Frame Relay Fragmentation Implementation Agreement

FRF.12

Frame Relay Forum Technical Committee

Note: The user's attention is called to the possibility that implementation of the Frame Relay Implementation Agreement contained herein may require the use of inventions covered by patent rights held by third parties. By publication of this Frame Relay Implementation Agreement the Frame Relay Forum makes no representation that the implementation of the specification will not infringe on any third party rights. The Frame Relay Forum take no position with respect to any claim that has been or may be asserted by any third party, the validity of any patent rights related to any such claims, or the extent to which a license to use any such rights may not be available.

Editor:

Andrew G. Malis

For more information contact:

The Frame Relay Forum
Suite 307
39355 California Street
Fremont, CA 94538 USA

Phone: +1 (510) 608-5920
FAX: +1 (510) 608-5917
E-Mail: frf@frforum.com

Table of Contents

Table of Figures

Revision History

Version	Change	Date
1.0	Approved	December 1997

(This page intentionally left blank)

Frame Relay Fragmentation Implementation Agreement

1. INTRODUCTION

1.1 Purpose

This document is a Frame Relay Fragmentation Implementation Agreement. It provides transmitting Frame Relay DTEs and DCEs with the ability to fragment long frames into a sequence of shorter frames, which will then be reassembled into the original frame by the receiving peer DTE or DCE. Frame fragmentation is necessary to control delay and delay variation when real-time traffic such as voice is carried across the same interfaces as data.

This agreement supports three applications of fragmentation:

1. Locally across a Frame Relay UNI interface between the DTE-DCE peers.
2. Locally across a Frame Relay NNI interface between the DCE peers.
3. End-to-End between two Frame Relay DTEs interconnected by one or more Frame Relay networks. When used end-to-end, the fragmentation procedure is transparent to Frame Relay network(s) between the transmitting and receiving DTEs.

The agreements herein were reached in the Frame Relay Forum, among vendors and suppliers of Frame Relay network products and services, and are based on the relevant Implementation Agreements and standards referenced in Section 2.

This document may be submitted to bodies involved in ratification of implementation agreements and conformance testing to facilitate multi-vendor interoperability, and to standards bodies for inclusion in international standards.

1.2 Terminology

Must, Shall, or Mandatory - the item is an absolute requirement of this IA.

Should - the item is highly desirable.

May or Optional - the item is not compulsory, and may be followed or ignored according to the needs of the implementer.

Not Applicable - the item is outside the scope of this IA.

1.3 Acronym List

AAL5 - ATM Adaptation Layer 5
ATM - Asynchronous Transfer Mode
DCE - Data Communications Equipment
DLCI - Data Link Connection Identifier

DTE - Data Terminal Equipment
FCS - Frame Check Sequence
FR - Frame Relay
IA - Implementation Agreement
IWF - Interworking Function
MTU – Maximum Transmission Unit
NLPID - Network Layer Protocol Identifier
NNI - Network-to-Network Interface
PDU - Protocol Data Unit
PPP - Point-to-Point Protocol
PVC - Permanent Virtual Connection
SVC - Switched Virtual Connection
UI - Unnumbered Information
UNI - User-to-Network Interface
VC - Virtual Connection
VoFR - Voice over Frame Relay

2. RELEVANT STANDARDS

The following is a list of standards and IAs on which this Frame Relay Fragmentation IA is based:

[1] FRF.1.1, D. Sinicrope (ed.), User-to-Network Implementation Agreement (UNI), January 19, 1996.

[2] FRF.2.1, L. Greenstein (ed.), Frame Relay Network-to-Network Interface Implementation Agreement, July 10, 1995.

[3] FRF.3.1, R. Cherukuri (ed.), Multiprotocol Encapsulation Implementation Agreement, June 22, 1995.

[4] FRF.5, D. O'Leary (ed.), Frame Relay/ATM PVC Network Interworking Implementation Agreement, December 20, 1994.

[5] FRF.8, D. O'Leary (ed.), Frame Relay/ATM PVC Service Interworking Implementation Agreement, April 14, 1995.

[6] FRF.9, D. Cantwell (ed.), Data Compression Over Frame Relay Implementation Agreement, January 22, 1996.

[7] FRF.11, K. Rehbehn, R. Kocen, T. Hatala (eds.), Voice Over Frame Relay Implementation Agreement, March 1997.

