitates the use of simple pre-configured devices within open systems.

d) the value UNKNOWN.

For a) through c), where the called address is known, it takes the form of

1) a DL-address in the request primitive; and

NOTE 1 If the Called Address is a DL(SAP)-address, and the requesting DLS-user has issued a DL-BIND request for the Called Address, then this parameter also can take the form of a DL-identifier in the request primitive.

NOTE 2 In some cases such a DL-CONNECT request may not result in corresponding DL-CONNECT indication(s) and response(s).

2) a local DLS-user-identifier for a DL(SAP)-address in the indication primitive.

For d), when the requesting DLS-user is attempting to establish

i) a peer DLCEP, and the DLS-user has learned the calling-DLCEP-address assigned to the requested DLCEP by extra-protocol means, and the remote peer DLS-user is specifying a DLCEP-address as the called address in its attempts to establish a DLC; or

ii) a publishing DLCEP, and both the publishing and subscribing DLS-users have learned the DLCEP-address assigned to the DLC by extra-protocol means then the requesting DLS-user may specify in this parameter the value UNKNOWN, rather than a DL(SAP)-address. In such a case, when the calling address is not itself a DLCEP DL-identifier, then it should also specify the calling DLCEP-address as described in 6.5.2.2.4.

NOTE A DL-address which does not designate an active DL-address of a DLE is not recognized by that DLE. Therefore it is not possible for a DLE to initiate a DLC erroneously by being addressing at an "inactive DLSAP-address or inactive DLCEP-address of the DLE"; such addresses do not exist.

6.5.2.2.2 Calling address

a) This parameter conveys an address of the local DLSAP from which the DLC has been requested. This parameter is a DLSAP-address in the indication primitive; except as in b), it takes the form of a local DLSAP-address DL-identifier in the request primitive.

   NOTE If the receiving DLS-user has issued a DL-BIND request for the Calling Address, then this parameter also can take the form of a DLSAP-address DLS-user-identifier in the indication primitive.

b) When the requesting DLS-user is the publisher of an existing multi-peer DLC, and wishes to use the request to solicit or add new subscribers to that DLC, then the requesting DLS-user should specify in this parameter the DLCEP DL-identifier returned by the earlier DL-CONNECT request or indication primitive which was used to establish that multi-peer DLC, rather than a DLSAP-address DL-identifier.

   NOTE The QoS parameters of the new DL-CONNECT request primitive should be compatible with those of the already-existing multi-peer DLC.

6.5.2.2.3 Responding address

This parameter normally conveys a DLSAP-address of the DLSAP with which the DLC has been established. In the response primitive this parameter takes the form of a local DLSAP-address DL-identifier; in the confirm primitive this parameter is either a DLSAP-address or the value **UNKNOWN**.

NOTE 1 The value UNKNOWN occurs when establishing a multi-peer PUBLISHER DLC.

NOTE 2 If the receiving DLS-user has issued a DL-BIND request for the Responding Address, then this parameter also can take the form of a DLSAP-address DLS-user-identifier in the confirm primitive.

When

a) the responding DLS-user is the publisher of an existing multi-peer DLC, and wishes to join the requesting DLS-user to that DLC as a new subscriber; or

b) the responding DLS-user detects a DL-Connect request collision and wishes to merge requested and indicated DLCs,

then the responding DLS-user should specify in this parameter the DLCEP DL-identifier returned by the earlier DL-CONNECT request or indication primitive used to establish that DLC, rather than a DLSAP-address DL-identifier. The DLS-provider interprets this response as an attempt to merge the newly-indicated DLC into the existing DLC, after which the newly-indicated DLCEP no longer exists.

NOTE 3 The QoS parameters of the new DL-CONNECT response primitive should be compatible with those of the already-existing DLC.

NOTE 4 The DLCEP-identifier for the newly-indicated DLCEP is no longer valid.

NOTE 5 No other primitives can or will be issued at that DLCEP.

6.5.2.2.4 Calling DLCEP-address

This optional parameter conveys a specific DLCEP-address (structurally similar to a DLSAP-address, and drawn from the same overall address space), for use as the local DLCEP-address for the DLC. This parameter is absent when the specified calling address in the request primitive, or the responding address in the response primitive, is a DLCEP DL-identifier.

When this parameter is both permitted and absent, and when the DLCEP requires a local DLCEP-address, then the DLE chooses a DLCEP-address from those available to it, and assigns the DLCEP-address to the DLC.

6.5.2.3 Quality-of-service parameter set

This parameter consists of a list of sub-parameters. For all parameters common to request and indication primitives, or to response and confirm primitives, the values on the primitives are related so that

a) on the request primitive, any defined value is allowed, provided that the specified value is consistent with the values of the other parameters of the same invocation of the request primitive;
b) on the response primitive, any defined value is allowed, subject to the constraints specified in the following subclauses; and
c) on the indication (or confirm) primitive, the quality of service indicated may differ from the value specified for the corresponding request (or response) primitive as indicated in Table 13.

In general the negotiation rules result in choosing the lesser of two offered levels of functionality. DLS-users which find the resulting QoS to be unacceptable should respond by issuing a DL-DISCONNECT request, specifying as a reason the unacceptability of the negotiated QoS. See 6.3.2 for the definitions of these QoS parameters.

6.5.2.3.1 DLCEP class

If the value specified in the DL-CONNECT indication is PEER, the response should be PEER. If the value specified in the DL-CONNECT indication is PUBLISHER, then the response should be SUBSCRIBER. If the value specified in the DL-CONNECT indication is SUBSCRIBER, then the response should be PUBLISHER.

6.5.2.3.2 DLCEP data delivery features

The specified values (from-requestor-to-responder(s), and from-responder(s)-to-requestor) should be returned in the DL-CONNECT response, except that the value UNORDERED may be upgraded to ORDERED, and the value DISORDERED may be upgraded to CLASSICAL.

6.5.2.3.3 DLL priority

Any defined priority lower than or equal to that specified in the corresponding indication is allowed for the response. Thus the value URGENT may be downgraded to NORMAL or TIME-AVAILABLE, and NORMAL may be downgraded to TIME-AVAILABLE.

6.5.2.3.4 DLL maximum confirm delay

This parameter is not negotiated by the DLE.

6.5.2.3.5 DLPDU-authentication

This parameter is negotiated by the DLE as specified in 4.3.3.

6.5.2.3.6 Residual activity

This parameter is negotiated by the DLE as specified in 4.3.3 and 6.3.2.6.

6.5.2.3.7 DL-scheduling-policy

This parameter is not negotiated by the DLE.

6.5.2.3.8 Maximum DLSDU sizes

Any defined value less than or equal to that specified in the corresponding indication is allowed for the response. The DLS-provider may reduce the sizes specified by the requesting DLS-user to meet DLSDU size limits set by local DLS management.

6.5.2.3.9 DLCEP buffer and queue bindings

The DLS-provider rejects any attempt to establish a DLC between a sending queue and a receiving buffer. All other possible combinations are permitted.

6.5.2.3.10 Timeliness

This parameter is not negotiated by the DLE.

6.5.2.4 DLS-user-data

This parameter allows the transmission of DLS-user-data between DLS-users, without modification by the DLS-provider. The DLS-user may transmit any integral number of octets up to the limit for DLPDUs of NORMAL priority.

NOTE The number of octets of DLS-user-data permitted is limited to that available for DLPDUs of NORMAL priority, even though DLC initiation DLPDUs are conveyed at TIME-AVAILABLE priority, to ensure that the maximum size of such DLPDUs, including all DLC parameters needed for negotiation, is not larger than the largest DLPDU used in the data transfer service.

6.5.3 Sequence of primitives

The sequence of primitives in a successful DLC/DLCEP establishment is defined by the time-sequence diagram in Figure 16 through Figure 22. These DLC/DLCEP establishment procedures may fail either

a) due to the inability of the DLS-provider to establish a DLCEP, and possibly a DLC; or

b) due to the unwillingness of the called DLS-user to accept a DL-Connect indication primitive.

NOTE For these cases, see the DLC/DLCEP release service (see 6.6.4).

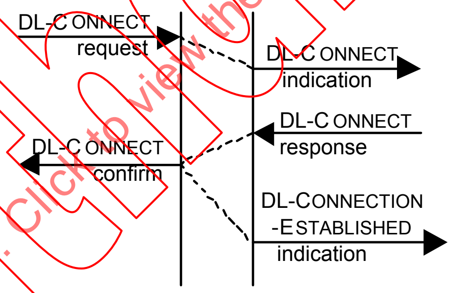


Figure 16 – Peer DLC/DLCEP establishment initiated by a single DLS-user

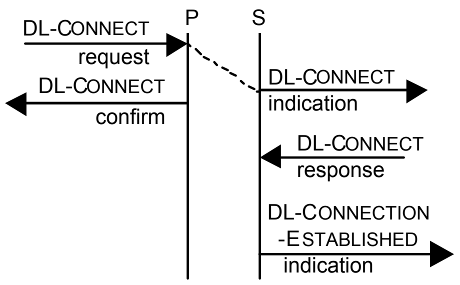


Figure 17 – Multi-peer DLC/DLCEP establishment initiated by the publishing DLS-user

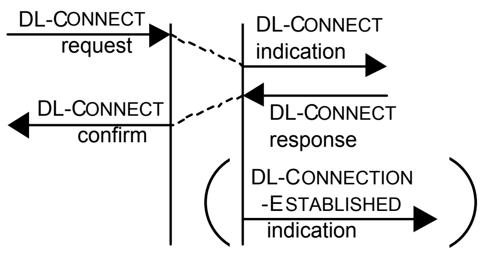


Figure 18 – Multi-peer DLC/DLCEP establishment initiated by a subscribing DLS-user

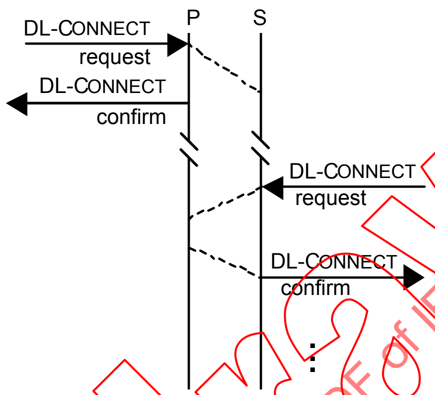


Figure 19 – Multi-peer DLC/DLCEP establishment using known DLCEP addresses initiated first by the publishing DLS-user

