INTERNATIONAL STANDARD

ISO 8571-4

First edition
1988-10-01

AMENDMENT 2
1993-08-15

Information processing systems – Open Systems Interconnection – File Transfer, Access and Management –

Part 4 : File Protocol Specification

AMENDMENT 2 : Overlapped access

Systèmes de traitement de l'information – Interconnexion de systèmes ouverts – Transfert, accès et gestion de fichiers –

Partie 4 : Spécification du protocole de fichiers

AMENDEMENT 2 : Chevauchement d'accès

Reference number
ISO 8571-4:1988/Amd.2:1993 (E)

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Amendment 2 to International Standard ISO 8571-4:1988 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

ISO 8571 consists of the following parts, under the general title *Information processing systems – Open Systems Interconnection – File Transfer, Access and Management*:

- *Part 1 : General introduction*
- *Part 2 : Virtual Filestore Definition*
- *Part 3 : File Service Definition*
- *Part 4 : File Protocol Specification*
- *Part 5 : Protocol Implementation Conformance Statement Proforma*

Information processing systems – Open Systems Interconnection – File Transfer, Access and Management –

Part 4:
File Protocol Specification

AMENDMENT 2 : Overlapped access

0. Introduction

Clause 0 provides an introduction to this amendment. The text in this clause is not intended for inclusion in ISO 8571 part 4.

0.1 General

ISO 8571 part 4 provides specifications of the protocols that support the internal and external file service interfaces.

This amendment extends these protocol specifications to provide support for 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

This amendment makes no additions to clause 1.

2. Field of application

This amendment makes no additions to clause 2.

3. References

This amendment makes no additions to clause 3.

4. Definitions and abbreviations

This amendment makes no additions to clause 4.

Section one: General

5. Overview of the protocol

5.2 Services assumed by the basic file protocol

Amend table 1.

5.6 Protocol functional units

Add to end of list.

k) consecutive access functional unit

l) concurrent access functional unit

FTAM Functional Unit	Session Functional Unit	Presentation Functional Unit
Kernel(4)	Kernel Duplex Optionally: Resynch(1) Minor synch(2)	Kernel Duplex Optionally: Resynch(1) Minor synch(2) Context Management(3)
Recovery	Minor Synch Symmetric Synchronize (5)	Minor Synch Symmetric Synchronize (5)
Restart	Minor Synch Resynchronize Symmetric Synchronize (5)	Minor Synch Resynchronize Symmetric Synchronize (5)

NOTES

5 The Symmetric Synchronize functional unit is used when overlapped access is in use. Pending the specification of presentation symmetric synchronisation services, recovery mechanisms will not be available for use during overlapped access.

Section two: Basic file protocol

6. State of the association provided

6.2 Additional state information

6.2.3 Next state indicator

Replace second sentence.

The defined values are the state names "initialised", "selected", "normal data transfer idle", "consecutive data transfer idle", "concurrent data transfer idle", and "unset".

6.2.5

Add as last paragraph:

In overlapped access, the bulk transfer number identifies the bulk data transfer that can be cancelled, restarted or recovered. Thus, the bulk transfer number is incremented only when there is an outstanding data transfer request and the previous data transfer cannot be cancelled, restarted or recovered. If an initiator and a responder have different bulk transfer numbers then it is the bulk data transfer associated with the smaller of the two numbers that is cancelled, restarted or recovered.

6.2.6 Transfer number

In overlapped access, the transfer number identifies the bulk data transfer within a sequence of transfers from one data transfer idle state to a next data transfer idle state within an open regime. It is set to zero at each data transfer idle state.

In concurrent overlapped access, two transfer numbers are maintained - one for reads and one for writes.

7. File protocol data units

This amendment makes no additions to clause 7.

8. File initiating entity actions

9. File responding entity actions

10. File general actions

This amendment makes no additions to clause 10.

Section three: The basic bulk data transfer protocol

11. State of bulk data transfer activity

Re-label sub-clause 11.2.

11.2 Additional state of the entities (without overlapped access)

Replace first paragraph.

The following sub-clauses define the items of state information associated with the basic protocol entities for the purposes of bulk data transfer without overlapped access.

Add the following sub-clauses.

