

AS/NZS 3646.3 Supp 1:1995

ISO 8571-3:1988/Amd.2:1993

AS/NZS 3646.3 Supplement 1:1995

**Information processing systems—
Open Systems Interconnection—
File transfer, access and
management**

**Part 3: File service definition—
Overlapped access**

**(Supplement to
AS/NZS 3646.3:1993)**

[ISO title: Information processing systems—Open Systems
Interconnection—File Transfer, Access and Management
Part 3: File Service Definition, Amendment 2: Overlapped access]



AS/NZS 3646.3 Supp 1:1995

This Joint Australian/New Zealand Standard was prepared by Joint Technical Committee IT/1, Information Systems—Interconnection. It was approved on behalf of the Council of Standards Australia on 22 September 1994 and on behalf of the Council of Standards New Zealand on 9 May 1994. It was published on 5 January 1995.

The following interests are represented on Committee IT/1:

Australian Bankers Association
 Australian Bureau of Statistics
 Australian Chamber of Commerce and Industry
 Australian Committee of Directors and Principals
 Australian Computer Society
 Australian Computer Users Association
 Australian Information Industry Association
 Australian Vice Chancellors Committee
 Department of Defence, Australia
 Department of Industry, Science and Technology, Australia
 Government Computing Service, New Zealand
 Information Exchange Steering Committee, Australia
 Institute of Information and Communication Technologies, CSIRO, Australia
 OPTUS Communications, Australia
 Telecom Australia
 Telecom New Zealand

Review of Standards. To keep abreast of progress in industry, Joint Australian/New Zealand Standards are subject to periodic review and are kept up to date by the issue of amendments or new editions as necessary. It is important therefore that Standards users ensure that they are in possession of the latest edition, and any amendments thereto.

Full details of all Joint Standards and related publications will be found in the Standards Australia and Standards New Zealand Catalogue of Publications; this information is supplemented each month by the magazines 'The Australian Standard' and 'Standards New Zealand', which subscribing members receive, and which give details of new publications, new editions and amendments, and of withdrawn Standards.

Suggestions for improvements to Joint Standards, addressed to the head office of either Standards Australia or Standards New Zealand, are welcomed. Notification of any inaccuracy or ambiguity found in a Joint Australian/New Zealand Standard should be made without delay in order that the matter may be investigated and appropriate action taken.

AS/NZS 3646.3 Supp 1:1995

AS/NZS 3646.3 Supplement 1:1995

**Information processing systems—
Open Systems Interconnection—
File transfer, access and
management**

**Part 3: File service definition—
Overlapped access**

**(Supplement to
AS/NZS 3646.3:1993)**

First published as Joint Standard
AS/NZS 3646.3 Supp 1:1995.

PUBLISHED JOINTLY BY:

STANDARDS AUSTRALIA
1 The Crescent,
Homebush NSW 2140 Australia

STANDARDS NEW ZEALAND
Level 10, Standards House,
155 The Terrace,
Wellington 6001 New Zealand

PREFACE

This Standard was prepared by the Joint Standards Australia/Standards New Zealand Committee IT/1 on Information Systems—Interconnection. It is identical with and has been reproduced from ISO 8571-3:1988/Amd.2:1993, *Information processing systems—Open Systems Interconnection—File Transfer, Access and Management, Part 3: File Service Definition, Amendment 2: Overlapped access*.

The objective of this Standard is to provide users of the FTAM Standards with an extension of the scope to include the services offered by overlapped access.

This Standard adds to the text of AS/NZS 3646.3:1993.

This Standard is one of a series of Open Systems Interconnection (OSI) Standards which are currently under development. Since OSI Standards are developmental, there may be some minor difficulties encountered in their implementation. For this reason, Standards Australia will be providing, through the OSI Help Desk, a service to coordinate and disseminate information concerning difficulties which are identified in using this Standard.

Users of this Standard are advised by Standards Australia and Standards New Zealand, under arrangements made with ISO and IEC, as well as certain other Standards organizations, that the number of this Standard is not reproduced on each page; its identity is shown only on the cover and title pages.

For the purpose of this Standard, the source text should be modified as follows:

- (a) *Terminology* The words 'this Australian/New Zealand Standard' should replace the words 'this International Standard' wherever they appear.
- (b) *Decimal marker* Substitute a full point for a comma where it appears as a decimal marker.

© Copyright — STANDARDS AUSTRALIA/STANDARDS NEW ZEALAND

Users of Standards are reminded that copyright subsists in all Standards Australia and Standards New Zealand publications and software. Except where the Copyright Act allows and except where provided for below no publications or software produced by Standards Australia or Standards New Zealand may be reproduced, stored in a retrieval system in any form or transmitted by any means without prior permission in writing from Standards Australia or Standards New Zealand. Permission may be conditional on an appropriate royalty payment. Australian requests for permission and information on commercial software royalties should be directed to the head office of Standards Australia. New Zealand requests should be directed to Standards New Zealand.

Up to 10 percent of the technical content pages of a Standard may be copied for use exclusively in-house by purchasers of the Standard without payment of a royalty or advice to Standards Australia or Standards New Zealand.

Inclusion of copyright material in computer software programs is also permitted without royalty payment provided such programs are used exclusively in-house by the creators of the programs.

Care should be taken to ensure that material used is from the current edition of the Standard and that it is updated whenever the Standard is amended or revised. The number and date of the Standard should therefore be clearly identified.

The use of material in print form or in computer software programs to be used commercially, with or without payment, or in commercial contracts is subject to the payment of a royalty. This policy may be varied by Standards Australia or Standards New Zealand at any time.

AUSTRALIAN/NEW ZEALAND STANDARD

**Information processing systems—Open Systems Interconnection—
File transfer, access and management****Part 3: File service definition—Overlapped access**

(Supplement to AS/NZS 3646.3:1993)

0 Introduction

Clause 0 provides an introduction to this amendment. The text in this clause is not intended for inclusion in ISO 8571 part 3.

0.1 General

ISO 8571 part 3 defines in an abstract way the externally visible file transfer, access and management service.

This amendment extends this service definition to incorporate the services offered by overlapped access.