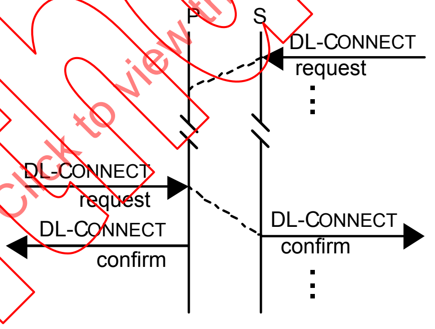


Figure 20 – Multi-peer DLC/DLCEP establishment using known DLCEP addresses initiated first by one or more subscribing DLS-users

6.5.3.1 DLC establishment collision

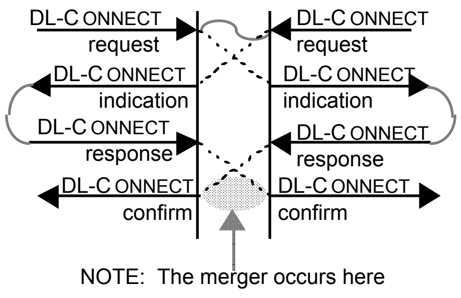


Figure 21 – Peer DLC/DLCEP establishment initiated simultaneously by both peer DLS-users, resulting in a merged DLC

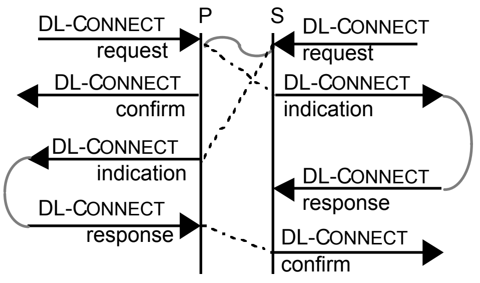


Figure 22 – Multi-peer DLC/DLCEP establishment initiated simultaneously by both publishing and subscribing DLS-users, resulting in a merged DLC

6.6 Connection release phase

6.6.1 Function

The DLC / DLCEP release service primitives are used to release a DLCEP, and possibly a DLC. The release may be initiated by any of the following:

a) either or both of the DLS-users, to release an established DLCEP;
b) the DLS-provider to release an established DLCEP (all failures to maintain a DLC at a DLCEP are indicated in this way);
c) the DLS-user, to reject a DL-Connect indication;
d) the DLS-provider, to indicate its inability to establish a requested DLC / DLCEP; or
e) the DLS-user that issued the DL-Connect request, to abandon the DLC / DLCEP establishment attempt before the DLCEP has been made available for use by receipt of a DL-CONNECT confirm primitive.

Initiation of the release service element is permitted at any time regardless of the current phase of the DLCEP. Once a release service has been initiated, the DLCEP will be disconnected and any associated buffers or queues will be unbound from the DLCEP.

A DL-DISCONNECT request cannot be rejected. The DLS does not guarantee delivery of any DLSDU associated with the DLC once the release phase is entered. Nevertheless, the release service may convey a limited amount of user data from a releasing peer (or publisher) to a peer (or subscriber), provided that neither that peer (or subscriber) nor the DLS-provider has concurrently invoked the release service.

NOTE 1 These primitives can only be used to release a DLCEP that was established by a prior DL-CONNECT primitive; they cannot be used to release a DLCEP that was established by prior local DL-management actions.

NOTE 2 This function may also be provided by local DL-management actions, which are beyond the scope of this standard.

6.6.2 Types of primitives and parameters

Table 15 indicates the types of primitives and the parameters needed for DLC/DLCEP release.

6.6.2.1 DLCEP-identifier-type

This parameter indicates whether the associated DLCEP identifier is

a) a DLS-user-identifier which was provided by the DLS-user as a parameter of the associated prior DL-Connect request or response primitive; or
b) a DL-identifier which was provided to the DLS-user as a parameter of the immediately-prior DL-Connect indication primitive.

The latter case should occur only when the DLS-provider disconnects the DLCEP after issuing a DL-CONNECT indication primitive and before receiving a corresponding DL-CONNECT response or DL-DISCONNECT request primitive from the local DLS-user.

Table 15 – DLC / DLCEP release primitives and parameters

DL-DISCONNECT Parameter name	Request input	Indication output
DLCEP-identifier-type		M
DLCEP DLS-user-identifier		C
DLCEP DL-identifier	M	C
Originator		M
Reason	U	M (=)
DLS-user-data	U	M (=)

6.6.2.2 DLCEP DLS-user-identifier

This parameter has the same value as the DLCEP DLS-user-identifier parameter provided with the DL-CONNECT request or response primitive that occurred during DLCEP initiation. It is present on the DL-DISCONNECT indication primitive except when disconnection occurs after issuing a DL-CONNECT indication primitive and before receiving a corresponding DL-CONNECT response or DL-DISCONNECT request primitive.

6.6.2.3 DLCEP DL-identifier

The DLCEP DL-identifier parameter has the same value as the DLCEP DL-identifier parameter returned by the DL-CONNECT request or indication primitive that initiated the DLCEP. It is always present on the DL-DISCONNECT request primitive, and is present on the DL-DISCONNECT indication primitive only when disconnection occurs after issuing a DL-CONNECT indication primitive and before receiving a corresponding DL-CONNECT response or DL-DISCONNECT request primitive.

6.6.2.4 Originator

This parameter identifies the source of the release. Its value indicates either the remote DLS-user, the remote DLS-provider, the local DLS-provider, or that the originator is unknown.

6.6.2.5 Reason

This parameter gives information about the cause of the release. The value conveyed in this parameter is as follows:

a) When the originator parameter indicates a DLS-provider-generated release, the value is one of

1) “disconnection — permanent condition”;
2) “disconnection — transient condition”;
3) “disconnection after timeout”;
4) “connection rejection — calling DLSAP DL-identifier is invalid”;
5) “connection rejection — DL(SAP)-address unknown”;
6) “connection rejection — DLSAP unreachable, permanent condition”;
7) “connection rejection — DLSAP unreachable, transient condition”;
8) “connection rejection — QoS not available, permanent condition”;
9) “connection rejection — QoS not available, transient condition”;
10) “connection rejection after timeout”; or
11) “reason unspecified”.

NOTE Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this standard.

b) When the originator parameter indicates a DLS-user-initiated release, the value is one of
   1) “disconnection — normal condition”;
   2) “disconnection — abnormal condition”;
   3) “disconnection — terminated by unbind of source DLSAP-address”;
   4) “connection rejection — unacceptable QoS, permanent condition”;
   5) “connection rejection — non-QoS reason, permanent condition”;
   6) “connection rejection — transient condition”; or
   7) “reason unspecified”.

c) When the originator parameter indicates an unknown originator, the value of the Reason parameter is “reason unspecified”. This allows parameters to be implied when they cannot be explicitly conveyed in a DL-protocol.

6.6.2.6 DLS-user-data

This parameter allows the transmission of DLS-user-data between DLS-users, without modification by the DLS-provider. The DLS-user may transmit any integral number of octets up to the limit for DLPDUs of NORMAL priority. Delivery of this data to the remote DLS-user(s) is not assured.

NOTE 1 The number of octets of DLS-user-data permitted is limited to that available for DLPDUs of NORMAL priority, even though DLC termination DLPDUs are conveyed at TIME-AVAILABLE priority, to provide uniformity with the DLS-user-data permitted in the DL-CONNECT service.

NOTE 2 The delivery of this data is assured only for the case specified in 6.6.3a).

6.6.3 Sequence of primitives when releasing an established DLC/DLCEP

The sequence of primitives depends on the origins of the release action. The sequence may be

a) initiated by one DLS-user, with a request from that DLS-user leading to an indication to the other;
b) initiated by both DLS-users, with a request from each of the DLS-users;
c) initiated by the DLS-provider, with an indication to each of the DLS-users; or
d) initiated independently by one DLS-user and the DLS-provider, with a request from the originating DLS-user and an indication to the other DLS-user.

In cases b) and d), any DLS-user data associated with a DLS-user-initiated request may not be delivered to the remote DLS-user(s).

The sequences of primitives in these four cases are defined by the time-sequence diagrams in Figure 23 through Figure 32.

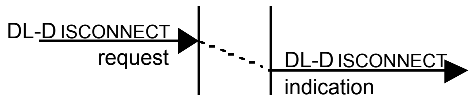


Figure 23 – Peer DLS-user invocation

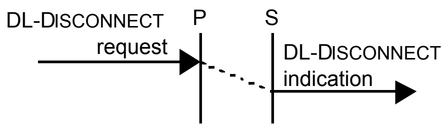


Figure 24 – Publishing DLS-user invocation

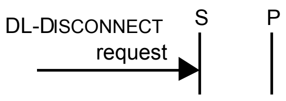


Figure 25 – Subscribing DLS-user invocation

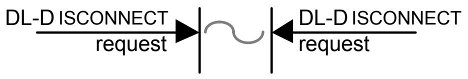


Figure 26 – Simultaneous invocation by both DLS-users

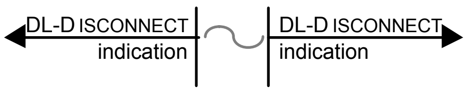


Figure 27 – Peer DLS-provider invocation

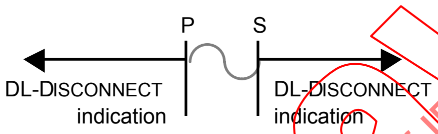


Figure 28 – Publishing DLS-provider invocation

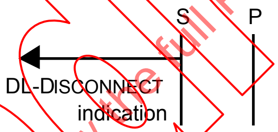


Figure 29 – Subscribing DLS-provider invocation

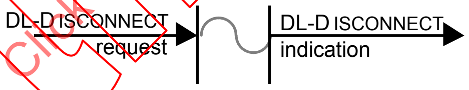


Figure 30 – Simultaneous peer DLS-user and DLS-provider invocations

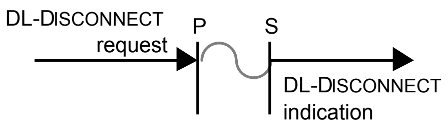


Figure 31 – Simultaneous publishing DLS-user and DLS-provider invocations

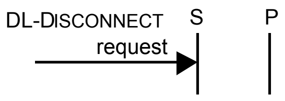


Figure 32 – Simultaneous subscribing DLS-user and DLS-provider invocations

6.6.4 Sequence of primitives in a DLS-user rejection of a DLC / DLCEP establishment attempt

A DLS-user may reject a DLC/DLCEP establishment attempt by using a DL-DISCONNECT request. The originator parameter in the DL-DISCONNECT primitives will indicate DLS-user-initiated release. The sequence of events is defined in the time-sequence diagram in Figure 33 through Figure 35.