[8] ITU Recommendation Q.922, ISDN Data Link Layer Specification For Frame Mode Bearer Services, 1992.

[9] RFC 1490, T. Bradley, C. Brown, A. Malis, Multiprotocol Interconnect over Frame Relay, July 26, 1993.

[10] RFC 1990, K. Sklower, B. Lloyd, G. McGregor, D. Carr, T. Coradetti, The PPP Multilink Protocol (MP), August 1996.

3. OVERVIEW

To properly support voice and other real-time (delay-sensitive) data on lower-speed UNI [1] or NNI [2] links, it is necessary to fragment long frames that share the same UNI or NNI link with shorter frames so that the shorter frames are not excessively delayed. Fragmentation enables interleaving delay-sensitive traffic on one VC with fragments of a long frame on another VC utilizing the same interface.

This Implementation Agreement:

- Allows real-time and non-real-time data to share the same Frame Relay UNI or NNI link.
- Allows the fragmentation of frames of all formats.
- Defines a fragmentation procedure that can be used by other protocols or IAs, such as FRF.11 [7]. See Section 9.2 for further details.
- Defines three fragmentation models, locally across a UNI, locally across a NNI, and End-to-End. All three models share common fragmentation procedures.

The fragmentation procedure previously specified in FRF.3.1 [3] and RFC 1490 [9] is no longer recommended, and is replaced by this IA.

Fragmentation may be used with all Q.922 [8] address formats. For clarity, the figures that include address octets will only illustrate the two-octet address.

4. FRAGMENTATION MODELS

4.1 UNI Fragmentation

UNI (DTE-DCE) fragmentation is used in order to allow real-time and data frames to share the same UNI interface between a DTE and the Frame Relay Network. Fragmentation is strictly local to the interface, and the fragment size can be optimally configured to provide the proper delay and delay variation based upon the *logical* speed of the DTE interface (the logical speed of an interface may be slower than the physical clocking rate if a channelized physical interface is used).

Since fragmentation is local to the interface, the network can take advantage of the higher internal trunk speeds by transporting the complete frames, which is more efficient than transporting a larger number of smaller fragments.

UNI fragmentation is also useful when there is a speed mismatch between the two DTEs at the ends of a VC. It also allows the network to proxy for a DTE that does not implement End-to-End fragmentation.

For UNI fragmentation, the DTE and DCE interfaces act as fragmentation and reassembly peers, as shown in Figure 4-1:

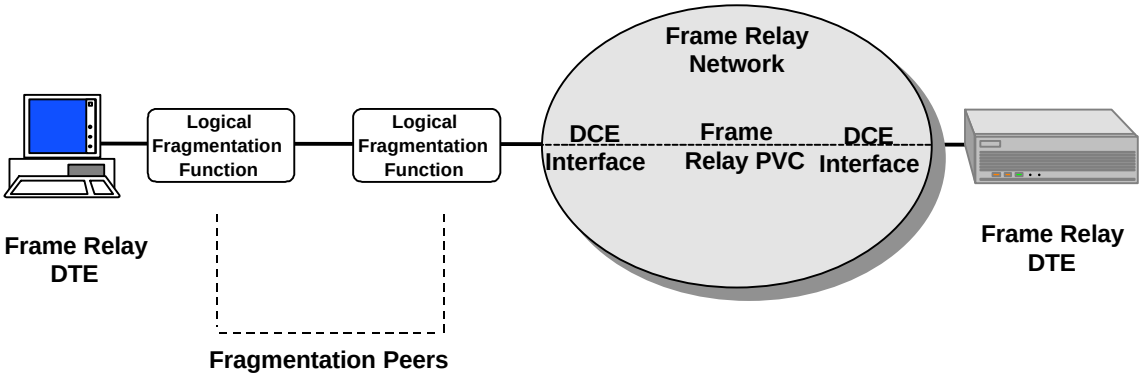

Figure 4-1 UNI Fragmentation/Reassembly Reference Diagram

Note that the functionality specified in this Implementation Agreement has been illustrated as a standalone "Logical Fragmentation Function", but it is expected to be implemented in the DTE and DCE interfaces shown in the diagram.

UNI fragmentation is provisioned on an interface-by-interface basis. When UNI fragmentation is used on an interface, then all frames on all DLCIs (including DLCI 0, PVCs, and SVCs) are preceded by the fragmentation header (described in Section 5.1).