0.2 Rationale

The objective in introducing overlapped access is to allow more efficient access to structured files when a single initiator has a need to perform many reading and updating operations; the serial nature of the current FTAM data transfer services introduces a significant control overhead if the FADUs are small. In this context, an FADU is small if its transmission time is comparable with the time to complete a confirmed service on the association (the association's round trip delay).

0.3 Summary

The current design envelope that there should be at most one file selection per association and one file open per file selection is maintained. If access to more than one file is to

be overlapped, more than one association is necessary. The overlapped access takes place within a constant set of presentation contexts established as at present when the file is opened, or previously.

Two different degrees of overlap have been identified. Firstly, requests for future accesses may be issued whilst a previously requested BDT action is in progress, allowing the creation of a queue of read and write requests. In general, PCI relating to a given BDT action may be overlapped with other BDT actions, subject to restrictions; this is called consecutive access. Secondly, read and write actions can be performed in parallel, so that both directions of data transfer are exploited at any one time. Requests are then taken from the queue whenever either direction of transfer becomes free; this is called concurrent access.

The transfer of a single FADU, specified in a single F-READ request has the same interpretation as in ISO 8571. The resultant effect on the virtual filestore of a set of overlapped requests using consecutive access shall be the same as that of the equivalent set of requests issued in series; the service provided is serializable. If concurrent access is used then the resultant effect of a set of write actions on the virtual filestore, is also serializable. However, due to the non-determinism introduced by the use of concurrent access, it is also possible that, in some uses of the service, the data transferred as a result of a read action is not consistent with the current state of the file.

1 Scope and field of application

This amendment makes no additions to clause 1.

2 References

This amendment makes no additions to clause 2.

3 Definitions

This amendment makes no additions to clause 3.

4 Abbreviations

This amendment makes no additions to clause 4.

5 Conventions

This amendment makes no additions to clause 5.

Table 1 - Services and functional units of the External File Service

U15 Consecutive access	O				see note
U16 Concurrent access	O				see note

the role of sender), to the initiator (in the role of receiver);

d) the write bulk data service (see 24.2) is used by the initiator to initiate a bulk data transfer from the initiator (in the role of sender) to the responder (in the role of receiver);

e) the data unit transfer service (see 24.3) is used by the sender to transmit bulk data;

f) the end of data transfer service (see 24.4) is used by the sender to indicate completion of the data transfer;

g) the end of transfer service (see 24.5) is used by the initiator to confirm that the data transfer procedure is complete;

h) the cancel data transfer service (see 24.6) is used by either the sender or the receiver to cancel a data transfer procedure. When overlapped access is allowed either direction may be cancelled independently of the other.

7.10 Checkpointing and restarting

Add sentence to end of b):

If overlapped access is in use then either direction of transfer may be restarted.

8 Functional units and service classes

8.1 Functional units

Add the following clauses:

8.1.15 Consecutive access functional unit

The consecutive access functional unit allows the overlap of read and write data transfer procedures as defined in ISO 8571-1.

8.1.16 Concurrent access functional unit

The concurrent access functional unit allows the overlap of read and write data transfer procedures as defined in ISO 8571-1.

8.1.17 Service classes and functional units

Add entries to Table 1 and re-number entries accordingly:

Add note after Table 1:

Note - the consecutive access and concurrent access functional units allow the services provided by the read and write functional units to be overlapped. The description of the overlap allowed is contained within the definitions of individual services.

8.2 Service classes

8.2.2 File access class

Insert the following items and re-label accordingly:

d) optionally, the consecutive access functional unit;

e) optionally, the concurrent access functional unit;

9 Levels of file service

Replace Table 3 - Functional units in the file services

Insert note:

Note - Pending the specification of presentation symmetric synchronisation services, recovery mechanisms will not be available for use during overlapped access.

Table 3 - Functional units in the file service

External file service	Internal file service
Kernel	Kernel
Read	Read
Write	Write
File access	File access
Limited file management	Limited file management
Enhanced file management	Enhanced file management
Grouping	Grouping
FADU locking	FADU locking
	Recovery
	Restart data transfer
Consecutive access	Consecutive access
Concurrent access	Concurrent access

10 Negotiation of service class, FTAM QoS and functional units

Amend third paragraph:

The availability of the functional units, read, write, consecutive access, concurrent access, file access, limited file management, enhanced file management, grouping and FADU locking is controlled by the service class negotiated (see Tables 1 and 2).

Section two : Definition of file service primitives

11 File service primitives

Amend entries in Table 6.

12 Sequence of primitives

This amendment makes no additions to clause 12.

13 Common file service parameters

13.8 Concurrency Control

Replace fifth paragraph:

If FADU locking is negotiated the locks set at file open have the scope of the relevant bulk data transfer procedure. If overlapped access is not in use then this is equivalent to the bulk data transfer regime.

If overlapped access is in use then the scope of the FADU locking is that of the read and write action.

13.12 File Access Data Unit Identity

Add note.

Note: if concurrent access is in use then the FADU identifiers previous, current and next are dependent on whether the action is a read or a write.

Add the following clauses:

13.14 Bulk Transfer Number

The bulk transfer number parameter is only visible in the internal file service. The parameter is used to identify a particular bulk data transfer within an open regime. If the use of concurrent access is negotiated when the file open regime is established then the first read data transfer within an open regime is numbered 1, and subsequent read transfers are consecutively numbered. Similarly, the first write data transfer within the open regime is numbered 1, and subsequent write transfers are consecutively numbered. If overlapped access is not negotiated, or if consecutive access is negotiated, then read and write bulk transfers are numbered consecutively, starting from 1 for the first bulk data transfer procedure i.e. there is one

Table 6 - File service primitives

Primitive	Confirmed	Request by	Parameters
F-OPEN	Yes	Initiator	State result Action result Processing mode Contents type Concurrency control Shared ASE information Enable FADU locking Diagnostic Degree of overlap Transfer window [Activity identifier] [Recovery mode]
[F-RECOVER]	Yes	Initiator	[State result Action result Activity identifier Bulk transfer number Requested access Access passwords Contents type Recovery point Diagnostic Concurrent bulk transfer number Concurrent recovery point Last transfer end read request Last transfer end read response Last transfer end write request Last transfer end write response]

numbering sequence.