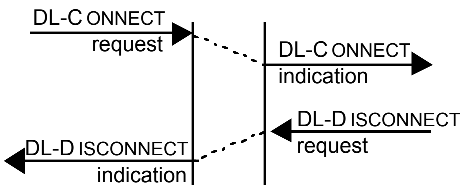


Figure 33 – Sequence of primitives in a peer DLS-user rejection of a DLC/DLCEP establishment attempt

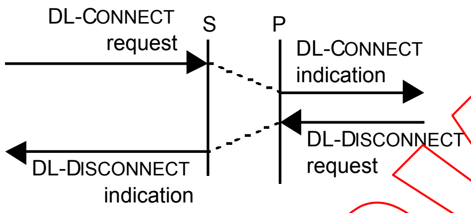


Figure 34 – Sequence of primitives in a publishing DLS-user rejection of a DLC/DLCEP establishment attempt

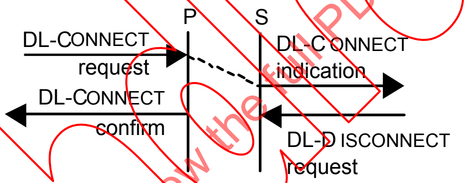


Figure 35 – Sequence of primitives in a subscribing DLS-user rejection of a DLC/DLCEP establishment attempt

If the DLS-provider is unable to establish a DLC/DLCEP, it indicates this to the requestor by a DL-DISCONNECT indication. The originator parameter in this primitive indicates a DLS-provider-originated release. The sequence of events is defined in the time-sequence diagram in Figure 36.

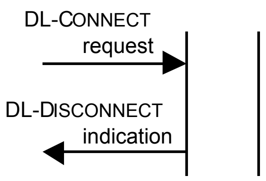


Figure 36 – Sequence of primitives in a DLS-provider rejection of a DLC/DLCEP establishment attempt

If the DLS-user, having previously sent a DL-CONNECT request and not having received a DL-CONNECT confirm or DL-DISCONNECT indication, wishes to abort the DLC/DLCEP establishment attempt, the DLS-user should issue a DL-DISCONNECT request. The resulting sequence of primitives is dependent upon the relative timing of the primitives involved and the transit delay of the DLS-provider as defined by the time-sequence diagrams in Figure 37 through Figure 41. No information can be implied by detecting which of these alternatives occur.

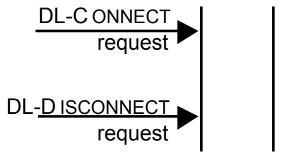


Figure 37 – Sequence of primitives in a DLS-user cancellation of a DLC/DLCEP establishment attempt: both primitives are destroyed in the queue

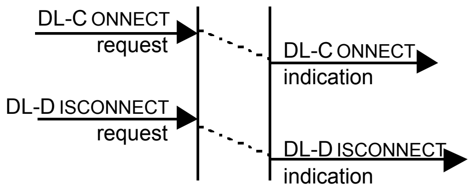


Figure 38 – Sequence of primitives in a DLS-user cancellation of a DLC/DLCEP establishment attempt: DL-DISCONNECT indication arrives before DL-CONNECT response is sent

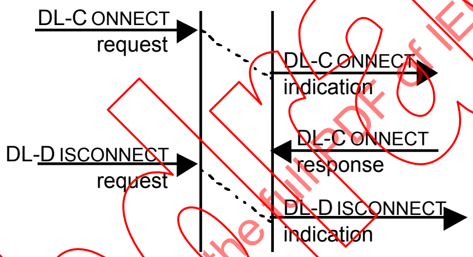


Figure 39 – Sequence of primitives in a DLS-user cancellation of a DLC/DLCEP establishment attempt: peer DL-DISCONNECT indication arrives after DL-CONNECT response is sent

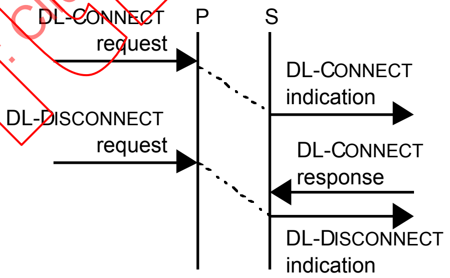


Figure 40 – Sequence of primitives in a DLS-user cancellation of a DLC/DLCEP establishment attempt: publisher's DL-DISCONNECT indication arrives after DL-CONNECT response is sent

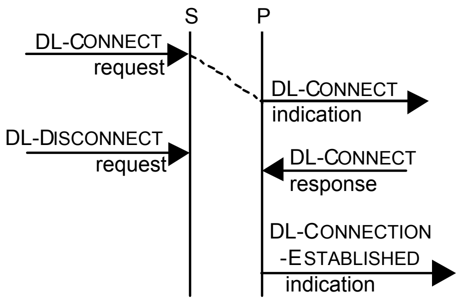


Figure 41 – Sequence of primitives in a DLS-user cancellation of a DLC/DLCEP establishment attempt: subscriber's DL-DISCONNECT request arrives after DL-CONNECT request has been communicated to the publisher

6.7 Data transfer phase

6.7.1 Queue data transfer

6.7.1.1 Function

The queue data transfer service primitives provide for conveyance of user data (DLSDUs) in either direction, or in both directions simultaneously, on a DLC. The DLS preserves the boundaries of the DLSDUs.

6.7.1.2 Types of primitives and parameters

Table 16 indicates the types of primitives and the parameters needed for queue data transfer.

Table 16 – Queue data transfer primitive and parameters

DL-DATA	Request	Indication	Confirm
Parameter name	input	output	output
Request DLS-user-identifier	M		
DLCEP DL-identifier	M		
DLCEP DLS-user-identifier		M	
Queue DLS-user-identifier		C	
DLS-user-data	M	C (=)	
Status			M
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter.			

6.7.1.2.1 DLCEP DL-identifier

This parameter, specified in the DL-DATA request primitive, has the same value as the DLCEP DL-identifier parameter returned by the DL-CONNECT request or indication primitive that initiated the DLCEP.

6.7.1.2.2 DLCEP DLS-user-identifier

This parameter has the same value as the DLCEP DLS-user-identifier parameter returned by the DL-CONNECT confirm or DL-CONNECTION-ESTABLISHED indication primitive that occurred during initiation of the DLCEP at which the DLS-user-data was received.

6.7.1.2.3 Queue DLS-user-identifier

This parameter has the same value as a queue DLS-user-identifier parameter from a prior DL-CREATE primitive (or as created by DL-management), and indicating the same queue as the one specified in the appropriate-direction buffer-or-queue-binding parameter (see 6.5.2.3.9) of the DL-CONNECT request or response primitive that initiated the DLCEP.

6.7.1.2.4 DLS-user-data

This parameter allows the transmission of data between DLS-users without alteration by the DLS-provider. The initiating DLS-user may transmit any integral number of octets greater than zero, up to the limit negotiated for the implied direction of data transfer.

6.7.1.2.4.1 Request primitive

If the initiating DLS-user has bound a buffer to the DLCEP as a source, then this primitive is invalid and the DLC is terminated by the DLS-provider, with a DL-DISCONNECT indication issued to the appropriate DLS-user(s).

If the initiating DLS-user has bound a FIFO queue of maximum depth *K* to the DLCEP as a source, then a DL-DATA request primitive attempts to append a DLSDU to the queue, but fails if the queue already contains *K* DLSDUs. If the append operation is successful, then the DLSDU will be transmitted at the first opportunity, after all preceding DLSDUs in the queue. A DL-PUT request primitive is not permitted. Instead, the queue serves to limit the number of DLS-user requests which can be outstanding (that is, not yet confirmed to the DLS-user).

If the initiating DLS-user has not bound a buffer or FIFO queue to the DLCEP as a source, then a DL-DATA request primitive attempts to append a DLSDU to an implicit queue of indeterminate capacity, but fails if the queue is full. If the append operation was successful, then the DLSDU will be transmitted at the first opportunity, after all preceding DLSDUs in the queue.

At the moment of the request, if the explicit or implicit FIFO queue is full, then the request will be rejected with an appropriate failure status on the DL-DATA request primitive.

6.7.1.2.4.2 Indication primitive

If the receiving DLS-user has bound a FIFO queue to the DLCEP as a sink, and that queue is not full, then

a) the newly-received DLSDU is appended to that queue, and the DLS-user-data parameter is omitted from the associated DL-Data indication primitive. The DLSDU can be read using a DL-Get request primitive.

If the receiving DLS-user has no binding to the DLCEP as a sink, then

b) an implicit queue of indeterminate capacity is used as a temporary receive queue, after which the DLSDU is delivered as the DLS-user-data parameter of the associated DL-DATA indication primitive.

NOTE When a buffer is bound to the DLCEP as a sink, a DL-DATA-Indication primitive cannot occur.

If it is not possible to append the received DLSDU to the implicit or explicit receive queue, then

c) if the DLCEP's data delivery features are unordered or ordered, then the DLSDU is discarded and a DL-Data indication primitive is not issued to the DLS-user; or

d) if the DLCEP's data delivery features are classical or disordered, then the DLSDU is retained by either the sending or receiving DLE (or both) until such delivery is possible, or until the DLCEP is reset or disconnected, or until the DLSDU is no longer available at the sending DLCEP (which will in turn cause a reset at the receiving DLCEP).

A DL-DATA-Indication primitive reporting the DLSDU's receipt occurs at the receiving DLS-user's DLSAP.