Note that UNI and NNI fragmentation formats and procedures are identical.

4.2 NNI Fragmentation

NNI interfaces may also act as fragmentation and reassembly peers, as shown in Figure 4-2.

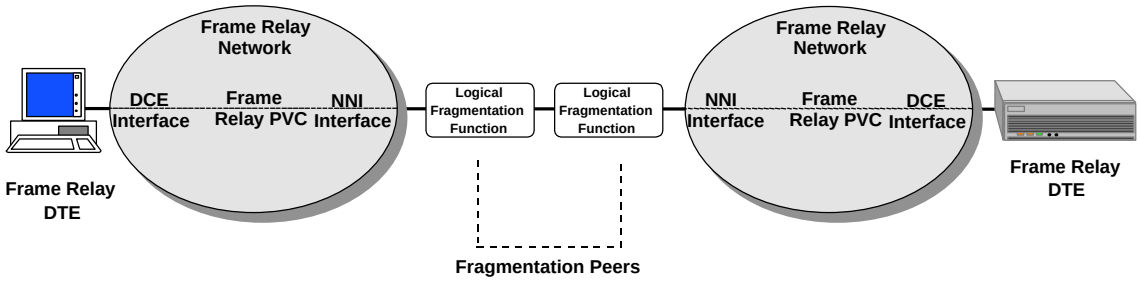

Figure 4-2 NNI-NNI Fragmentation/Reassembly Reference Diagram

Fragmentation on slower NNI links allows delay sensitive traffic on one NNI VC to be interleaved with fragments of a long data frame on another VC using the same NNI. As in Figure 4-1, the fragmentation and reassembly functionality has been illustrated as a standalone "Logical Fragmentation Function", but it is expected to be implemented in the NNI interfaces shown in the diagram.

NNI fragmentation is also provisioned on an interface-by-interface basis. When fragmentation is in use on an NNI interface, then all frames on all DLCIs (including DLCI 0, PVCs, and SVCs) are preceded by the fragmentation header (described in Section 5.1).

Note that UNI and NNI fragmentation formats and procedures are identical.

4.3 End-to-End Fragmentation

End-to-End fragmentation is used between peer DTEs, and is restricted to use on PVCs only. It is most useful when peer DTEs wish to exchange both real-time and non-real-time traffic using slower interface(s), but either one or both UNI interfaces does not support UNI fragmentation. Alternatively, they may be on separate FR networks and the path interconnecting the networks may be slow enough to require fragmentation to properly support the real-time traffic, but fragmentation is not supported over the NNI.

When used between DTEs, as shown in Figure 4-3, the fragmentation procedure is transparent to Frame Relay network(s) between the transmitting and receiving DTEs. The transmitting Frame Relay DTEs fragment long frames into a sequence of shorter frames, which will then be reassembled into the original frame by the receiving DTE. Again, the functionality specified in this Implementation Agreement has been illustrated as a standalone "Logical Fragmentation Function", but in this case it is expected to be implemented in the DTEs shown in the diagram.

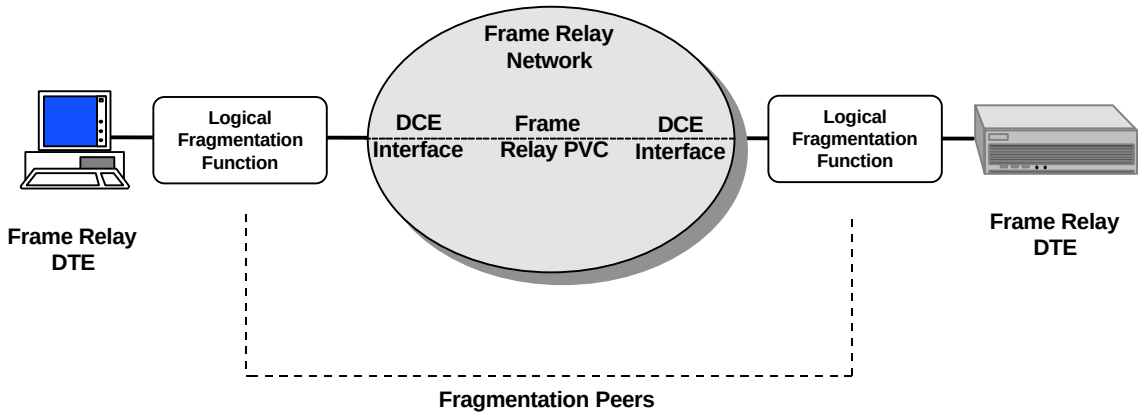

Figure 4-3 End-to-End Fragmentation/Reassembly Reference Diagram

Unlike UNI and NNI fragmentation, which fragments all frames on an interface, End-to-End fragmentation is limited to fragmenting frames on selected PVCs. When End-to-End fragmentation is provisioned on a particular PVC, frames that exceed the configured maximum fragment size must conform to the fragmentation format. Because DLCI 0 is never carried end-to-end, it is never fragmented using End-to-End fragmentation.