13.15 Transfer Number

The transfer number parameter is conditional on the use of overlapped access. The parameter is used by the initiator and responder to identify an uncompleted bulk data transfer. The value of the parameter is an integer. If overlapped access is not in use, or consecutive access is in use, then overlapping read and write transfers are consecutively numbered, starting from 1. The value is reset when all outstanding requests are completed.

Note - The transfer number of a bulk data transfer may exceed the value of the transfer window (17.1.2.10) as the latter parameter constrains the maximum number of transfers that are uncompleted at any one time.

13.16 Last transfer end read request

The last transfer end read request parameter is only visible in the internal file service, and is conditional on the use of overlapped access for the open regime. The value of the parameter is a transfer number (see 13.14). For the initiator it indicates the transfer number of the last read data transfer for which a F-TRANSFER-END request has been issued; for the responder it indicates the transfer number of the last read data transfer for which a F-TRANSFER-END indication has been received.

The default value 0 indicates that there are no outstanding F-TRANSFER-END requests or indications to be issued or received.

13.17 Last transfer end read response

The last transfer end read response parameter is only visible in the internal file service, and is conditional on the use of overlapped access for the open regime. The value of the parameter is a transfer number (see 13.13). For the initiator it indicates the transfer number of the last read data transfer for which a F-TRANSFER-END confirm has been received; for the responder it indicates the transfer number of the last read data transfer for which a F-TRANSFER-END response has been issued.

The default value 0 indicates that there are no outstanding F-TRANSFER-END responses or confirms to be issued or received.

13.18 Last transfer end write request

The last transfer end write request parameter is only visible in the internal file service, and is conditional on the use of overlapped access for the open regime. The value of the parameter is a transfer number (see 13.14). For the initiator it indicates the transfer number of the last write data transfer for which a F-TRANSFER-END request has been issued;

for the responder it indicates the transfer number of the last write data transfer for which a F-TRANSFER-END indication has been received.

The default value 0 indicates that there are no outstanding F-TRANSFER-END requests or indications to be issued or received.

13.19 Last transfer end write response

The last transfer end write response parameter is only visible in the internal file service, and is conditional on the use of concurrent access for the open regime. The value of the parameter is a transfer number (see 13.14). For the initiator it indicates the transfer number of the last write data transfer for which a F-TRANSFER-END confirm has been received; for the responder it indicates the transfer number of the last write data transfer for which a F-TRANSFER-END response has been issued.

The default value zero indicates that there are no outstanding F-TRANSFER-END responses or confirms to be issued or received.

14 FTAM regime control

This amendment makes no additions to clause 14.

15 File selection regime control

This amendment makes no additions to clause 15.

16 File management

This amendment makes no additions to clause 16.

17 File open regime control

Amend second sentence:

This regime establishes the degree of overlapped access, processing mode, presentation contexts, and concurrency control for the data transfer activity which is to be performed.

17.1 File open service

17.1.1 Function

Amend last sentence of first paragraph:

It also establishes concurrency control, the degree of overlapped access and possible processing modes.

17.1.2 Types of primitives and parameters

Replace entries in table 21.

Add the following sub clauses and re-number:

17.1.2.9 Degree Of Overlap

The degree of overlap parameter indicates the degree of overlapped access required during the file open regime: either no overlapped access (normal), consecutive access or concurrent access.

The degree of overlap available during the open regime is the lowest of the values offered by the initiator and responder. For the purposes of negotiation, the degrees of overlapped access are rated as: concurrent access (highest), consecutive access, no overlapped access (lowest). This value is used to set the current degree of overlap attribute.

17.1.2.10 Transfer Window

The transfer window parameter indicates the number of bulk data transfers that may be uncompleted at any time. There is an independent transfer window for reads and writes in concurrent access, although the value of the limit is the same, otherwise the transfer window is for both reads and writes.

The value of the transfer window is the lowest of the values offered by the initiator or responder.

18 Grouping control

This amendment makes no additions to Clause 18 of ISO 8571 part 3.

19 Recovery (Internal service only)

19.1 Regime recovery

19.1.2 Types of primitives and parameters

Replace table 24.

19.1.2.4 Bulk Transfer Number

The bulk transfer number is defined in 13.14. The parameter indicates the bulk transfer procedure that is to be recovered.

If concurrent access is in use then this parameter takes the value of the read transfer that is to be recovered (the write transfer is identified by the concurrent bulk transfer number parameter). The default value 0 indicates that there is no read bulk transfer to be recovered.

Add the following clauses.

19.1.2.9 Recovery Point

Add last paragraph.

If concurrent overlapped access is in use then the parameter is with respect to the read data transfer that is identified by the bulk data transfer number. If the bulk transfer number is zero then the recovery point parameter shall take the value zero.

19.1.2.10 Concurrent Bulk Transfer Number

The concurrent bulk transfer number parameter is conditional on the selection of the concurrent access functional unit and the degree of overlap (see 17.1.2.9) being set to concurrent access. The parameter takes the value of a bulk transfer number (defined in 13.14) and identifies the write transfer that is to be recovered (see 19.1.2.4).

The default value 0 indicates that there is no write bulk transfer to be recovered.

19.1.2.11 Concurrent Recovery Point

The concurrent recovery point parameter is conditional on the concurrent access functional unit and the degree of overlap (see 17.1.2.9) being set to concurrent access. The parameter indicates that recovery is to a point before the start of the write bulk data transfer (value zero), or to a checkpoint within the bulk data transfer, or to a point following its completion (see note). The recovery point is within the write bulk data transfer identified by the concurrent bulk transfer number (see 19.1.2.10); if the write bulk transfer number parameter is zero then the write recovery point parameter shall take the value zero. The recovery point is determined by the entity which was receiving data at the time of failure. Consequently the parameter is required to be present on the response primitive.

Table 21 - F-OPEN parameters

Degree of overlap	Conditional	Conditional (=)	Conditional	Conditional (=)
Transfer window	Conditional	Conditional (=)	Conditional	Conditional (=)

If the concurrent bulk transfer number is zero, the concurrent recovery point parameter shall take the value zero.