6.7.1.2.5 Status

This parameter allows the DLS-user to determine whether the requested DLS was provided successfully, or failed for the reason specified. The value conveyed in this parameter is as follows:

a) “success”;

b) “temporary failure — queue full”;

c) “failure — incorrect amount of requesting DLS-user data specified”;

d) “failure — incompatible DLC state”;

e) “failure — reset or disconnection”;

f) “failure — timeout”; or

g) “failure — reason unspecified”.

NOTE 1 The only compatible DLC state is Data Transfer Ready (see Figure 15).

NOTE 2 Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this standard.

6.7.1.3 Sequence of primitives

The operation of the DLS in transferring DLSDUs can be modeled as a queue of unknown size within the DLS-provider. The ability of a DLS-user to issue a DL-DATA request or of a DLS-provider to issue a DL-DATA indication depends on the behavior of the receiving DLS-user and the resulting state of the queue.

The sequence of primitives in a successful queue to queue data transfer is defined in the time-sequence diagrams in Figure 42 through Figure 44. The sequence of primitives in a failed queue to queue data transfer is defined in the time-sequence diagram in Figure 45.

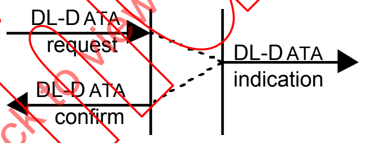


Figure 42 – Sequence of primitives for a CLASSICAL or DISORDERED peer-to-peer queue-to-queue data transfer

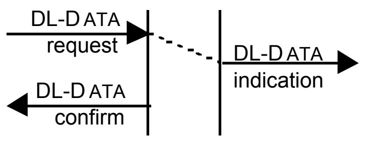


Figure 43 – Sequence of primitives for an ORDERED or UNORDERED peer-to-peer, or an UNORDERED subscriber-to-publisher queue-to-queue data transfer

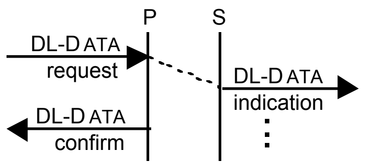


Figure 44 – Sequence of primitives for a publisher-to-subscribers queue-to-queue data transfer

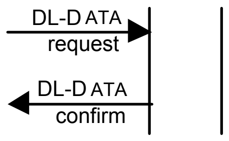


Figure 45 – Sequence of primitives for a failed queue-to-queue data transfer

If a DL-RESET or a DL-DISCONNECT primitive occurs, then the above sequences of causality may be overridden. In such a case, if a DL-DATA request primitive is outstanding, then a corresponding DL-DATA confirm primitive is issued before the reset or disconnection is signaled to the DLS-user.

6.7.2 Buffer data transfer

6.7.2.1 Function

The buffer data transfer primitives provide a means of notifying the DLS-user responsible for a buffer that the buffer was just transmitted, or that a DLSDU was just received into the buffer.

NOTE A DL-BUFFER-SENT indication will never be coincident with a DL-DATA confirm, and may or may not be related to a DL-DATA indication at one or more remote DLS-users. A DL-BUFFER-RECEIVED indication will never be coincident with a DL-DATA indication, and cannot be related to a DL-DATA request at a remote DLS-user.

6.7.2.2 Types of primitives and parameters

Table 17 and Table 18 indicate the types of primitives and the parameters needed for buffer data transfer.

Table 17 – Buffer sent primitive and parameter

DL-BUFFER-SENT Parameter name	Indication output
DLCEP DLS-user-identifier	M
Buffer DLS-user-identifier	M

Table 18 – Buffer received primitive and parameter

DL-BUFFER-RECEIVED Parameter name	Indication output
DLCEP DLS-user-identifier	M
Buffer DLS-user-identifier	M
DLSDU sequencing inference	M

6.7.2.2.1 DLCEP DLS-user-identifier

This parameter has the same value as the DLCEP DLS-user-identifier parameter returned by the DL-CONNECT confirm or DL-CONNECTION-ESTABLISHED indication primitive that occurred during initiation of the DLCEP at which the DLS-user-data was received.

6.7.2.2.2 Buffer DLS-user-identifier

This parameter has the same value as a buffer DLS-user-identifier parameter from a prior DL-CREATE primitive (or as created by DL-management), designating the same retentive buffer as the one specified in the appropriate-direction buffer-or-queue-binding parameter (see 6.5.2.3.9) of the DL-CONNECT request or response primitive that initiated the DLCEP.

6.7.2.2.3 DLSDU sequencing inference

This parameter allows the DLS-user to determine whether the DLSDU was inferred by the DLL

a) to be a duplicate of a previously-received DLSDU, or

b) to imply the loss of one or more previously-transmitted but unreceived DLSDUs, or

c) neither of the above.

When the indication is of the re-receipt of a known duplicated DLSDU, this parameter has the value “DUPLICATE”; when the indication is of the loss of one or more DLSDUs, this parameter has the value “LOSS”; in all other cases it has the value “NONE”. Since UNORDERED DLCs do not provide a basis for inferring sequencing, this parameter always has the value NONE when the DLSDU was received at a DLCEP whose data delivery features are UNORDERED.

NOTE The DLS-user may be able to use this information to distinguish between problems within the remote peer DLS-user-entity and problems within the distributed DLS-provider.

6.7.2.3 Sequence of primitives

The sequence of primitives in a successful buffer to buffer, or buffer to queue, data transfer is defined in the time-sequence diagrams in Figure 46 through Figure 49.

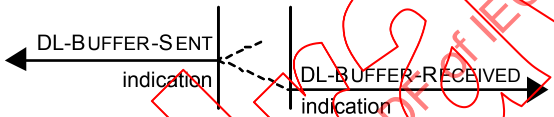


Figure 46 – Sequence of primitives for an ORDERED or UNORDERED peer to peer, or an UNORDERED subscriber to publisher, buffer to buffer data transfer

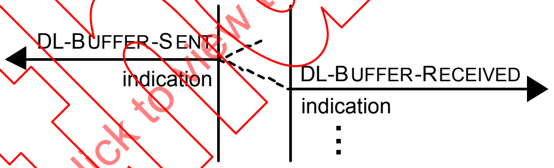


Figure 47 – Sequence of primitives for a publisher to subscribers buffer to buffer data transfer

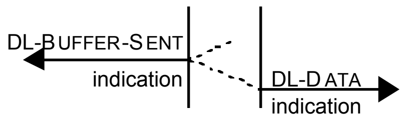


Figure 48 – Sequence of primitives for an ORDERED or UNORDERED peer to peer, or an UNORDERED subscriber to publisher, buffer to queue data transfer

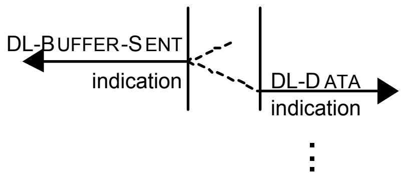


Figure 49 – Sequence of primitives for a publisher to subscribers buffer to queue data transfer

The above sequences of primitives may remain incomplete if a DL-RESET or a DL-DISCONNECT primitive occurs.

6.7.3 Reset

6.7.3.1 Function

The Reset service may be used

a) by the DLS-user of a peer or publisher DLCEP, to resynchronize the use of the DLC, and optionally to exchange a limited amount of user data with the DLC's other DLS-user(s); or

b) by the DLS-provider, to report detected loss of data unrecoverable within the DLS. All loss of data that does not involve loss of the DLC is reported in this way.

If the DLC is congested, invocation of the Reset service will unblock the flow of DLSDUs by causing the DLS-provider

1) to discard DLSDUs; and

2) to notify the appropriate DLS-user(s) that did not invoke Reset that a Reset has occurred.

The service will be completed in a finite time, whether previously-queued DLSDUs are accepted by the DLS-user or not. Any DLSDUs not delivered to the DLS-users before completion of the service will be discarded by the DLS-provider.

NOTE 1 A Reset may require a recovery procedure to be performed by the DLS-users.

NOTE 2 These primitives cannot result in the disconnection of DLCEPs that were established by prior local DL-management actions.

6.7.3.2 Types of primitives and parameters

Table 19 and Table 20 indicate the types of primitives and the parameters needed for the reset service.

Table 19 – DLC/DLCEP reset primitives and parameters (portion 1)

DL-RESET	Request	Indication	Response	Confirm
Parameter name	**input**	**output**	**input**	**output**
DLCEP DL-identifier	M			
DLCEP DLS-user-identifier		M		
Originator		M		
Reason	U	M (=)		
DLS-user-data	U	M (=)	U	M (=)
Status				M
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive, and by which a response primitive is correlated with its corresponding preceding indication primitive, is a local matter.				

Table 20 – DLC/DLCEP reset primitives and parameters (portion 2)

DL-RESET-COMPLETED	Indication
Parameter name	**output**
DLCEP DLS-user-identifier	M

6.7.3.2.1 DLCEP DL-identifier

The DLCEP DL-identifier parameter has the same value as the DLCEP DL-identifier parameter returned by the DL-CONNECT request or indication primitive that initiated the DLCEP.

6.7.3.2.2 DLCEP DLS-user-identifier

This parameter has the same value as the DLCEP DLS-user-identifier parameter returned by the DL-CONNECT confirm or DL-CONNECTION-ESTABLISHED indication primitive that occurred during initiation of the DLCEP at which the DL-RESET or DL-RESET-COMPLETED indication primitive was received.

6.7.3.2.3 Originator

This parameter indicates the source of the Reset; the set of values is identical to that specified in 6.6.2.2. The parameter's value indicates either the remote DLS-user, the remote DLS-provider, the local DLS-provider, or that the originator is unknown.

6.7.3.2.4 Reason

This parameter gives information about the cause of the reset. The value conveyed in this parameter is as follows:

a) When the originator parameter indicates a DLS-provider-generated reset, the value is one of
   1) "resynchronization after activation of a DL-management-established DLCEP";
   2) "resynchronization after timeout";
   3) "resynchronization after maximum number of retransmission requests or attempts";
   4) "resynchronization after detected sequence number error";
   5) "resynchronization after other detected DLCEP state inconsistencies";
   6) "reason unspecified".