5. DATA FRAGMENTATION FORMATS

There are two fragmentation formats, one for Interface (UNI and NNI) fragmentation and the other for End-to-End fragmentation.

5.1 Interface Fragmentation Format

For Interface (UNI and NNI) fragmentation, a two-octet fragmentation header precedes the Frame Relay header.

The format for each fragment of a data frame is:

	8	7	6	5	4	3	2	1
Fragmentation header	B	E	C	Seq. # high 4 bits				1
	Sequence # low 8 bits							
Frame Relay header	DLCI high six bits						C/R	0
	DLCI low 4 bits				F	B	DE	1
	Fragment Payload							
	FCS (two octets)							

Figure 5-1 UNI and NNI Data Fragment Format

The (B)eginning fragment bit is a one-bit field set to 1 on the first data fragment derived from the original frame and set to 0 for all other fragments from the same frame.

The (E)nding fragment bit is a one-bit field set to 1 on the last data fragment and set to 0 for all other data fragments. A data fragment may have both the (B)eginning and (E)nding fragment bits set to 1.

The (C)ontrol bit is set to 0 in all fragments. It is reserved for future control functions.

The sequence number is a 12-bit binary number that is incremented modulo 2^{12} for every data fragment transmitted on a VC. There is a separate sequence number maintained for each DLCI across the interface.

Note that the low order bit of the first octet of the fragmentation header is set to 1. This allows the fragmentation header to be distinguishable from the Frame Relay header. This allows a fragmentation entity (UNI or NNI) to detect the misconfiguration of its peer, since the peers must be configured identically to use or not use fragmentation across an interface. If a peer is configured for UNI or NNI fragmentation and receives frames that do not contain the fragmentation header, those frames are discarded. If a peer is not configured for UNI or NNI fragmentation and receives frames with the fragmentation header, those frames will be discarded due to the violation of the Q.922 header format.

The fragment payload and fragmentation procedures are described in Section 6.1.

5.2 End-to-End Fragmentation Format

For End-to-End fragmentation, a two-octet fragmentation header follows the FRF.3.1 [3] multiprotocol encapsulation header. The Network Layer Protocol ID (NLPID) 0xB1 has been assigned to identify this fragmentation header format. See FRF.3.1 for further details concerning NLPIDs and multiprotocol encapsulation over FR.

The format for each fragment of a data frame is:

	8	7	6	5	4	3	2	1
Frame Relay header	DLCI high six bits						C/R	0
	DLCI low 4 bits				F	B	DE	1
UI (0x03)	0	0	0	0	0	0	1	1
NLPID (0xB1)	1	0	1	1	0	0	0	1
Fragmentation header	B	E	C	Seq. # high 4 bits				0
	Sequence # low 8 bits							
	Fragment Payload							
	FCS (two octets)							

Figure 5-2 End-to-End Data Fragment Format

The (B)eginning fragment bit is a one-bit field set to 1 on the first data fragment derived from the original frame and set to 0 for all other fragments from the same frame.

The (E)nding fragment bit is a one-bit field set to 1 on the last data fragment and set to 0 for all other data fragments. A data fragment may have both the (B)eginning and (E)nding fragment bits set to 1.

The (C)ontrol bit is set to 0 in all fragments. It is reserved for future control functions.

The sequence number is a 12-bit binary number that is incremented modulo 2^{12} for every data fragment transmitted on a PVC. There is a separate sequence number maintained for each fragmented PVC between DTE peers.

The fragment payload and fragmentation procedures are described in Section 6.1.

6. PROCEDURES

6.1 Fragmentation Procedure

This fragmentation procedure is based upon RFC 1990 [10]. Note that the procedure is identical for both header formats.