NOTE - Recovery to a point following a write bulk data transfer completion is defined for the case where the receiver has received an F-DATA-END indication but the F-TRANSFER-END exchange has not completed. In such a situation it would be unnecessary to go back to the last confirmed checkpoint. The only requirement is for both sender and receiver to have an agreed understanding of the completion of the transfer. A value one larger than the highest checkpoint number issued by the sender is used to indicate recovery after bulk data transfer.

19.1.2.12 Last Transfer End Read Request

The last transfer end read request parameter is defined in 13.16. It is only present on the response primitive, and therefore indicates the bulk transfer number of the last read data transfer for which a transfer end read indication primitive had been received by the responder. It is only present if recovery is to an open regime that had negotiated the use of concurrent access.

19.1.2.13 Last Transfer End Read Response

The last transfer end read response parameter is defined in 13.17. It is only present on the request primitive, and therefore indicates the bulk transfer number of the last read data transfer for which a transfer end read confirm primitive had been received by the initiator. It is only present if recovery is to an open regime that had negotiated the use of concurrent access.

19.1.2.14 Last Transfer End Write Request

The last transfer end write request parameter is defined in 13.18. It is only present on the response primitive, and therefore indicates the bulk transfer number of the last write data transfer for which a transfer end read indication primitive had been received by the responder. It is only present if recovery is to an open regime that had negotiated the use of concurrent access.

19.1.2.15 Last Transfer End Write Response

The last transfer end write response parameter is defined in 13.19. It is only present on the request primitive, and therefore indicates the bulk transfer number of the last write data transfer for which a transfer end read confirm primitive had been received by the initiator. It is only present if recovery is to an open regime that had negotiated the use of concurrent access.

Replace Table 24.

20 Access to file contents

20.1 Bulk data transfer

Replace second paragraph:

When overlapped access is not in use, these procedures start and finish in a single data transfer idle state, and so can be considered as a self-contained procedural unit, which is primitive in the definition of the remainder of the file service.

If overlapped access is in use then these procedures may be overlapped. It is therefore not necessarily true that read and write procedures start and finish in a data transfer idle state.

21 Bulk data transfer service primitives

Replace Table 30.

Add sentence to end of fourth paragraph:

If overlapped access is in use then the initiator and responder may each be acting as both sender and receiver, with respect to the read and write procedures, as previously described.

Table 24 - F-RECOVER parameters

Parameter	F-RECOVER request	F-RECOVER indication	F-RECOVER response	F-RECOVER confirm
State Result			Mandatory	Mandatory
Action Result			Mandatory	Mandatory
Activity Identifier	Mandatory	Mandatory (=)		
Bulk Transfer number	Mandatory	Mandatory (=)		
Requested Access	Mandatory	Mandatory (=)		
Access Passwords	Optional	Optional (=)		
Contents Type			Mandatory	Mandatory (=)
Recovery Point	Conditional	Conditional (=)	Conditional	Conditional (=)
Diagnostic			Optional	Optional
Concurrent Bulk Transfer Number	Conditional	Conditional (=)	Conditional	Conditional (=)
Concurrent Recovery Point	Conditional	Conditional (=)	Conditional	Conditional (=)
Last Transfer End Read Request			Conditional	Conditional (=)
Last Transfer End Read Response	Conditional	Conditional (=)		
Last Transfer End Write Request			Conditional	Conditional (=)
Last Transfer End Write Response	Conditional	Conditional (=)		

Section three: Definition of bulk data transfer primitives

22 Sequence of bulk data transfer primitives

22.1 Normal sequences

The normal progress of individual read and write bulk data transfer procedures is illustrated by the state transition diagram shown in Figure 8. Full state transition diagrams are contained in Annex E.

The sequences of primitives allowed as a result of the use

of the bulk data transfer service with overlapped access, is specified in Annex F.

Note - The use of a common Data Transfer Idle state for read and write bulk data transfer procedures in figure 8, serves to illustrate the progress of use of the bulk transfer service itself when overlapped access is not in use.

Re-label figure 8.

Figure 8 - Simplified State Transition Diagram for Read and Write Bulk Data Transfer Procedures

22.2 Constraints on issue of primitives

Table 30 - Bulk data transfer service primitives

Primitive Name	Confirmed Service	Request by	Parameters	Failure Notification
F-READ	No	Initiator	Bulk data transfer specification Transfer number	F-DATA-END action result
F-WRITE	No	Initiator	Bulk data transfer specification Transfer number	F-CANCEL
F-DATA	No	Sender	Data value	F-CANCEL
F-DATA-END	No	Sender	Action result Diagnostic	F-CANCEL
F-TRANSFER-END	Yes	Initiator	Action result Shared ASE information Diagnostic Request type Transfer number [Last transfer end read response] [Last transfer end write response]	action result
F-CANCEL	Yes	Either	Action result Shared ASE information Diagnostic Request type Transfer number [Last transfer end read request] [Last transfer end read response] [Last transfer end write request] [Last transfer end write response]	none
[F-CHECK]	Yes	Either	[Checkpoint identifier Transfer number]	F-CANCEL
[F-RESTART]	Yes	Either	[Checkpoint identifier Request type Transfer number Last transfer end read request Last transfer end read response Last transfer end write request Last transfer end write response]	F-CANCEL

The primitives may be issued in any sequence consistent with the LOTOS specification in Annex F. In addition, for the non-overlapped case, the constraints are presented in tables 31 and 32. The sequences given with the individual primitive definitions apply.

22.3 Key to tables 31 and 32

This amendment makes no changes to sub-clause 22.3.

23 Common bulk data transfer parameters

23.2 Checkpoint identifier

Replace last sentence of first paragraph:

If the use of concurrent access has been negotiated for the open regime then there may be more than one checkpoint number sequence valid at any one time. The bulk transfer number is used to identify the correct sequence. For the F-RESTART primitive, the value is between 0 and 999998 for each direction of transfer.

23.43 Request type

The request type parameter identifies the type of transfer to which a primitive is related i.e. read or write.