11.3 Additional state of the entities (with consecutive access)

The following sub-clauses define the items of state information associated with the basic protocol entities for the purposes of bulk data transfer with consecutive access.

11.3.1 Current Transfer Number

The current transfer number indicates the transfer number of the bulk data transfer procedure that is currently considered "in progress".

11.3.2 Start Bulk Transfer Number

The start bulk transfer number indicates the bulk transfer number of the first in a sequence of overlapped data transfers. The start bulk transfer number is used to calculate the bulk transfer number and transfer number for cancel, recover and restart.

11.3.3 Checkpoint identifier expected

The checkpoint identifier expected reflects the sequence of checkpoints within bulk data, and is reset by the start of the bulk data transfer, and by error recovery mechanisms. The value is incremented when a checkpoint is made. The value is an integer in the range 1 to 999998. Initially, the value is determined by the state of the association.

The checkpoint identifier expected applies only to FTAM regimes for which the use of the restart and/or recovery functional units have been successfully negotiated.

11.3.4 First next synchronisation point number

The first next synchronisation point number reflects the sequence of events in the supporting synchronisation services for a sender. The number is the serial number of the next session synchronisation point to be issued by the session service provider. The value is an integer in the range 0 to 999998. Initially on a newly created session connection, the value is 1.

The first next synchronisation point number applies only to FTAM regimes for which the use of the presentation symmetric synchronisation functional unit has been successfully negotiated.

11.3.5 Second next synchronisation point number

The second next synchronisation point number reflects the sequence of events in the supporting synchronisation services for a receiver. The number is the serial number of the next session synchronisation point to be issued by the session service provider. The value is an integer in the range 0 to 999998. Initially on a newly created session connection, the value is 1.

The second next synchronisation point number applies only to FTAM regimes for which the use of the presentation symmetric synchronisation functional unit has been successfully negotiated.

Table 5 - Protocol Data Units

Name	Carried by	Functional units
F-CHECK request (see note 2)	P-SYNC-MINOR request	recovery, restart
F-CHECK response (see note 2)	P-SYNC-MINOR response	recovery, restart

NOTES
1 The data value corresponds to an F-DATA request service primitive. There is no F-DATA request PDU as such.

2 If overlapped access is not in use then the F-CHECK request and response primitives are mapped directly onto the P-SYNC-MINOR request and response primitives, with no additional syntax.

11.3.6 Synchronisation offset

The synchronisation offset is a constant established when a read or write bulk data transfer is initiated or recovered, which gives the difference between the checkpoint identifier expected and the next resynchronisation point number.

The synchronisation offset applies only to FTAM regimes for which use of the presentation symmetric synchronisation functional unit has been successfully negotiated.

NOTE - Except during the issue of a checkpoint, or performance of the restart procedure the value of the synchronisation offset is equal to the difference between the expected checkpoint number and the next synchronisation point number.

11.3.7 Outstanding Checkpoint counter

The outstanding checkpoint counter records the number of checkpoints which are unacknowledged.

11.3.8 Read/Write Indicator

The read/write indicator records whether the current bulk data transfer is to or from the initiator. The value is set upon the beginning of the data transfer regime. The defined values are "reading", "writing" and "unset". The initial value is unset.

11.3.9 Discard Indicator

The discard indicator is used to signal that data received during cancellation or before recovery is invalid and should be thrown away. If the recovery or restart functional units are selected and presentation resynchronisation functional unit has been successfully negotiated, it is used in conjunction with session resynchronisation to produce a recovery without user visibility of the error; otherwise it is used during the cancel phase. The defined values are "unset" and "set". Initially, the value is "unset".

11.3.10 Transfer Request Queue

The transfer request queue records the transfer number and transfer type (read or write) of all outstanding data transfers.

11.3.11 Transfer End Queue

The transfer end queue records the transfer numbers of all data transfers, already recorded on the transfer request queue, for which a transfer end PDU request or response has been issued.

11.3.12 Data End Queue

The data end queue records the transfer numbers of all data transfers, already recorded on the data request queue, for which a data end PDU has been issued or received.