A series of data fragments is created by taking a frame, removing the leading flag and Q.922 address octets, removing the original FCS and trailing flag following the data, and sending the remaining octets, in their original order, as a series of data fragments. If an FRF.3.1-encapsulated frame is being fragmented, then the Q.922 control, optional pad, and NLPID octets of the original multiprotocol frame are only contained in the first data fragment. The resulting fragments must be transmitted in the same sequence as they occurred in the frame prior to being fragmented. Fragments from multiple VCs may be interleaved with each other on one interface (this is the principal objective of fragmentation).

The first data fragment in the series has the B bit set, and the final data fragment has the E bit set. Every fragment in the series contains the same address octets that were on the original unfragmented frame, including the Frame Relay congestion bits (FECN, BECN, DE).

The first fragment sent on a VC (following a VC becoming active) may have the sequence number set to any value (including zero), and the sequence number must subsequently be incremented by one for each fragment sent. The sequence number is incremented without regard to the original frame boundaries; if the last fragment in one frame used sequence number "N", then the first fragment of the following frame will use sequence number "N+1".[1] This allows lost fragments (and bursts of lost fragments) to be easily detected. Each VC has its own fragmentation sequence number sequence, independent of all other VCs.

If sufficient fragments are sent on an active VC, the sequence number will wrap from all ones to zero, and will eventually also wrap past the original sequence number sent on the VC after it became active. This wrapping may or may not occur on an original frame boundary (it is transparent to frame boundaries).

NNI interfaces are allowed to further fragment End-to-End fragments. The End-to-End fragments are encapsulated within the NNI interface fragments. Thus, the NNI fragments containing the End-to-End fragments can easily be reassembled by the receiving NNI interface, and the End-to-End fragments sent to the destination DTE for final reassembly.

Similarly, UNI interfaces may also further fragment End-to-End fragments, as described in the previous paragraph.

[1] For FRF.12 UNI, NNI, and End-to-End fragmentation, the sequence number is incremented modulo 2^{12}; for FRF.11 sub-frame data payload, the sequence number is incremented modulo 2^{13}.

See Section 8 for fragmentation examples.

6.2 Reassembly Procedure

The reassembly procedures are identical for both header formats.

For each VC, the receiver must keep track of the incoming sequence numbers and maintain the most recently received sequence number. The receiver detects the end of a reassembled frame when it receives a fragment bearing the (E)nding bit. Reassembly of the frame is complete if all sequence numbers up to that fragment have been received.

Note that the Frame Relay congestion bits (FECN, BECN, DE) must be logically ORed for all fragments, and the results included in the reassembled frame.

The receiver detects lost fragments when one or more sequence numbers are skipped. When a lost fragment or fragments are detected on a VC, the receiver must discard all currently unassembled and subsequently received fragments for that VC until it receives the first fragment that bears the (B)eginning bit. The fragment bearing the (B)eginning bit is used to begin accumulating a new frame.

In the event of an error (e.g., one or more fragments lost due to transmission error or reassembly buffer overflow), fragments which cannot be reconstructed back into the original frame must be discarded by the receiver.

7. FRAGMENT AND FRAME SIZES

This IA does not recommend any specific fragment size. The fragment size is configured in the transmitter, and two peer transmitters need not use the same fragment size.

For End-to-End fragmentation, the maximum fragment size used on a particular PVC depends on both the access line speeds at the two ends of the connection, the speeds of any NNI interfaces on the path, and the delay and delay variation requirements of the particular applications in use on the PVC. The maximum fragment size to send should be configured on a per-PVC basis. If one or more of the Frame Relay network(s) interconnecting the DTEs are implemented using FRF.5 [4] FR/ATM Network Interworking, then it may be advantageous to make the End-to-End fragment size (not including the FCS) an even multiple of the underlying ATM cell payload size in order to optimize the performance at the ATM layer. See FRF.5 [4] for additional details on how the end-to-end fragments are carried in ATM cells. Also see Appendix A for an additional discussion of End-to-End fragment size considerations in this case.

For UNI and NNI fragmentation, the maximum fragment size should be configured on a per-interface basis. The optimal fragment size for these interfaces is the result of a tradeoff between the efficiencies of large frames, the MTU size of the constituent networks, and the required delay and delay variation characteristics of the applications.

Receivers must be able to reassemble complete frames up to at least 1600 octets in length.