24 Bulk data transfer

Replace the first paragraph:

This group of services performs the transfer of bulk data. Each procedure begins with the service initiator issuing either an F-READ request or an F-WRITE request as appropriate. This leads to the issue of a sequence of F-DATA requests followed by an F-DATA-END request by the sender of the data. A procedure is completed by the initiator issuing an F-TRANSFER-END request, as appropriate. If overlapped access has been negotiated, then the service primitives for read and write procedures may be interspersed, subject to the constraints defined for consecutive access and concurrent access. The primitives are defined in 24.1 to 24.6, and the valid sequences, with and without overlapped access, are defined in 24.7 and 24.8.

24.1 Read bulk data service

Replace sub-clause 24.1.1:

24.1.1 Function

The F-READ service specifies a data transfer from the service responder (that is the sender) to the service initiator (that is the receiver).

24.1.1.1 Service without overlapped access

Only one F-READ procedure may be in progress at any time on a single application association. The direction of data flow established continues until the exchange of F-TRANSFER-END primitives. Rejection of an F-READ indication is by issue of an F-DATA-END with an action result indicating unsuccessful.

These primitives signal a transfer of control from the initiator to the sender. They mark a reversal of the service asymmetry for the duration of the data transfer.

24.1.1.2 Service with consecutive access

F-READ procedures may be requested at any time during the open regime, up to the limit negotiated as the transfer window when the open regime was established. The requests for bulk transfer procedures are queued by the responder. If there are no outstanding bulk transfer requests then the F-DATA requests may be issued immediately, otherwise the F-DATA requests are issued subject to the following constraints:

- a) if the preceding bulk transfer procedure was a read procedure then the responder may issue F-DATA requests after issuing the F-DATA-END request for the preceding read procedure;
- b) if the preceding bulk transfer procedure was a write procedure then the responder may issue F-DATA requests after issuing the F-TRANSFER-END response for that write procedure.

A F-READ procedure concludes with the exchange of F-TRANSFER-END primitives.

Rejection of an F-READ indication is by issue of an F-DATA-END request with an action result indicating unsuccessful, although this may not be issued until the responder has issued a F-DATA-END indication or F-TRANSFER-END (write) response, as appropriate, for the preceding bulk transfer procedure.

24.1.1.3 Service with concurrent access

F-READ procedures may be requested at any time during the open regime, up to the limit negotiated as the transfer window when the open regime was established. The responder maintains two independent queues for the read and write bulk transfer requests. If there are no outstanding requests for read procedures then the responder may issue F-DATA requests immediately, otherwise the F-DATA requests are issued after the issue of an F-DATA-END

request for the preceding read procedure. A F-READ procedure concludes with the exchange of F-TRANSFER-END primitives.

Rejection of an F-READ indication is by issue of an F-DATA-END request with an action result indicating unsuccessful, although this may not be issued until the responder has issued a F-DATA-END indication for the preceding read procedure.

24.1.2 Types of primitives and parameters

Replace table 33.

Add sub-clause:

24.1.2.2 Transfer Number

The transfer number parameter is defined in 13.15.

24.2 Write bulk data service

Replace sub-clause 24.2.1:

24.2.1 Function

The F-WRITE service specifies a data transfer from the service initiator (that is the sender) to the service responder (that is the receiver).

24.2.1.1 Service without overlapped access

Only one F-WRITE procedure may be in progress at any time on a single application association. The direction of data flow established continues until the exchange of F-TRANSFER-END primitives. An F-WRITE indication can be rejected by issuing an F-CANCEL request (see 24.6). If the transfer is rejected, no further F-DATA indication primitives are received by the responder.

24.2.1.2 Service with consecutive overlapped access

F-WRITE procedures may be requested at any time during the open regime, up to the limit negotiated as the transfer window when the open regime was established. The requests for bulk transfer procedures are queued. If there are no outstanding bulk transfer requests then the F-DATA requests may be issued immediately, otherwise the F-DATA requests are issued subject to the following constraints:

Table 33 - F-READ parameters

Parameter	F-READ request	F-READ indication
Bulk data transfer specification	Mandator	Mandatory (=)
Transfer number	Conditional	Conditional (=)

a) if the preceding bulk transfer procedure was a read procedure then the initiator may issue F-DATA requests after issuing a F-TRANSFER-END request for the preceding read procedure;

b) if the preceding bulk transfer procedure was a write procedure then the initiator may issue F-DATA requests after issuing a F-TRANSFER-END request for the preceding write procedure.

An F-WRITE procedure concludes with the exchange of F-TRANSFER-END primitives.

A responder may reject an F-WRITE indication by issuing an F-CANCEL request, although this may not be issued until the responder has received an F-TRANSFER-END (read) indication or issued an F-TRANSFER-END (write) response, as appropriate, for the preceding bulk transfer procedure (see 24.7).

24.2.1.3 Service with concurrent access

F-WRITE procedures may be requested at any time during the open regime, up to the limit negotiated as the transfer window when the open regime was established. Two independent queues for the read and write bulk transfer requests. If there are no outstanding requests for write procedures then the initiator may issue F-DATA requests immediately, otherwise the F-DATA requests are issued after the issue of an F-TRANSFER-END request for the preceding write procedure. A F-WRITE procedure concludes with the exchange of F-TRANSFER-END primitives.

Rejection of an F-WRITE indication is by issue of an F-CANCEL request, although this may not be issued until the responder has issued a F-TRANSFER-END response, for the preceding write procedure (see 24.6).

24.2.2 Types of primitives and parameters

Replace table 34.

Add sub-clause:

24.2.2.2 Transfer Number

The transfer number parameter is defined in 13.15.

24.4 End of data transfer service

24.4.1 Function

Replace last sentence:

The sender may issue an F-DATA-END request with an unsuccessful action as a rejection of an F-READ indication, subject to the constraints defined in 24.1.

24.4.2 Types of primitives and parameters

Table 34 - F-WRITE parameters

Parameter	F-WRITE request	F-WRITE indication
Bulk Data Transfer Specification	Mandatory	Mandatory (=)
Transfer Number	Conditional	Conditional (=)

24.4.2.1 Action Result

Replace last sentence:

The initiator responds with an F-TRANSFER-END request as in the non error case.