   NOTE Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this standard.

b) When the originator parameter indicates a DLS-user-initiated reset, the value is one of
   1) "resynchronization after DLS-user timeout";
   2) "resynchronization after DLS-user-detected DLS-user-state inconsistencies";
   3) "reason unspecified"; or

   NOTE Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this standard.

c) When the originator parameter indicates an unknown originator, the value of the Reason parameter is "reason unspecified". This allows parameters to be implied when they cannot be explicitly conveyed in the DL-protocol.

6.7.3.2.5 DLS-user-data

This parameter allows the transmission of DLS-user-data between DLS-users, without modification by the DLS-provider. The DLS-user may transmit any integral number of octets up to the limit for DLPDUs of the DLC's priority.

NOTE The delivery of this data is assured only for the case where the reset service is invoked by only one peer or publishing DLS-user, and not simultaneously by another DLS-user or the DLS-provider, and where the reset service completes before the initiation of a subsequent reset or disconnect.

6.7.3.2.6 Status

This parameter allows the DLS-user to determine whether the requested DLS was provided successfully, or failed for the reason specified. The value conveyed in this parameter is as follows:

a) "success";
b) "failure — incompatible DLC state";
c) "failure — disconnection"; or
d) "failure — reason unspecified".

NOTE 1 The only compatible DLC state is Data Transfer Ready (see Figure 15).

NOTE 2 Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this clause.

6.7.3.3 Sequence of primitives

The interaction between each DLS-user and the DLS-provider should be one of the following exchanges of primitives:

a) a DL-RESET request from the DLS-user, followed by a DL-Reset confirm from the DLS-provider; or

b) a DL-RESET indication from the DLS-provider, followed by a DL-Reset response from the DLS-user, followed by a DL-RESET-COMPLETED indication from the DLS-provider.

The DL-RESET request acts as a synchronization mark in the stream of DLSDUs that are sent by the issuing DLS-user. The DL-RESET indication acts as a synchronization mark in the stream of DLSDUs that are received by the peer DLS-user. The DL-RESET response acts as a synchronization mark in the stream of DLSDUs that are sent by the responding DLS-user. The DL-RESET confirm acts as a synchronization mark in the stream of DLSDUs that are received by the DLS-user that originally issued the reset.

When the requesting DLS-user is the publisher of a multi-peer DLC, then this behavior is modified to account for the fact that synchronization marks corresponding to DL-RESET response primitives are not actually sent to the publishing DLCEP, but rather a DL-RESET confirm primitive is locally generated at the publishing DLCEP after sending the DL-RESET request synchronization mark to the subscribing DLCEPs.

In general, the resynchronization properties of the Reset service are that

1) No DLSDU sent by the DLS-user before the DL-RESET request or response in that sent stream is delivered to the peer DLS-user after the corresponding DL-Reset indication or confirm in that received stream.

The DLS-provider discards all DLSDUs submitted before the issuing of the DL-Reset request that have not been delivered to the peer DLS-user when the DLS-provider issues the DL-Reset indication.

Also, the DLS-provider discards all DLSDUs submitted before the issuing of the DL-Reset response that have not been delivered to the requestor of the DL-Reset when the DLS-provider issues the DL-Reset confirm.

2) No DLSDU sent by the DLS-user after the synchronization mark in that sent stream is delivered to another DLS-user **before** the synchronization mark in that received stream.

However, this partitioning of DLSDUs into pre-reset and post-reset epochs is only approximate when the DLCEP data delivery features are UNORDERED, because the lack of complete ordering includes a lack of complete ordering of the entries in the abstract queues.

The complete sequence of primitives depends upon the origin of the Reset action and the occurrence or otherwise of conflicting origins. Thus the Reset service may be

i) invoked by one DLS-user of a peer DLC, and possibly simultaneously by the DLS-provider, leading to interaction a) with that DLS-user and interaction b) with the peer DLS-user;

ii) invoked by both DLS-users of a peer DLC, leading to interaction a) with both DLS-users;

iii) invoked by the DLS-provider of a peer DLC, leading to interaction b) with both DLS-users;

iv) invoked by the subscribing user of a multi-peer DLC, or by that part of the DLS-provider associated with the subscribing DLS-user, leading to interaction a) with that DLS-user, and no interaction with the DLC's other DLS-user(s);

v) invoked by that part of the DLS-provider associated with the publishing DLS-user, leading to interaction b) with the DLC's DLS-users; or

vi) invoked by the publishing user of a multi-peer DLC, and possibly simultaneously by one or more subscribing users, leading to interaction a) with the invoking DLS-user(s) and interaction b) with all the DLC's other DLS-users.

The sequences of primitives for these six alternatives are defined in the time-sequence diagrams in Figure 50 through Figure 60.

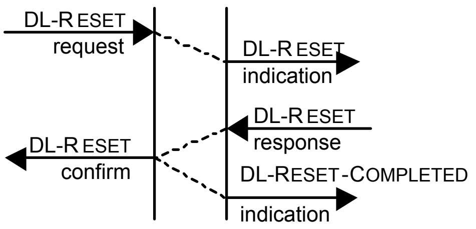


Figure 50 – Sequence of primitives in a peer DLS-user initiated Reset

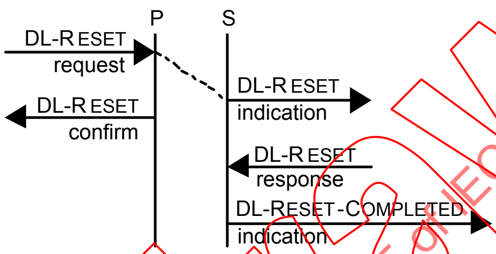


Figure 51 – Sequence of primitives in a publishing DLS-user initiated Reset

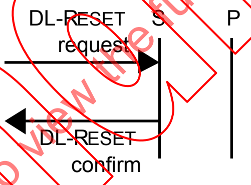


Figure 52 – Sequence of primitives in a subscribing DLS-user initiated Reset

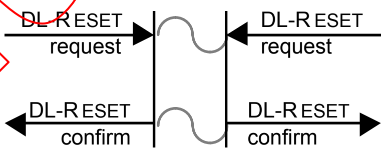


Figure 53 – Sequence of primitives in a simultaneous peer DLS-users initiated Reset

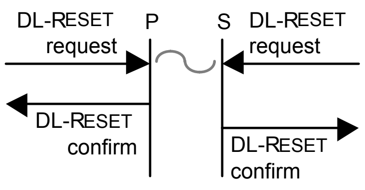


Figure 54 – Sequence of primitives in a simultaneous multi-peer DLS-users initiated Reset

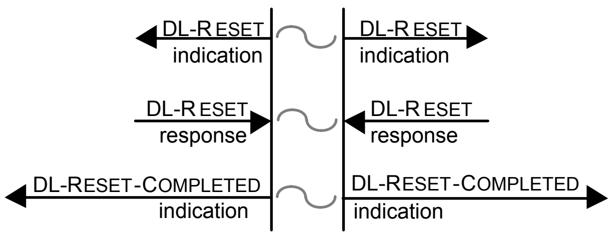


Figure 55 – Sequence of primitives in a peer DLS-provider initiated Reset

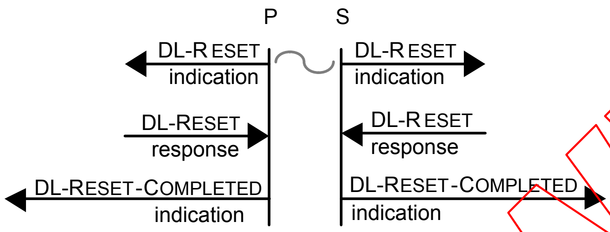


Figure 56 – Sequence of primitives in a publishing DLS-provider initiated Reset

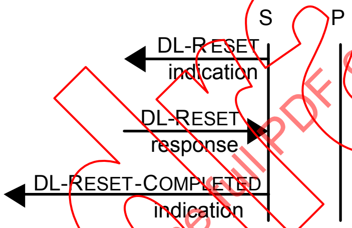


Figure 57 – Sequence of primitives in a subscribing DLS-provider initiated Reset

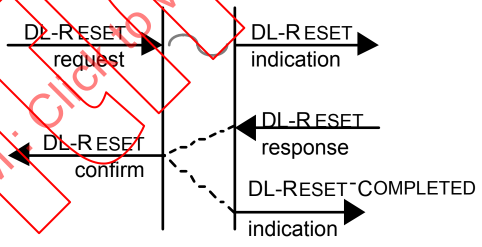


Figure 58 – Sequence of primitives in a simultaneous peer DLS-user and DLS-provider initiated Reset

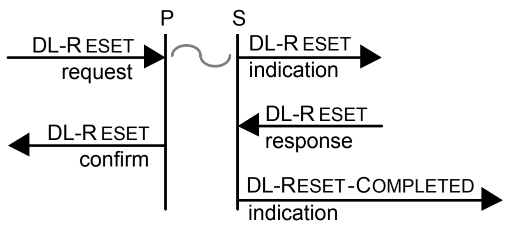


Figure 59 – Sequence of primitives in a simultaneous publishing DLS-user and DLS-provider initiated Reset

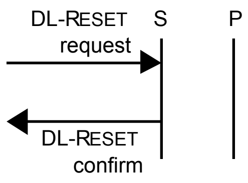


Figure 60 – Sequence of primitives in a simultaneous subscribing DLS-user and DLS-provider initiated Reset

The above sequences of primitives may remain incomplete if a DL-DISCONNECT primitive occurs. In such a case, if a DL-RESET request primitive is outstanding, then a corresponding DL-RESET confirm primitive should be issued before the disconnection is signaled to the DLS-user.

6.7.4 Subscriber query

6.7.4.1 Function

The subscriber query service primitives provide for the publishing DLS-user to query the existence of any subscribers on a multi-peer DLC. The information returned specifies whether the set of subscribing DLS-users of the multi-peer DLC appears to be empty.

6.7.4.2 Types of primitives and parameters

Table 21 indicates the types of primitives and the parameters needed for subscriber query.

Table 21 – Subscriber query primitives and parameters

DL-SUBSCRIBER-QUERY	Request	Confirm
Parameter name	input	output
DLCEP DL-identifier	M	
DLCEP DLS-user-identifier		M
Status		M
NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter.		