8. FRAGMENTATION AND OTHER FRF IMPLEMENTATION AGREEMENTS

8.1 FRF.11 VoFR Encapsulation of Data Fragments

Annex C of FRF.11 [7] defines a format allowing encapsulation of fragmented data frames in a Voice over Frame Relay (VoFR) sub-frame data payload. PVCs that utilize the VoFR sub-frame data payload for non-voice frames must use the Data Transfer Syntax Payload Format defined in Annex C of FRF.11, instead of formats indicated in this IA. Such PVCs shall use procedures described in Section 6 of this IA, except as indicated within this section.

Fragmentation using the FRF.11 Annex C format provides End-to-End fragmentation, enabling interleaving of delay-sensitive traffic on one sub-channel with fragments of a long frame on another sub-channel within the PVC.

VoFR data fragments are first created using the procedure in Section 6.1, except as follows:

- Each fragment is preceded by an FRF.11 header that identifies it as a primary payload, using sub-channel(s) provisioned to carry data.
- The data fragments are interleaved with voice sub-frames on difference sub-channels of the same PVC.
- The order of the data fragments must be maintained within each sub-channel.
- Because the sequence numbers of FRF.11 Annex C frames use a 13-bit sequence number, the sequence numbers procedures will increment modulo 2^{13}.
- UNI or NNI interfaces are allowed to further fragment frames that have been fragmented using the Data Transfer Syntax Payload Format of FRF.11 Annex C. See Section 6.1 of this IA.

8.2 Fragmentation Interaction with FRF.3.1 [3] Multiprotocol Encapsulation and non-Multiprotocol Encapsulated Data

If a VC is configured to carry data other than FRF.3.1, the specification of the relevant protocol applies to the reassembled frame.

If a VC is configured to carry FRF.3.1 multiprotocol encapsulated data, the procedures of FRF.3.1 apply to the reassembled frame. In addition, the procedures of FRF.9 may apply if data compression is enabled for the VC. See Section 8.3.

8.3 Fragmentation Interaction with FRF.9 [6] Compression

Because FRF.9 compressed frames are multiprotocol-encapsulated according to FRF.3.1, compressed frames may also be fragmented. When compression is being used on a PVC, compression is first applied to the frame, then the compressed frame is fragmented using the procedure in Section 6.1. Like any other data frame, Interface and/or End-to-End fragmentation may be used.

On reception, the fragments are first reassembled, and the resulting compressed frame is then uncompressed as per FRF.9.

8.4 End-to-End Fragmentation Interaction with FRF.8 [5] FR/ATM Service Interworking

End-to-End fragmentation must be terminated in the FR side of the FRF.8 Service Interworking Function (IWF). In the FR to ATM direction, fragments will be reassembled by the IWF to form complete frames, which are then passed to the ATM side of the IWF according to the FRF.8 procedures. In the ATM to FR direction, AAL5 PDUs will be passed to the FR side according to the FRF.8 procedures, and then fragmented according to this IA.

Note that the Frame Relay congestion bits (FECN, BECN, DE) must be logically ORed for all fragments and the result passed with the reassembled frame to the ATM side of the interworking unit. In the ATM to Frame Relay direction the ATM congestion bits (EFCN and CLP) are mapped to each Frame Relay frame fragment.

9. FRAGMENTATION EXAMPLES

9.1 Interface Fragmentation Example

An example of the Interface (UNI and NNI) fragmentation procedure, using an FRF.3.1-encapsulated frame as the data to be fragmented, is diagrammed in Figure 9-1. The octets in white indicate the data portion of the original frame that is split into fragments (three fragments in this example). While this example uses an FRF.3.1 frame for illustration purposes, any arbitrary frame contents may be fragmented. For this example, the starting sequence number of 42 was chosen at random. Note that when fragmenting FRF.3.1 data, the control octet, the optional pad (if present), and the NLPID of the original frame are transported in the first frame fragment and are part of the reassembled frame.