11.3.13 Read and write checkpoint tables

The read and write checkpoint tables (two separate tables are kept) record checkpointing information for each ongoing bulk data transfer. The following information is kept for each currently active bulk data transfer: transfer number, bulk transfer number, checkpoint expected, synchronisation offset, and checkpoint counter. The size of the table is that of the transfer window negotiated when the file is opened.

11.3.14 Last transfer end confirm indicator

The last transfer end confirm indicator records the last transfer end received by the initiator. The last transfer end confirm received is sent to the responder on a transfer end request PDU. It is also included on cancel, recover and restart PDUs. The responder uses the information in removing items from the transfer end response queue and in the re-issuing of transfer-end response after a session resynchronisation in the direction of responder to initiator.

11.3.15 Last transfer end indication indicator

The last transfer end indication indicator records the last transfer end indication received by the responder. It is included on cancel, recover and restart PDUs. The initiator uses the information in the re-issuing of transfer end requests after a session resynchronisation in the direction of initiator to responder.

11.4 Additional state of the entities (with concurrent access)

The following sub-clauses define the items of state information associated with the basic protocol entities for the purposes of bulk data transfer with concurrent access.

11.4.1 Current Read Transfer Number and Current Write Transfer Number

The current transfer read number and the current transfer write number indicate the transfer numbers of the read and write data transfer procedures that are in progress.

11.4.2 Start Bulk Transfer Number

The start bulk transfer number indicates the bulk transfer number of the first in a sequence of overlapped data transfers. The start bulk transfer number is used to calculate the bulk transfer number and transfer number for cancel, recover, and restart.

11.4.3 Checkpoint identifier expected

The checkpoint identifier expected reflects the sequence of checkpoints within bulk data, and is reset by the start of the bulk data transfer, and by error recovery mechanisms. The value is incremented when a checkpoint is made. The value

is an integer in the range 1 to 999998. Initially, the value is determined by the state of the association.

The checkpoint identifier expected applies only to FTAM regimes for which the use of the restart and/or recovery functional units have been successfully negotiated.

11.4.4 First next synchronisation point number

The first next synchronisation point number reflects the sequence of events in the supporting synchronisation services for a sender. The number is the serial number of the next session synchronisation point to be issued by the session service provider. The value is an integer in the range 0 to 999998. Initially on a newly created session connection, the value is 1.

The first next synchronisation point number applies only to FTAM regimes for which the use of the presentation symmetric synchronisation functional unit has been successfully negotiated.

11.4.5 Second next synchronisation point number

The second next synchronisation point number reflects the sequence of events in the supporting synchronisation services for a receiver. The number is the serial number of the next session synchronisation point to be issued by the session service provider. The value is an integer in the range 0 to 999998. Initially on a newly created session connection, the value is 1.

The second next synchronisation point number applies only to FTAM regimes for which the use of the presentation symmetric synchronisation functional unit has been successfully negotiated.

11.4.6 Synchronisation offset

The synchronisation offset is a constant established when a read or write bulk data transfer is initiated or recovered, which gives the difference between the checkpoint identifier expected and the next resynchronisation point number.

The synchronisation offset applies only to FTAM regimes for which use of the presentation symmetric synchronisation functional unit has been successfully negotiated.

NOTE - Except during the issue of a checkpoint, or performance of the restart procedure the value of the synchronisation offset is equal to the difference between the expected checkpoint number and the next synchronisation point number.

11.4.7 Outstanding Checkpoint counter

The outstanding checkpoint counter records the number of checkpoints which are unacknowledged.

11.4.8 Read Indicator and Write Indicator

The read indicator and the write indicator record whether or not there are read or write data transfers in progress. The defined values are "reading"/"writing" and "unset". The initial value is unset.

11.4.9 Discard indicator

The discard indicator is used to signal that data received during cancellation or before recovery is invalid and should be thrown away. If the recovery or restart functional units are selected and presentation resynchronisation functional unit has been successfully negotiated, it is used in conjunction with session resynchronisation to produce a recovery without user visibility of the error; otherwise it is used during the cancel phase. The defined values are "unset" and "set". Initially, the value is "unset".

11.4.10 Transfer Read Request Queue and Transfer Write Request Queue

The transfer read request queue and the transfer write request queue record the transfer numbers of all outstanding read and write data transfers.