6.7.4.2.1 DLCEP DL-identifier

The DLCEP DL-identifier parameter has the same value as the DLCEP DL-identifier parameter returned by the DL-CONNECT request or indication primitive that initiated the DLCEP.

6.7.4.2.2 DLCEP DLS-user-identifier

This parameter has the same value as the DLCEP DLS-user-identifier parameter returned by the DL-CONNECT confirm or DL-CONNECTION-ESTABLISHED indication primitive that occurred during initiation of the DLCEP at which the DL-SUBSCRIBER-QUERY request was received.

6.7.4.2.3 Status

This parameter allows the DLS-user to determine whether the requested DLS was initiated successfully, or failed for the reason specified, and whether any receiving DLS-users appear to exist or not. The value conveyed in this parameter is as follows:

a) “success” — a subscriber exists;

b) “indeterminate — timeout”;

c) “failure — incompatible DLC state”;

d) “failure — disconnection”; or

e) “failure — reason unspecified”.

NOTE 1 The status “failure — timeout” may be interpreted with an unknown degree of confidence as “success — no subscribers”

NOTE 2 Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard which provides services as specified in this clause.

6.7.4.3 Sequence of primitives

The sequence of primitives in a successful subscriber is defined in the time-sequence diagram in Figure 61. The scheduling of DL-SUBSCRIBER-QUERY is always IMPLICIT.

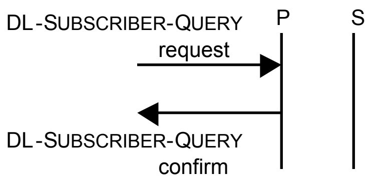


Figure 61 – Sequence of primitives for Subscriber Query

The above sequence of primitives may remain incomplete if a DL-DISCONNECT primitive occurs.

7 Connectionless-mode data-link layer service

7.1 Facilities of the connectionless-mode data-link layer service

NOTE These facilities may not be adequate of themselves to provide the ordered-delivery enhancement to the basic connectionless OSI data-link services of ISO/IEC 8886 which ISO/IEC 15802-1 promises to users of local area network medium access control protocols. In such a case it may be necessary to use an ORDERED DLC and a convergence sub-protocol to emulate the enhanced connectionless services of ISO/IEC 15802-1, with a PEER DLC used to support point-to-point (unicast) services, and a MULTI-PEER DLC used to support multicast services.

The DLS provides the following facilities to the DLS-user:

a) A means of transferring DLSDUs of limited length from one source DLSAP to a destination DLSAP or group of DLSAPs, without establishing or later releasing a DLC. The transfer of DLSDUs is transparent, in that the boundaries of DLSDUs and the contents of DLSDUs are preserved unchanged by the DLS, and there are no constraints on the DLSDU content (other than limited length) imposed by the DLS. QoS for this transmission can be selected by the sending DLS-user.

   NOTE The length of a DLSDU is limited because of internal mechanisms employed by the DL-protocol (see ISO/IEC 7498-1).

b) A means by which the status of delivery to that destination DLSAP can be returned to the source DLSAP.

c) A means by which previously submitted DLSDUs of limited length can be exchanged between two DLSAPs, and status about the exchange can be provided to the DLS-users, without establishing or later releasing a DLC. The transfer of the DLSDUs is transparent, in that the boundaries of the DLSDUs and the contents of the DLSDUs are preserved unchanged by the DLS, and there are no constraints on the DLSDU content (other than limited length) imposed by the DLS.

   NOTE 1 The length of a DLSDU is limited because of internal mechanisms employed by the DL-protocol (see ISO/IEC 7498-1).

   NOTE 2 Because this is a unitdata service it is inherently asymmetric, the status available to the two DLS-users reflects that asymmetry. Each DLS-user receives status that the exchange occurred, but only the one with the DL(SAP)-address role of INITIATOR can receive status that its DLSDU was successfully delivered to the other DLS-user.

   NOTE 3 The ability to constrain a DLSAP-address bound in a RESPONDER role to unitdata-exchange only with a specified DLSAP-address bound in an INITIATOR role can be viewed as a form of asymmetric light-weight peer-to-peer unordered DLC.

d) A means by which a DLS-user can be notified that such an exchange was attempted, but that no DLSDUs were available for the exchange.

e) A means by which a DLS-user can query if there are any DLS-users that can receive DLSDUs sent to a specified (usually group) DL(SAP)-address.

7.2 Model of the connectionless-mode data-link layer service

This clause uses the abstract model for a layer service defined in ISO/IEC 10731, Clause 5. The model defines interactions between the DLS-user and the DLS-provider that take place at a DLSAP. Information is passed between the DLS-user and the DLS-provider by DLS primitives that convey parameters.

7.2.1 Model of DL-connectionless-mode unitdata transmission

A defining characteristic of data-link layer connectionless-mode unitdata transmission is the independent nature of each invocation of the DLS.

As a descriptive aid, the data-link layer connectionless-mode unitdata transmission service, as provided between any two DLSAPs, can be modeled in the abstract as an association between the two DLSAPs. This association is permanent, but its activation requires autonomous actions, in the form of appropriate DL-BIND requests, at the two DLSAPs.

Only one type of object, the unitdata object, can be handed over to the DLS-provider, via a DLSAP, for transmission. In Figure 62, DLS-user A represents the DLS-user that passes objects to the DLS-provider. DLS-user B represents the DLS-user that accepts objects from the DLS-provider.

The operations that are performed by the DLS-provider for a particular DLL association depend on the QoS specified by the requesting DLS-user. Awareness of the characteristics of the DLS provider is part of the DLS-user's à priori knowledge of the OSI environment.

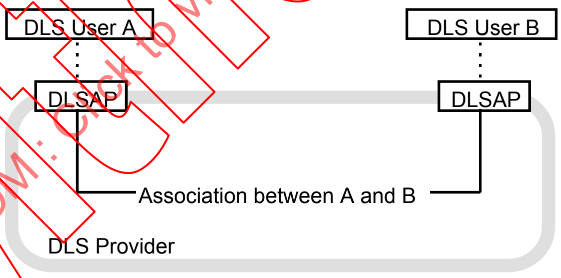


Figure 62 – Model for a data-link layer connectionless-mode unitdata transmission or unitdata exchange

7.2.2 Model of DL-connectionless-mode unitdata exchange

A defining characteristic of data-link layer connectionless-mode unitdata exchange is the independent nature of each invocation of the DLS.

As a descriptive aid, the data-link layer connectionless-mode unitdata exchange service, as provided between any two DLSAPs, can be modeled in the abstract as an association between the two DLSAPs. This association is permanent, but its activation requires autonomous actions, in the form of appropriate DL-BIND requests, at the two DLSAPs.

Only one type of object, the unitdata object, can be handed over to the DLS-provider, via a DLSAP, for transmission. In Figure 62, DLS-user A represents the DLS-user which has configured its relevant DLSAP-address in the role of INITIATOR. DLS-user B represents the DLS-user which has configured its relevant DLSAP-address in the role of CONSTRAINED RESPONDER or UNCONSTRAINED RESPONDER.

For each DLSAP-address, the configuring DL-BIND service specifies up to three buffers which can provide DLSDUs for transmission, and up to three buffers or queues which can hold received DLSDUs.

Each invocation of the unitdata-exchange DLS specifies a service priority, and causes each DLE to select, from those sending buffers which are bound at the specified or higher priority, the highest-priority DLSDU which is available for transmission.

The unitdata-exchange DLS consists of two phases. During the first phase, DLS-user A's DLE selects the DLSDU for transmission, if any, and sends it to DLS-user B's DLE, where the DLSDU is received and placed in the receive buffer or queue, if any, associated with the DLSDU's priority.

During the second phase, if a DLSDU was sent but was unable to be delivered in the first phase, then an error report is returned to DLS-user A's DLE. Otherwise, if either no DLSDU was sent, or the sent DLSDU was successfully placed in the appropriate receive buffer or queue, then DLS-user B's DLE selects the DLSDU for transmission, if any, and sends it to DLS-user A's DLE, where the DLSDU is received and placed in the receive buffer or queue, if any, associated with the DLSDU's priority.

For each of the two DLSAPs involved in the transaction, if a DLSDU was either sent or received at the DLSAP, or if the DLSAP-address binding specified that unitdata-exchange indications are always required, then the DLE issues a DL-UNITDATA-EXCHANGE indication primitive at that DLSAP.

7.3 Quality of connectionless-mode service

The term quality-of-service (QoS) refers to certain characteristics of a connectionless-mode data transmission as observed between the DLSAPs. QoS describes aspects of a connectionless-mode data transmission that are attributable solely to the DLS-provider. QoS can only be properly determined when DLS-user behavior does not constrain or impede the performance of the DLS.

Whether the QoS during each instance of connectionless-mode data transmission appears the same to each DLS-user associated with the service depends

a) on the nature of their association; and

b) on the type of information, concerning the nature of the service, made available to the DLS-user(s) by the DLS-provider prior to the invocation of the service.

7.3.1 Determination of QoS for connectionless-mode service

A basic characteristic of a connectionless-mode service is that no long-term dynamic association is set up between the parties involved. Thus, associated with each DL-connectionless-mode data transmission, certain measures of QoS are requested by the sending or initiating DLS-user when the primitive action is initiated.

NOTE The ability to constrain a DLSAP-address bound in a CONSTRAINED-RESPONDER role to unitdata-exchange only with a specified DLSAP-address bound in an INITIATOR role can be viewed as a form of asymmetric association by configuration, but without any DL-related communication between the correspondent DLS-users.

7.3.2 Definition of QoS parameters

The QoS attributes for connectionless service are:

7.3.2.1 DLL priority

Connectionless data transfer and data exchange primitives specify the priority of the transferred DLSDU(s). This parameter is defined and its default value specified in 4.3.1 and 5.4.3.2.6.1.

7.3.2.2 DLL maximum confirm delay

This parameter is defined and its default value specified in 4.3.2 and 5.4.3.2.6.2.

NOTE For DLEs that do not support a time resolution of 1 ms, the requested time interval may be rounded up to the next-greatest multiple of the resolution that the DLE does support.

7.3.2.3 Remote-DLE-confirmed

This Boolean parameter specifies whether the DLS-user requests confirmation of receipt of the associated DLSDU by the (implicitly identified) remote DLE. Its permissible values are **TRUE** and **FALSE**, and its default value is FALSE.