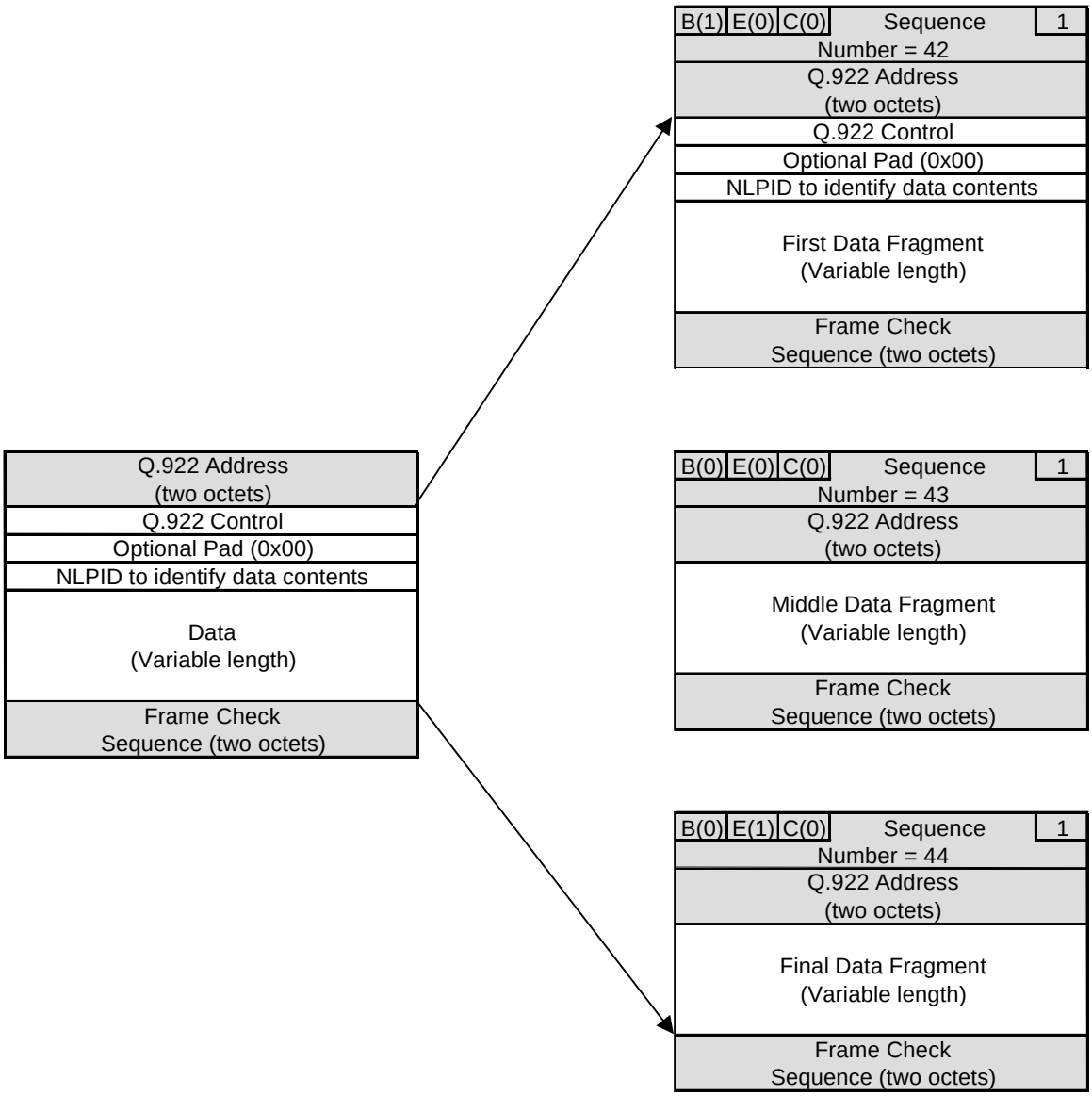

Figure 9-1 Interface (UNI and NNI) Fragmentation Example

9.2 End-to-End Fragmentation Example

An example of the End-to-End fragmentation procedure, using an FRF.3.1-encapsulated frame as the data to be fragmented, is diagrammed in Figure 9-2. The octets in white indicate the data portion of the original frame that is split into fragments (three fragments in this example). While this example uses an FRF.3.1 frame for illustration purposes, any arbitrary frame contents may be fragmented. For this example, the starting sequence number of 42 was chosen at random. Note that when fragmenting FRF.3.1 data, the control octet, the optional pad (if present), and the NLPID of the original frame are transported in the first frame fragment and are part of the reassembled frame.