11.4.11 Transfer End Read Queue and Transfer End Write Queue

The transfer end read queue and the transfer end write queue record the transfer numbers of all read and write data transfers, already recorded on the transfer request read (write) queue, for which a transfer end PDU request or response has been issued.

11.4.12 Data End Read Queue and Data End Write Queue

The data end read queue and the data end write queue record the transfer numbers of all read and write data transfers, all ready recorded on the transfer request read (write) queue, for which a data end PDU has been issued/ received.

11.4.13 Read and write checkpoint tables

The read and write checkpoint tables (two separate tables are kept) record checkpointing information for each ongoing bulk data transfer. The following information is kept for each currently active bulk data transfer: transfer number, bulk transfer number, checkpoint expected, synchronisation offset, and checkpoint counter. The size of the table is that of the transfer window negotiated when the file is opened.

11.4.14 Last transfer end confirm indicator

The last transfer end confirm indicator records the last

transfer end read (write) received by the initiator. The last transfer end confirm received is sent to the responder on a transfer end request PDU. It is also included on cancel, recover and restart PDUs. The responder uses the information in removing items from the transfer end response queue and in the re-issuing of transfer-end response after a session resynchronisation in the direction of responder to initiator.

11.4.15 Last transfer end indication indicator

The last transfer end indication indicator records the last transfer end read (write) indication received by the responder. It is included on cancel, recover and restart PDUs. The initiator uses the information in the re-issuing of transfer end requests after a session resynchronisation in the direction of initiator to responder.

12. Bulk data transfer protocol data units

Amend table 5.

13. Bulk data transfer initiating entity actions

14. Bulk data transfer responding entity actions

15. Bulk data transfer sending entity actions

16. Bulk data transfer receiving entity actions

17. Bulk data transfer general actions

Section four: The error recovery protocol

18. Protocol mechanisms

19. Specification of the error control protocol

Section five: Abstract Syntax

20. Abstract Syntax Definition

20.3 ASN.1 Module Definition

Amend Figure 7.

Amend Figure 8.

Amend Figure 9.

Amend Figure 10.

21. Application Context Name

This amendment makes no additions to clause 21.

22. Conformance

This amendment makes no additions to clause 22.

```
ISO8571-FTAM  DEFINITIONS  ::=

BEGIN

   Functional-Units  ::= [4]  IMPLICIT  BIT  STRING
                     {  read                                ( 2),
                        write                               ( 3),
                        file-access                         ( 4),
                        limited-file-management             ( 5),
                        enhanced-file-management            ( 6),
                        grouping                            ( 7),
                        fadu-locking                        ( 8),
                        recovery                            ( 9),
                        restart-data-transfer               (10),
                        limited-filestore-management        (11),
                        enhanced-filestore-management       (12),
                        object-manipulation                 (13),
                        group-manipulation                  (14),
                        consecutive-access                  (15),
                        concurrent-access                   (16)  }

END
```