24.5 End of transfer service

24.5.2 Types of primitives and parameters

Replace Table 37.

Add the following sub-clauses:

24.5.2.4 Request Type

The request type parameter is defined in 23.3.

24.5.2.5 Transfer Number

The transfer number parameter is defined in 13.15.

24.5.2.6 Last Transfer End Read Response

The last transfer end read response parameter is defined in 13.17.

24.5.2.7 Last Transfer End Write Response

The last transfer end write response parameter is defined in 13.18.

Replace sub-clause 24.6:

24.6 Cancel data transfer

24.6.1 Function

Either of the service users may cancel a data transfer activity by issuing an F-CANCEL request primitive. The conditions for use are described in the following clauses.

24.6.1.1 Service without overlapped access

The F-CANCEL primitive may be issued during data transfer after the issue or receipt of an F-READ or F-WRITE request or indication. At the end of data transfer, it may not be issued:

a) by the initiator acting as sender after issue of an F-DATA-END request;

b) by the responder acting as receiver after issue of an F-TRANSFER-END response;

c) by the responder acting as sender after the issue of an F-DATA-END request;

d) by the initiator acting as receiver after the issue of an F-TRANSFER-END request.

If either the cancel data transfer service or the end of data transfer service is used, the data transfer regime ceases.

Table 37 - F-TRANSFER-END parameters

Parameter	F-TRANSFER-END request	F-TRANSFER-END indication	F-TRANSFER-END response	F-TRANSFER-END confirm
Action result			Mandatory	Mandatory
Shared ASE information	Optional	Optional (=)	Optional	Optional (=)
Diagnostic			Optional	Optional
Request Type	Conditional	Conditional (=)	Conditional	Conditional (=)
Transfer Number	Conditional	Conditional (=)	Conditional	Conditional (=)
[Last transfer end read response	Conditional	Conditional (=)		
Last transfer end write response	Conditional	Conditional (=)]		

Where use of these services collides, the cancel data transfer service takes precedence.

After an F-CANCEL procedure the two users may have different views of the state of the activity. The F-CANCEL primitives interrupt any activity in progress (including an F-RESTART sequence) and any undelivered indications or confirms may be discarded.

The file remains open after a sequence of F-CANCEL primitives, although the result of interrupted operations is not defined. Further F-READ or F-WRITE operations, not necessarily related to any previous read or write attempt, may be attempted after the completion of the sequence of F-CANCEL primitives has disposed of any previous activity.

24.6.1.2 Service with consecutive access

The cancel data transfer service may be used to cancel either a read transfer or a write transfer. If there are no uncompleted bulk transfer procedures then the constraints on the use of the service are as defined for the service without overlapped access. Otherwise, it may not be used:

- a) by the initiator acting as sender before the issue of an F-TRANSFER-END request (as appropriate) for the preceding bulk transfer procedure, or after the issue of an F-DATA-END request for that write transfer;
- b) by the responder acting as receiver before the receipt of an F-TRANSFER-END (read) indication or the issue of an F-TRANSFER-END (write) response (as appropriate) for the preceding bulk transfer procedure, or after the issue of an F-TRANSFER-END response for that write transfer;
- c) by the responder acting as sender before the issue of an F-DATA-END request or F-TRANSFER-END (write) response (as appropriate) for the preceding bulk transfer, or after the issue of an F-DATA-END request for that read transfer;
- d) by the initiator acting as receiver before the issue of an F-TRANSFER-END (read) request or receipt of an F-TRANSFER-END (write) confirm (as appropriate) for the preceding bulk transfer procedure, or after the issue of an F-TRANSFER-END request for that read transfer.

Outstanding F-TRANSFER-END primitives may not be exchanged until the F-CANCEL sequence has been completed. Also, the initiator may not issue further requests for bulk transfer procedures during the exchange of F-CANCEL primitives.

If the cancel data transfer service is used then all outstanding bulk transfer requests are cancelled and the data transfer regime ceases. Where uses of this service and the end of data transfer service collide, the cancel data transfer service takes precedence.

After an F-CANCEL procedure the two users may have

different views of the state of the activity. The F-CANCEL primitives interrupt any activity in progress (including an F-RESTART sequence) and any undelivered indications or confirms may be discarded.

The file remains open after a sequence of F-CANCEL primitives, although the result of interrupted operations is not defined. Further F-READ or F-WRITE operations, not necessarily related to any previous read or write attempt, may be attempted after the completion of the sequence of F-CANCEL primitives has disposed of any previous activity.

24.6.1.3 Service with concurrent access

The cancel data transfer may be used to cancel a read transfer or a write transfer. The constraints that apply to cancelling read and write transfers are independent of each other and the F-CANCEL service may only be used to cancel one type of transfer. To cancel both the current read transfer and the current write transfer, the file service user must issue two F-CANCEL requests.

24.6.1.3.1 Cancel write bulk transfer constraints

If there are no uncompleted write bulk transfer procedures the constraints on the use of the service are as defined for the service without overlapped access. Otherwise, the service may not be used to cancel a write transfer:

- a) by the initiator before the receipt of an F-TRANSFER-END confirm for the preceding write transfer, or after the issue of an F-DATA-END request for that write transfer;
- b) by the responder before the issue of an F-TRANSFER-END response for the preceding write transfer, or after the issue of an F-TRANSFER-END response for that write transfer.

Outstanding F-TRANSFER-END (write) primitives may not be exchanged until the F-CANCEL sequence has been completed. Also, the initiator may not issue further requests for write bulk transfer procedures during the exchange of F-CANCEL primitives.

If the cancel data transfer service is used then all outstanding write bulk transfer requests are cancelled and the data transfer regime ceases. Where uses of this service and the end of data transfer service collide, the cancel data transfer service takes precedence.

After an F-CANCEL procedure the two users may have different views of the state of the activity. The F-CANCEL primitives interrupt any activity in progress (including an F-RESTART sequence) and any undelivered indications or confirms may be discarded.

The file remains open after a sequence of F-CANCEL primitives, although the result of interrupted operations is not defined. Further F-WRITE operations, not necessarily

related to any previous write attempt, may be attempted after the completion of the sequence of F-CANCEL primitives has disposed of any previous activity.