NOTE When providing a unitdata service, selection of the value TRUE inevitably uses more link capacity than does selection of the value FALSE.

7.4 Sequence of primitives

7.4.1 Constraints on sequence of primitives

This subclause defines the constraints on the sequence in which the primitives defined in 7.5 may occur. The constraints determine the order in which primitives occur, but do not fully specify when they may occur. Other constraints, such as flow control of data, will affect the ability of a DLS-user or a DLS-provider to issue a primitive at any particular time.

The connectionless-mode primitives and their parameters are summarized in Table 22.

Table 22 – Summary of DL-connectionless-mode primitives and parameters

Service	Service subtype	Primitive		Parameter
Data Transfer	Unitdata	DL-UNITDATA request	(*in*	Called DL(SAP)-address, Calling DLSAP-address DL-identifier, QoS parameter set, limited DLS-user-data)
		DL-UNITDATA indication	(*out*	Called DL(SAP)-address DLS-user-identifier, Calling DLSAP-address, QoS parameter set, Queue DLS-user-identifier, limited DLS-user-data)
		DL-UNITDATA confirm	(*out*	Status)
	Unitdata Exchange	DL-UNITDATA-EXCHANGE indication (3)	(*out*	Local DLSAP-address DLS-user-identifier, Remote DLSAP-address, QoS parameter set, Buffer-or-queue DLS-user-identifier, Status)
Listener Query	Listener Query	DL-LISTENER-QUERY request	(*in*	DL(SAP)-address, QoS parameter)
		DL-LISTENER-QUERY confirm	(*out*	Status)

NOTE 1 DL-identifiers in parameters are local and assigned by the DLS-provider and used by the DLS-user to designate a specific DL(SAP)-address, schedule, buffer-or-queue to the DLS-provider at the DLS interface. DLS-user-identifiers in parameters are local and assigned by the DLS-user and used by the DLS-provider to designate a specific DL(SAP)-address, schedule, buffer-or-queue to the DLS-user at the DLS interface.

NOTE 2 The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter.

NOTE 3 DL-Unitdata-Exchange indication occurs in isolation, without either request or confirm primitives, at both DLSAPs involved in an instance of the explicitly scheduled Unitdata-Exchange service. The DL-Compel-Service and DL-Schedule-Sequence services (see 8.5.2 and 8.5.3) provide the means for the explicit scheduling of the Unitdata-Exchange service.

7.4.2 Relation of primitives at the end-points of connectionless service

With few exceptions, a primitive issued at one DLSAP will have consequences at one or more other DLSAPs. The relations of primitives of each type at one DLSAP to primitives at the other DLSAPs are defined in the appropriate subclause in 7.5; all these relations are summarized in the diagrams of Figure 63.

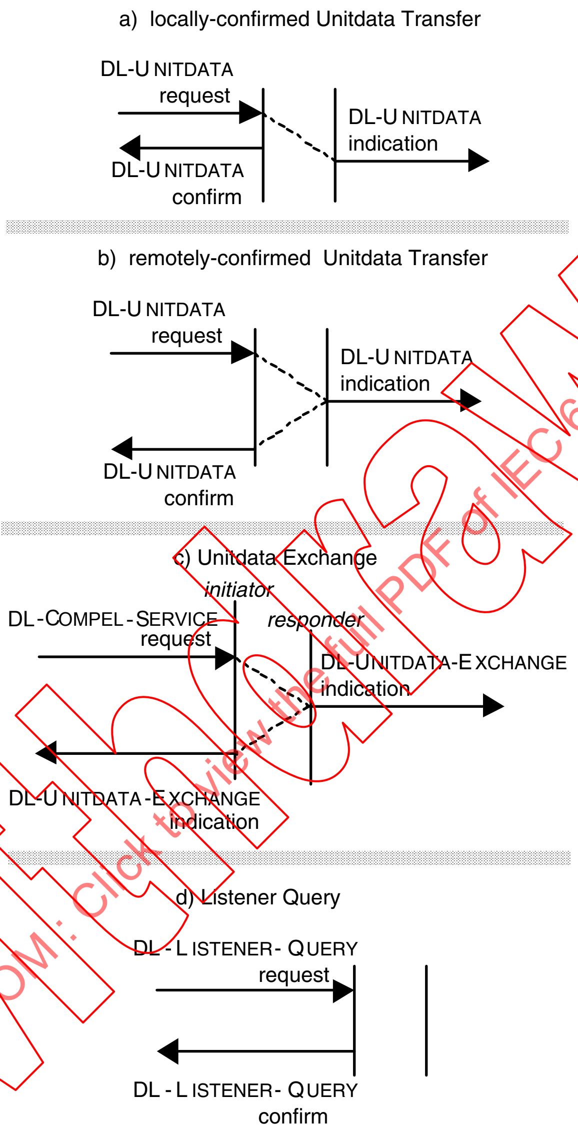

a) locally-confirmed Unitdata Transfer
DL-UNITDATA request
DL-UNITDATA indication
DL-UNITDATA confirm
b) remotely-confirmed Unitdata Transfer
DL-UNITDATA request
DL-UNITDATA indication
DL-UNITDATA confirm
c) Unitdata Exchange
initiator
responder
DL-COMPEL-SERVICE request
DL-UNITDATA-EXCHANGE indication
DL-UNITDATA-EXCHANGE indication
d) Listener Query
DL-LISTENER-QUERY request
DL-LISTENER-QUERY confirm

Figure 63 – Summary of DL-connectionless-mode service primitive time-sequence diagrams

7.4.3 Sequence of primitives at one DLSAP

The possible overall sequences of primitives at a DLSAP are defined in the state transition diagram, Figure 64. In the diagram, the use of a state transition diagram to describe the allowable sequences of service primitives does not impose any requirements or constraints on the internal organization of any implementation of the service.

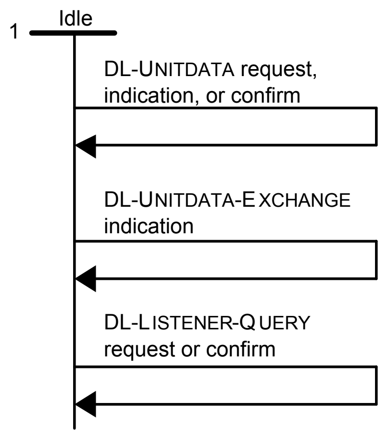

1
Idle
DL-UNITDATA request, indication, or confirm
DL-UNITDATA-EXCHANGE indication
DL-LISTENER-QUERY request or confirm

Figure 64 – State transition diagram for sequences of connectionless-mode primitives at one DLSAP

7.5 Connectionless-mode functions

DL-connectionless-mode unitdata transmission and unitdata exchange service primitives can be used to transmit independent DLSDUs from one DLSAP to another DLSAP. Each DLSDU is transmitted in a single DLPDU. The DLSDU is independent in the sense that it bears no relationship to any other DLSDU transmitted through an invocation of the DLS. The DLSDU is self-contained in that all the information required to deliver the DLSDU is presented to the DLS-provider, together with the user data to be transmitted, in a single service access.

A DLSDU transmitted using DL-connectionless-mode unitdata transmission or unitdata exchange services is not considered by the DLS-provider to be related in any way to any other DLSDU. Although the DLS maintains the integrity of individual DLSDUs, it does not necessarily deliver them to the receiving DLS-user in the order in which they are presented by the sending DLS-user.

NOTE Although the DL-UNITDATA-EXCHANGE service provides status about the delivery or transmission of one DLSDU when it reports the reception of a second DLSDU, there is no constraining relationship between the DLSDUs themselves.

No means are provided by which the receiving DLS-user may control the rate at which the sending DLS-user may send DLSDUs (peer-to-peer flow control). The DLS-provider will not maintain any state information about the flow of information between DLSAPs. Flow control exerted by the DLS-provider upon a sending DLS-user can only be described for a specific interface.

7.5.1 Data transfer

7.5.1.1 Function

This service provides the facilities of 20a) and 20b). It can be used to transmit an independent, self-contained DLSDU from one DLSAP to another DLSAP in a single service access, and to return the status of that delivery to the originating DLSAP.

This service can also be used to transmit an independent, self-contained DLSDU from one DLSAP to a group of DLSAPs, all in a single service access. In this case delivery status is not available to the originating DLSAP.

A DLSDU transmitted using DL-connectionless-mode data transfer is not considered by the DLS-provider to be related in any way to any other DLSDU. Although the DLS maintains the integrity of individual DLSDUs, it does not necessarily deliver them to the receiving DLS-user in the order in which they are presented by the sending DLS-user.

NOTE The possibility, probability and reasons for such misordering are protocol-specific.

7.5.1.2 Types of primitives and parameters

Table 23 indicates the types of primitives and the parameters needed for the DL-connectionless-mode unitdata transmission service. This service may be initiated from any DLSAP-address whose binding DL(SAP)-role is BASIC. This service may be addressed to any DL(SAP)-address whose binding DL(SAP)-role is BASIC or GROUP.

Table 23 – DL-connectionless-mode unitdata transfer primitives and parameters

DL-UNITDATA	Request	Indication	Confirm
Parameter name	**input**	**output**	**output**
Called address	M	M (=)	
Calling address	M	M (=)	
QoS parameter set			
DLL priority	U	M (=)	
DLL maximum confirm delay	U		
Remote-DLE-confirmed	U		
Queue DLS-user-identifier		C	
DLS-user-data	M	C (=)	
Status			M

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter.

7.5.1.2.1 Addresses

The parameters that take addresses as values (see 7.5.1.2.1.1 and 7.5.1.2.1.2) both refer to DL(SAP)-addresses.

7.5.1.2.1.1 Called address

This parameter conveys an address identifying the remote DLSAP(s) with which the DLS is to be provided. It is a DL(SAP)-address in the request primitive, but takes the form of a local DL(SAP)-address DLS-user-identifier in the indication primitive(s). It may be a DLSAP-address or a group DL-address.

NOTE If the requesting DLS-user has issued a DL-BIND request for the Called Address, then this parameter also can take the form of a DL(SAP)-address DL-identifier in the request primitive.