Figure 7 - FTAM regime PDUs

```
ISO8571-FTAM DEFINITIONS ::=

BEGIN

   F-OPEN-request ::= SEQUENCE {
      processing-mode           [0] IMPLICIT BIT STRING
                                    { f-read       (0),
                                      f-insert          (1),
                                      f-replace         (2),
                                      f-extend          (3),
                                      f-erase           (4) } DEFAULT { f-read },
      contents-type             [1] CHOICE {
                  unknown       [0] IMPLICIT NULL,
                  proposed      [1] Contents-Type-Attribute },
      concurrency-control             Concurrency-Control OPTIONAL,
      shared-ASE-information    Shared-ASE-Information OPTIONAL,
      enable-fadu-locking       [2] IMPLICIT BOOLEAN DEFAULT FALSE,
      activity-identifier       Activity-Identifier OPTIONAL,
            - Only used in the recovery functional unit.
      recovery-mode             [3] IMPLICIT INTEGER
                  { none (0),
                    at-start-of-file (1),
                    at-any-active-checkpoint (2) } DEFAULT none,
      remove-contexts           [4] IMPLICIT SET OF Abstract-Syntax-Name OPTIONAL,
      define-contexts           [5] IMPLICIT SET OF Abstract-Syntax-Name OPTIONAL,
            - The following are conditional on the negotiation of the consecutive overlap or
            - concurrent overlap functional units.
      degree-of-overlap         Degree-Of-Overlap OPTIONAL,
      transfer-window           [7] IMPLICIT INTEGER OPTIONAL }

   F-OPEN-response ::= SEQUENCE {
      state-result        State-Result DEFAULT success,
      action-result             Action-Result DEFAULT success,
      contents-type             [1] Contents-Type-Attribute,
      concurrency-control       Concurrency-Control OPTIONAL,
      shared-ASE-information    Shared-ASE-Information OPTIONAL,
      diagnostic                Diagnostic OPTIONAL,
      recovery-mode                   [3] IMPLICIT INTEGER
                  { none (0),
                    at-start-of-file (1),
                    at-any-active-checkpoint (2) } DEFAULT none,
      presentation-action       [6] IMPLICIT BOOLEAN DEFAULT FALSE,
            - This flag is set if the responder is going to follow this response by a P-ALTER-CONTEXT
            -  exchange.
            - The following are conditional on the negotiation of the consecutive access or
            - concurrent access functional units.
      degree-of-overlap         Degree-Of-Overlap OPTIONAL,
      transfer-window           [7] IMPLICIT INTEGER OPTIONAL }
```


Figure 8 - File selection and file open regime PDUs

```
F-RECOVER-request ::= SEQUENCE {
    activity-identifier                 Activity-Identifier,
    bulk-transfer-number                [0] IMPLICIT INTEGER,
          - If concurrent access was in use then this parameter indicates the read bulk
          -  transfer.
    requested-access                Access-Request,
    access-passwords                Access-Passwords OPTIONAL,
    recovery-point                      [2] IMPLICIT INTEGER DEFAULT 0,
          - zero indicates beginning of file point after last checkpoint indicates end of file
    remove-contexts                 [3] IMPLICIT SET OF Abstract-Syntax-Name OPTIONAL,
    define-contexts                 [4] IMPLICIT SET OF Abstract-Syntax-Name OPTIONAL,
          - The following are conditional on the negotiation of overlapped access.
    concurrent-bulk-transfer-number     [7] IMPLICIT INTEGER OPTIONAL,
          - conditional on use of concurrent access
    concurrent-recovery-point       [8] IMPLICIT INTEGER OPTIONAL,
          - conditional on use of concurrent access. Zero indicates beginning of file
          - point after last checkpoint indicates end of file
    last-transfer-end-read-response [9] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-write-response    [10] IMPLICIT INTEGER OPTIONAL }

F-RECOVER-response ::= SEQUENCE {
    state-result        State-Result Default success,
    action-result           Action-Result DEFAULT success,
    contents-type           [1] Contents-Type-Attribute,
    recovery-point          [2] IMPLICIT INTEGER DEFAULT 0,
          - Zero indicates beginning of file; point after last checkpoint indicates end of file
    diagnostic          Diagnostic OPTIONAL,
    presentation-action     [6] IMPLICIT BOOLEAN DEFAULT FALSE,
          - This flag is set if the responder is going to follow this response
          - by a P-ALTER-CONTEXT exchange.
          - The following are conditional on the negotiation of overlapped access.
    concurrent-recovery-point       [8] IMPLICIT INTEGER OPTIONAL,
          - conditional on use of concurrent access. Zero indicates beginning of file; point after
          - last checkpoint indicates end of file
    last-transfer-end-read-request  [9] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-write-request [10] IMPLICIT INTEGER OPTIONAL }

END
```