24.6.1.3.2 Cancel read bulk transfer constraints

If there are no uncompleted read bulk transfer procedures then the constraints on the service user are as defined for the service without overlapped access. Otherwise, the service may not be used to cancel a read transfer:

a) by the initiator before the issue of an F-TRANSFER-END request for the preceding read bulk transfer procedure, or after the issue of an F-TRANSFER-END request for that procedure;

b) by the responder before the receipt of an F-TRANSFER-END indication for the preceding read bulk transfer procedure, or after the issue of an F-DATA-END request for that read transfer.

Outstanding F-TRANSFER-END (read) primitives may not be exchanged until the F-CANCEL sequence has been completed. Also, the initiator may not issue further requests for read bulk transfer procedures during the exchange of F-CANCEL primitives.

If the cancel data transfer service is used then all outstanding read bulk transfer requests are cancelled and the data transfer regime ceases. Where uses of this service and the end of data transfer service collide, the cancel data transfer service takes precedence.

After an F-CANCEL procedure the two users may have different views of the state of the activity. The F-CANCEL primitives interrupt any activity in progress (including an F-RESTART sequence) and any undelivered indications or confirms may be discarded.

The file remains open after a sequence of F-CANCEL primitives, although the result of interrupted operations is not defined. Further F-READ operations, not necessarily related to any previous read attempt, may be attempted after the completion of the sequence of F-CANCEL primitives has disposed of any previous activity.

24.6.2 Types of primitives and parameters

Replace table 38.

Add the following sub-clauses:

24.6.2.4 Request Type

The request type parameter is defined in 23.3.

24.6.2.5 Transfer Number

The transfer number parameter is defined in 13.15.

24.6.2.7 Last Transfer End Read Request

The last transfer end read request parameter is defined in 13.16.

24.6.2.8 Last Transfer End Read Response

The last transfer end read response parameter is defined in 13.17.

24.6.2.9 Last Transfer End Write Request

The last transfer end write request parameter is defined in

Table 38 - F-CANCEL parameters

Parameter	F-CANCEL request	F-CANCEL indication	F-CANCEL response	F-CANCEL confirm
Action result	Mandatory	Mandatory	Mandatory	Mandatory
Shared ASE information	Optional	Optional (=)	Optional	Optional (=)
Diagnostic	Optional	Optional	Optional	Optional
Request type	Conditional	Conditional (=)	Conditional	Conditional (=)
Transfer number	Conditional	Conditional (=)	Conditional	Conditional (=)
[Last transfer end read request	Conditional (Responder only)	Conditional (=)	Conditional (Initiator only)	Conditional (=)
Last transfer end read response	Conditional (Initiator only)	Conditional (=)	Conditional (Responder only)	Conditional (=)
Last transfer end write request	Conditional (Responder only)	Conditional (=)	Conditional (Initiator only)	Conditional (=)
Last transfer end write response	Conditional (Initiator only)	Conditional (=)	Conditional (Responder only)	Conditional (=)]

13.18.

24.6.2.10 Last Transfer End Write Response

The last transfer end write response parameter is defined in 13.19.

24.7 Sequence of primitives on write

Add the following after first sentence:

If overlapped access is in use then the resultant sequence of primitives is the result of interleaving the two sequences for read and write operations. The rules for interleaving are defined for each individual primitive.

24.8 Sequence of primitives on Read

Add the following after first sentence:

If overlapped access is in use then the resultant sequence of primitives is the result of interleaving the two sequences for read and write operations. The rules for interleaving are defined for each individual primitive.

25 Checkpointing and restart (Internal BDT Service Only)

25.1 Checkpointing

Replace last sentence:

The checkpointing and restart primitives may be issued while a data transfer activity is in progress to control the progress of the data transfer.

25.1.1 Function

Add following paragraphs:

If overlapped access is not in use then checkpoint primitives may be issued after an F-READ or F-WRITE and before an F-DATA-END, for the sender and F-TRANSFER-END (read) or F-TRANSFER-END (write) for the recipient.

If the use of overlapped access has been negotiated for the open regime then checkpoints for read and write data

transfers are independent of each other. A sender may only issue an F-CHECK request if all data has been sent for the preceding bulk data transfers; a responder may only issue an F-CHECK response if all data has been secured for the preceding bulk data transfer.

25.1.2 Types of primitives and parameters

Replace table 39.

Add the following sub-clause:

25.1.2.2 Transfer Number

The transfer number is defined in 13.15. The parameter identifies the data transfer to which the checkpoint is associated.

25.2 Restarting data transfer service

Replace sub-clause 25.2.1:

25.2.1 Function

The F-RESTART group of primitives interrupts any bulk data transfer activity in progress, with possible loss of any undelivered indications or confirms. It negotiates a point at which data transfer is to be restarted.

25.2.1.1 Service without overlapped access

At the end of data transfer, the service may not be used:

- a) by the initiator acting as sender after issue of an F-DATA-END request;
- b) by the responder acting as receiver after the issue of an F-TRANSFER-END response;
- c) by the responder acting as sender after the issue of an F-DATA-END request;
- d) by the initiator acting as receiver after the issue of an F-TRANSFER-END request.

If the receiver and the sender of data both issue an F-RESTART request, the service provider resolves the

Table 39 - F-CHECK parameters

Parameter	F-CHECK request	F-CHECK indication	F-CHECK response	F-CHECK confirm
Checkpoint identifier	Mandatory	Mandatory (=)	Mandatory	Mandatory (=)
Transfer number	Conditional	Conditional (=)	Conditional	Conditional (=)

collision and issues a confirmation to each user giving the point at which the transfer is to be restarted (see ISO 8571-4). An F-RESTART indication is rejected by using an F-CANCEL request primitive.