7.5.1.2.1.2 Calling address

This parameter conveys an address of the local DLSAP from which the DLS has been requested. It is a DLSAP-address in the indication primitive, but takes the form of a local DLSAP-address DL-identifier in the request primitive.

NOTE If the receiving DLS-user has issued a DL-BIND request for the Calling Address, then this parameter also can take the form of a DLSAP-address DLS-user-identifier in the indication primitive.

7.5.1.2.2 Quality-of-service

This parameter consists of a list of sub-parameters. For each parameter, the values on the primitives are related so that

a) on the request primitive, any defined value is allowed, consistent with the other parameters; and

b) on the indication primitive, the quality of service indicated is equal to the value specified for the corresponding request primitive.

7.5.1.2.2.1 DLL priority

This is the priority of the actual DLSDU which is sent to the remote peer DLS-user.

7.5.1.2.2.2 DLL maximum confirm delay

This QoS attribute is not reported on the indication primitive.

7.5.1.2.2.3 Remote-DLE-confirmed

When the called address is a group DL-address the specified value for this QoS attribute should be FALSE. This QoS attribute is not reported on the indication primitive.

7.5.1.2.3 Queue DLS-user-identifier

This parameter has the same value as a Queue DLS-user-identifier parameter from a prior DL-CREATE primitive (or as created by DL-management). It is present when an explicit queue was specified for reception at the indicated priority by the DL-BIND request primitive which established the DL(SAP)-address indicated in the same primitive.

7.5.1.2.4 DLS-user data

This parameter allows the transmission of data between DLS-users without alteration by the DLS-provider. The initiating DLS-user may transmit any integral number of octets greater than zero, up to the limit determined by the DLL priority QoS parameter specified in the service request.

7.5.1.2.4.1 Request primitive

If the initiating DLS-user has bound a FIFO queue of maximum depth *K* to the DLSAP-address at the specified priority as a source, then a DL-UNITDATA request primitive attempts to append a DLSDU to the queue, but fails if the queue already contains *K* DLSDUs. If the append operation is successful, then the DLSDU will be transmitted at the first opportunity, after all preceding DLSDUs in the queue. A DL-PUT request primitive is not permitted. Instead, the queue serves to limit the number of DLS-user requests which can be outstanding (that is, not yet confirmed to the DLS-user).

If the initiating DLS-user has not bound a FIFO queue to the DLSAP-address at the specified priority as a source, then a DL-UNITDATA request primitive attempts to append a DLSDU to an implicit queue (for this DLSAP-address and priority) of indeterminate capacity, but fails if the queue is full. If the append operation was successful, then the DLSDU will be transmitted at the first opportunity, after all preceding DLSDUs in the queue.

At the moment of the request, if the explicit or implicit FIFO queue is full, then the request is terminated and a DL-UNITDATA confirm primitive is issued immediately with an appropriate negative status.

7.5.1.2.4.2 Indication primitive

If the receiving DLS-user has bound a FIFO queue to the DL(SAP)-address at the received DLSDU's priority as a sink, and that queue is not full, then

a) the newly-received DLSDU is appended to that queue, and the DLS-user-data parameter is omitted from the associated DL-UNITDATA indication primitive. The DLSDU can be read using a DL-GET request primitive.

If no such binding exists, then

b) an implicit queue of indeterminate capacity is used as the receive queue, and the DLSDU is delivered as the DLS-user-data parameter of the associated DL-UNITDATA indication primitive.

If it is not possible to append the received DLSDU to the implicit or explicit receive queue, then

c) the DLSDU is discarded and a DL-UNITDATA indication primitive is not issued to the DLS-user.

A DL-UNITDATA indication primitive reporting the DLSDU's receipt occurs at the receiving DLS-user's DLSAP.

7.5.1.2.5 Status

This parameter allows the DLS-user to determine whether the requested service was provided successfully, or failed for the reason specified. The value conveyed in this parameter is as follows:

a) "success";
b) "failure — sending queue full";
c) "failure — no requesting DLS-user data specified";
d) "failure — requested QoS unavailable";
e) "failure — calling DLSAP DL-identifier is invalid";
f) "failure — incompatible DL(SAP)-role for calling address";
g) "failure — incompatible DL(SAP)-role for called address";
h) "failure — terminated by unbind of source DLSAP-address";
j) "failure — resource limitation in responder";
k) "failure — fault in responder";
m) "failure — timeout before transmission";
n) "failure — timeout after transmission, before acknowledgment"; or
p) "failure — reason unspecified".

NOTE 1 Failure g) can result from specifying a group DL-address as the called address, and requiring remote DLE confirmation.

NOTE 2 Addition to, or refinement of, this list of values to convey more specific diagnostic and management information is permitted in a DL-protocol standard that provides services as specified in this clause.

7.5.1.3 Sequence of primitives

The sequence of primitives in a successful unitdata transfer is defined in the time-sequence diagrams in Figure 65 through Figure 67.

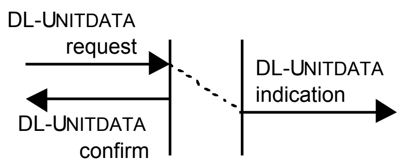


Figure 65 – Sequence of primitives for a successful locally-acknowledged connectionless-mode unitdata transfer

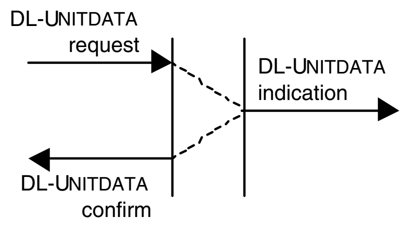


Figure 66 – Sequence of primitives for a successful remotely-acknowledged connectionless-mode unitdata transfer

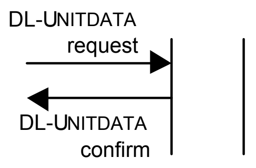


Figure 67 – Sequence of primitives for an unsuccessful connectionless-mode unitdata transfer

7.5.2 Data exchange

NOTE DL-connectionless-mode data exchange services are an extension of the Unitdata services specified in ISO/IEC 7498-1.

7.5.2.1 Function

The DL-connectionless-mode unitdata exchange service is invoked through use of the DL-COMPEL-SERVICE service (see 8.5.2). The DL-connectionless-mode unitdata exchange service can be used

a) to send a self-contained DLSDU from one local DLSAP, the initiator, to another DLSAP, the responder;

b) to cause a second independent self-contained DLSDU, which is ready for transmission at that responder DLSAP, to be returned to the initiator DLSAP; and

c) when neither DLSAP has a DLSDU ready to transmit, to notify the associated DLS-users that such an exchange was attempted.

The DLSDUs are independent in the sense that they bear no relationship to any other DLSDUs transmitted through other invocations of the DLS. The DLSDUs are self-contained in that all the information required to deliver each DLSDU is presented to the DLS-provider, together with the user data to be transmitted, in a single DL-PUT (see 5.4.5) service access. Thus no initial establishment or subsequent release of a DLC is required.

A DLSDU transmitted using DL-connectionless-mode data exchange is not considered by the DLS-provider to be related in any way to any other DLSDU. Although the DLS maintains the integrity of individual DLSDUs, it does not necessarily deliver them to the receiving (responding or initiating) DLS-user in the order in which they are presented by the sending (initiating or responding, respectively) DLS-user.

NOTE The possibility, probability and reasons for such misordering are protocol-specific.

Limited means are provided by which the receiving DLS-user may control the rate at which the sending DLS-user may send DLSDUs (peer-to-peer flow control).

1) The initiating DLS-user limits the rate at which both it and the responding DLS-user can send DLSDUs by controlling the frequency of the DL-Unitdata-Exchange service.

2) The responding DLS-user can limit the rate at which it can receive DLSDUs of a specified priority by explicitly binding a queue at that priority as a sink, and keeping the queue full. The initiating DLS-user will be informed that the responding DLE discarded the received DLSDU (due to resource limitations), and may use this information as a form of back-

pressure flow control.

The DLS-provider will not maintain any state information about the flow of information between DLSAPs. Flow control exerted by the DLS-provider upon a sending DLS-user can only be described for a specific interface.

7.5.2.2 Types of primitives and parameters

Table 24 indicates the primitive and the parameters used by the DL-connectionless-mode unitdata exchange service. This service may occur between any DLSAP-address (the initiator) whose binding DL(SAP)-role is INITIATOR, and any DLSAP-address (the responder)

a) whose binding DL(SAP)-role is UNCONSTRAINED RESPONDER; or
b) whose binding DL(SAP)-role is CONSTRAINED RESPONDER and whose associated remote-DL(SAP)-address as specified in the relevant DL-BIND request primitive (see 5.4.3.2.3.2) is equal to the initiator-DLSAP-address.

Table 24 – DL-connectionless-mode unitdata exchange primitive and parameters

DL-UNITDATA-EXCHANGE	Indication at INITIATOR	Indication at RESPONDER
Parameter name	**output**	**output**
Local address	M (1)	M (2)
Remote address	M (2)	M (1)
QoS parameter set		
DLL priority of sent DLS-user data	C (3)	C (4)
DLL priority of received DLS-user data	C (4)	C (3)
Buffer-or-queue DLS-user-identifier	C	C
Status	M	M

NOTE 1 These two parameters designate the same DLSAP-address.

NOTE 2 These two parameters designate the same DLSAP-address.

NOTE 3 These two DLL priorities are equal.

NOTE 4 These two DLL priorities are equal.

7.5.2.2.1 Addresses

The parameters that take addresses as values (7.5.2.2.1.1, 7.5.2.2.1.2) both refer to DLSAP-addresses — one which has a DL(SAP)-role of INITIATOR, and one which has a DL(SAP)-role of CONSTRAINED RESPONDER or UNCONSTRAINED RESPONDER.

7.5.2.2.1.1 Local address

This parameter conveys an address identifying the local DLSAP at which the DLS occurred. It takes the form of a DLSAP-address DLS-user-identifier in the indication primitive.

7.5.2.2.1.2 Remote address

This parameter conveys an address identifying the peer DLSAP at which the DLS occurred. It takes the form of a DLSAP-address in the indication primitive.

NOTE If the DLS-user has issued a DL-BIND request for the Remote Address, then this parameter also can take the form of a DLSAP-address DLS-user-identifier in the indication primitive.