Figure 8 (continued) - File selection and file open regime PDUs

Amend Figure 9.

```
ISO8571-FTAM DEFINITIONS ::

BEGIN

   F-READ-request ::= SEQUENCE {
      file-access-data-unit-identity  FADU-Identity,
      access-context                          Access-Context,
      fadu-lock                               FADU-Lock OPTIONAL,
            - The following is conditional on the negotiation of consecutive of concurrent access.
      transfer-number                         [0] IMPLICIT INTEGER OPTIONAL }

   F-WRITE-request ::= SEQUENCE {
      file-access-data-unit-operation [0] IMPLICIT INTEGER
                        { insert      (0),
                          replace     (1),
                          extend      (2) },
      file-access-data-unit-identity  FADU-Identity,
      fadu-lock                               FADU-Lock OPTIONAL,
            - The following is conditional on the negotiation of consecutive or concurrent access.
      transfer-number                         [1] IMPLICIT INTEGER OPTIONAL }

   F-TRANSFER-END-request ::= SEQUENCE {
      shared-ASE-information                  Shared-ASE-Information OPTIONAL,
            - The following are conditional on the negotiation of consecutive or concurrent access.
      request-type                            Request-Type OPTIONAL,
      transfer-number                         [0] IMPLICIT INTEGER OPTIONAL,
      last-transfer-end-read-response [1] IMPLICIT INTEGER OPTIONAL,
      last-transfer-end-write-response        [2] IMPLICIT INTEGER OPTIONAL }

   F-TRANSFER-END-response ::= SEQUENCE {
      action-result             Action-Result DEFAULT success,
      shared-ASE-information    Shared-ASE-Information OPTIONAL,
      diagnostic                Diagnostic OPTIONAL,
            - The following are conditional on the negotiation of consecutive or concurrent access.
      request-type              Request-Type OPTIONAL,
      transfer-number           [0] IMPLICIT INTEGER OPTIONAL }

   F-CANCEL-request ::= SEQUENCE {
      action-result                           Action-Result DEFAULT success,
      shared-ASE-information                  Shared-ASE-Information OPTIONAL,
      diagnostic                              Diagnostic OPTIONAL,
            - The following are conditional on the negotiation of consecutive or concurrent access.
      request-type                      Request-Type,
      transfer-number                         [0] IMPLICIT INTEGER OPTIONAL,
      last-transfer-end-read-request  [1] IMPLICIT INTEGER OPTIONAL,
      last-transfer-end-read-response [2] IMPLICIT INTEGER OPTIONAL,
      last-transfer-end-write-request [3] IMPLICIT INTEGER OPTIONAL,
      last-transfer-end-write-response        [4] IMPLICIT INTEGER OPTIONAL }

   F-CANCEL-response ::= SEQUENCE {
      action-result                           Action-Result DEFAULT success,
      shared-ASE-information                  Shared-ASE-Information OPTIONAL,
```


Figure 9 - Bulk data transfer PDUs

```
    diagnostic                          Diagnostic OPTIONAL,
          -- The following are conditional on the negotiation of consecutive or concurrent access.
    request-type                  Request-Type OPTIONAL,
    transfer-number                     [0] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-read-request  [1] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-read-response [2] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-write-request [3] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-write-response    [4] IMPLICIT INTEGER OPTIONAL }

F-CHECK-request ::= SEQUENCE {
    checkpoint-identifier    [0] IMPLICIT INTEGER,
    transfer-number    [1] IMPLICIT INTEGER }

F-CHECK-response ::= SEQUENCE {
    checkpoint-identifier    [0] IMPLICIT INTEGER,
    transfer-number    [1] IMPLICIT INTEGER }

F-RESTART-request ::= SEQUENCE {
    checkpoint-identifier               [0] IMPLICIT INTEGER,
          -- The following are conditional on the negotiation of consecutive or concurrent access.
    request-type                  Request-Type OPTIONAL,
    transfer-number               [1] IMPLICIT INTEGER,
    last-transfer-end-read-request  [2] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-read-response [3] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-write-request [4] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-write-response    [5] IMPLICIT INTEGER OPTIONAL }

F-RESTART-response ::= SEQUENCE {
    checkpoint-identifier               [0] IMPLICIT INTEGER,
          -- The following are conditional on the negotiation of consecutive or concurrent access.
    request-type                  Request-Type OPTIONAL,
    transfer-number               [1] IMPLICIT INTEGER,
    last-transfer-end-read-request  [2] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-read-response [3] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-write-request [4] IMPLICIT INTEGER OPTIONAL,
    last-transfer-end-write-response    [5] IMPLICIT INTEGER OPTIONAL }

END
```


Figure 9 (continued) - Bulk data transfer PDUs