25.2.1.2 Service with consecutive access

The restarting data transfer service may be used to restart either a read transfer or a write transfer. If there are no uncompleted bulk transfer procedures then the constraints on the use of the service are as defined for the service without overlapped access. Otherwise, it may not be used:

a) by the initiator acting as sender before the issue of an F-TRANSFER-END request for the preceding bulk transfer procedure, or after the issue of an F-DATA-END request for that write transfer;

b) by the responder acting as receiver before the receipt of an F-TRANSFER-END (read) indication or the issue of an F-TRANSFER-END (write) response (as appropriate) for the preceding bulk transfer procedure, or after the issue of an F-TRANSFER-END response for that write transfer;

c) by the responder acting as sender before the issue of an F-DATA-END request (read) or F-TRANSFER-END (write) response, as appropriate, for the preceding bulk transfer, or after the issue of an F-DATA-END request for that read transfer;

d) by the initiator acting as receiver before the issue of an F-TRANSFER-END (read) request or receipt of an F-TRANSFER-END (write) confirm, as appropriate, for the preceding bulk transfer procedure, or after the issue of an F-TRANSFER-END request for that read transfer.

If the restarting data transfer service is used then all outstanding bulk transfer requests are cancelled and the initiator must reissue the bulk transfer requests. Where use of this service causes a collision, the service provider resolves the collision and issues a confirmation to each user giving the transfer, and the point at which the transfer is to be restarted (see ISO 8571-4). An F-RESTART indication is rejected by using an F-CANCEL request primitive.

25.2.1.3 Service with concurrent access

The restarting data transfer may be used to restart a read transfer or a write transfer. The constraints that apply to restarting read and write procedures are independent of each other. To issue an F-RESTART request for a read transfer and a write transfer, a service user must satisfy both sets of constraints.

25.2.1.3.1 Restart write bulk transfer constraints

If there are no uncompleted write bulk transfer procedures

then the constraints on the use of the service are as defined for the service without overlapped access. Otherwise, the service may not be used to restart a write transfer:

a) by the initiator before the issue of an F-TRANSFER-END request for the preceding write transfer, or after the issue of an F-DATA-END request for that write transfer;

b) by the responder before the issue of an F-TRANSFER-END response for the preceding write transfer, or after the issue of an F-TRANSFER-END response for that write transfer.

25.2.1.3.2 Restart read bulk transfer constraints

If there are no uncompleted read bulk transfer procedures then the constraints on the service user are as defined for the service without overlapped access. Otherwise, the service may not be used to restart a read transfer:

a) by the initiator before the issue of an F-TRANSFER-END request for the preceding read bulk transfer procedure, or after the issue of an F-TRANSFER-END request for that read procedure;

b) by the responder before the issue of an F-DATA-END request for the preceding read bulk transfer procedure, or after the issue of an F-DATA-END request for that read transfer.

Subject to the following, a bulk transfer procedure that is not in a direction to be restarted, may continue as normal:

i) outstanding F-TRANSFER-END primitives may not be exchanged until the F-RESTART sequence has been completed;

ii) the initiator may not issue further requests for bulk transfer procedures during the exchange of F-RESTART primitives.

If the receiver and the sender of data both issue an F-RESTART request for the same direction, the service provider resolves the collision and issues a confirmation to each user giving the transfer, and the point at which the transfer is to be restarted (see ISO 8571-4).

25.2.2 Types of primitives and parameters

Replace table 40.

Add the following clauses:

25.2.2.2 Request Type

The request type parameter is defined in 23.4.

25.2.2.3 Transfer Number

The transfer number is defined in 13.15.

The last transfer end read response parameter is defined in 13.17.

25.2.2.4 Last Transfer End Read Request

The last transfer end read request parameter is defined in 13.16.

25.2.2.6 Last Transfer End Write Request

The last transfer end write request parameter is defined in 13.18.

25.2.2.5 Last Transfer End Read Response

25.2.2.7 Last Transfer End Write Response

The last transfer end write response parameter is defined in 13.19.

Table 40 - F-RESTART parameters

Parameter	F-RESTART request	F-RESTART indication	F-RESTART response	F-RESTART confirm
Checkpoint identifier	Mandatory	Mandatory (=)	Mandatory	Mandatory (=)
Request type	Conditional	Conditional (=)	Conditional	Conditional (=)
Transfer number	Mandatory	Mandatory (=)	Mandatory	Mandatory (=)
Last transfer end read request	Conditional (Responder only)	Conditional (=)	Conditional (Initiator only)	Conditional (=)
Last transfer end read response	Conditional (Initiator only)	Conditional (=)	Conditional (Responder only)	Conditional (=)
Last transfer end write request	Conditional (Responder only)	Conditional (=)	Conditional (Initiator only)	Conditional (=)
Last transfer end write response	Conditional (Initiator only)	Conditional (=)	Conditional (Responder only)	Conditional (=)

Annex A
Diagnostic parameter values

(This annex forms part of the standard.)

Amend table 48:

Table 48 - Access related diagnostics

Type	Identifier	Observer	Source	Reason
01	5nnn	5	5	Degree of overlap not available
0 2	5nnn	5	1	Transfer window - too large
0 2	5nnn	5	1	Transfer window - too small

Amend table 49:

Type	Identifier	Observer	Source	Reason
012	5nnn	5	5	Degree of overlap not available

The identifiers 5nnn are the next numbers in the sequence of identifiers for diagnostics.

Annex B
Relation of attributes to primitives
(This annex forms part of the standard.)

Amend table 51:

Table 51 - Activity attributes

Attribute Name	F-INITIALIZE	F-SELECT	F-CREATE	F-OPEN	F-READ & F-WRITE	F-LOCATE & F-ERASE	F-RECOVER
current degree of overlap				change			change

Annex E

State transition diagrams

(This annex forms part of the standard.)

Replace first sentence.

This annex contains descriptions of the sequences of service primitives in the external file service. State transition diagrams are used to define all sequences of primitives except those in overlapped access. The sequences of bulk data transfer primitives in overlapped access are defined by the LOTOS specifications in Annex F. This annex contains diagrams that illustrate the structure of the specifications and identify those events that (may) result in a change in the

allowed behaviour i.e. the 'transition' between LOTOS processes (see note 1).

Note 1 - The notion of 'transition' between different processes is used in an informal manner for the purpose of this introduction. A transition relates to the situation in the LOTOS specifications where the use of a service primitive does not result in the same LOTOS process being instantiated.