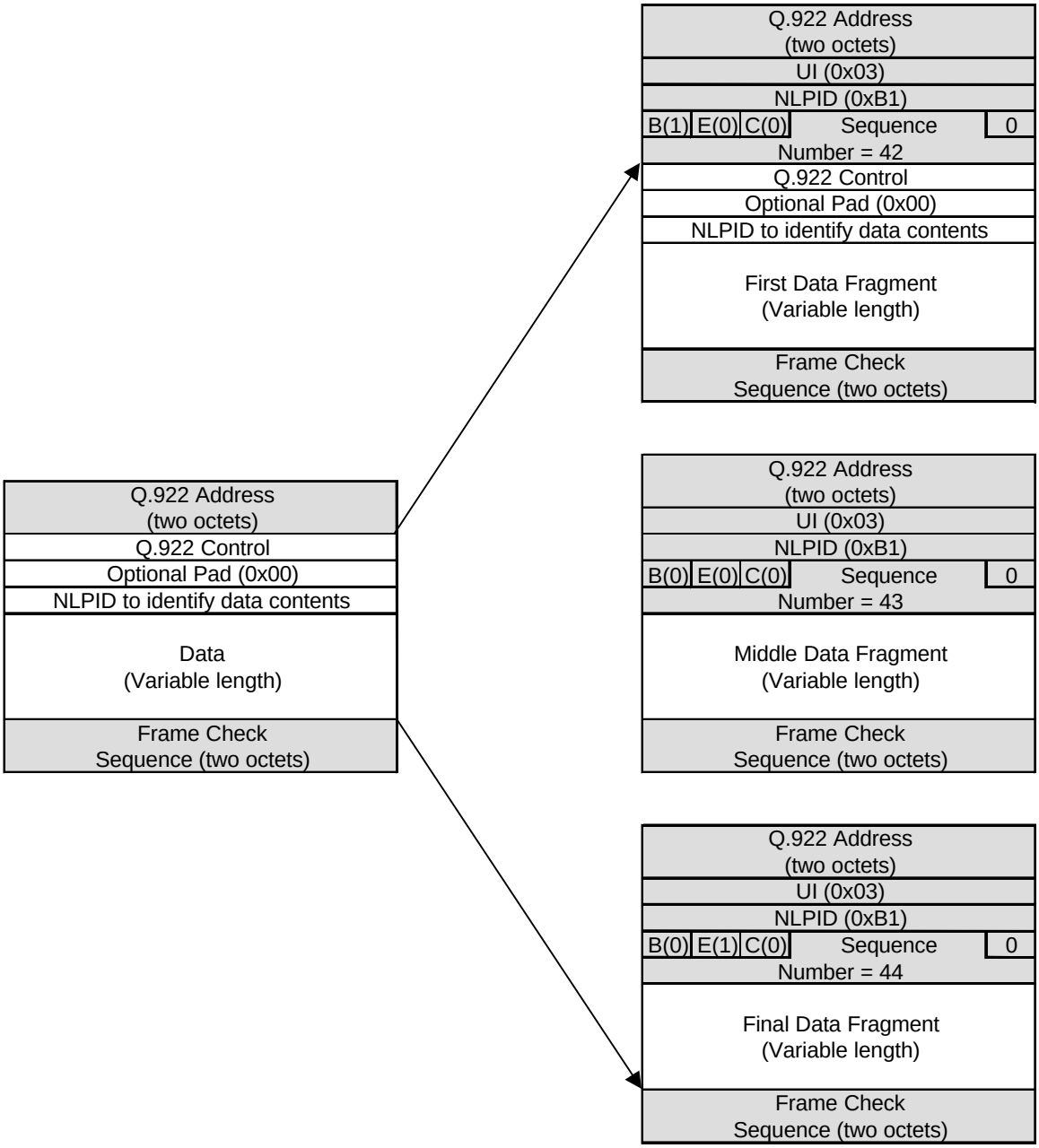

Figure 9-2 End-to-End Fragmentation Example

9.3 FRF.11 Fragmentation Example

An example of the FRF.11 fragmentation procedure, using an FRF.3.1-encapsulated frame as the data to be fragmented, is diagrammed in Figure 9-3. The octets in white indicate the data portion of the original frame that is split into fragments (three fragments in this example). While this example uses an FRF.3.1 frame for illustration purposes, any arbitrary frame contents may be fragmented. For this example, the starting sequence number of 42 was chosen at random. Note that when fragmenting FRF.3.1 data, the control octet, the optional pad (if present), and the NLPID of the original frame are transported in the first frame fragment and are part of the reassembled frame.

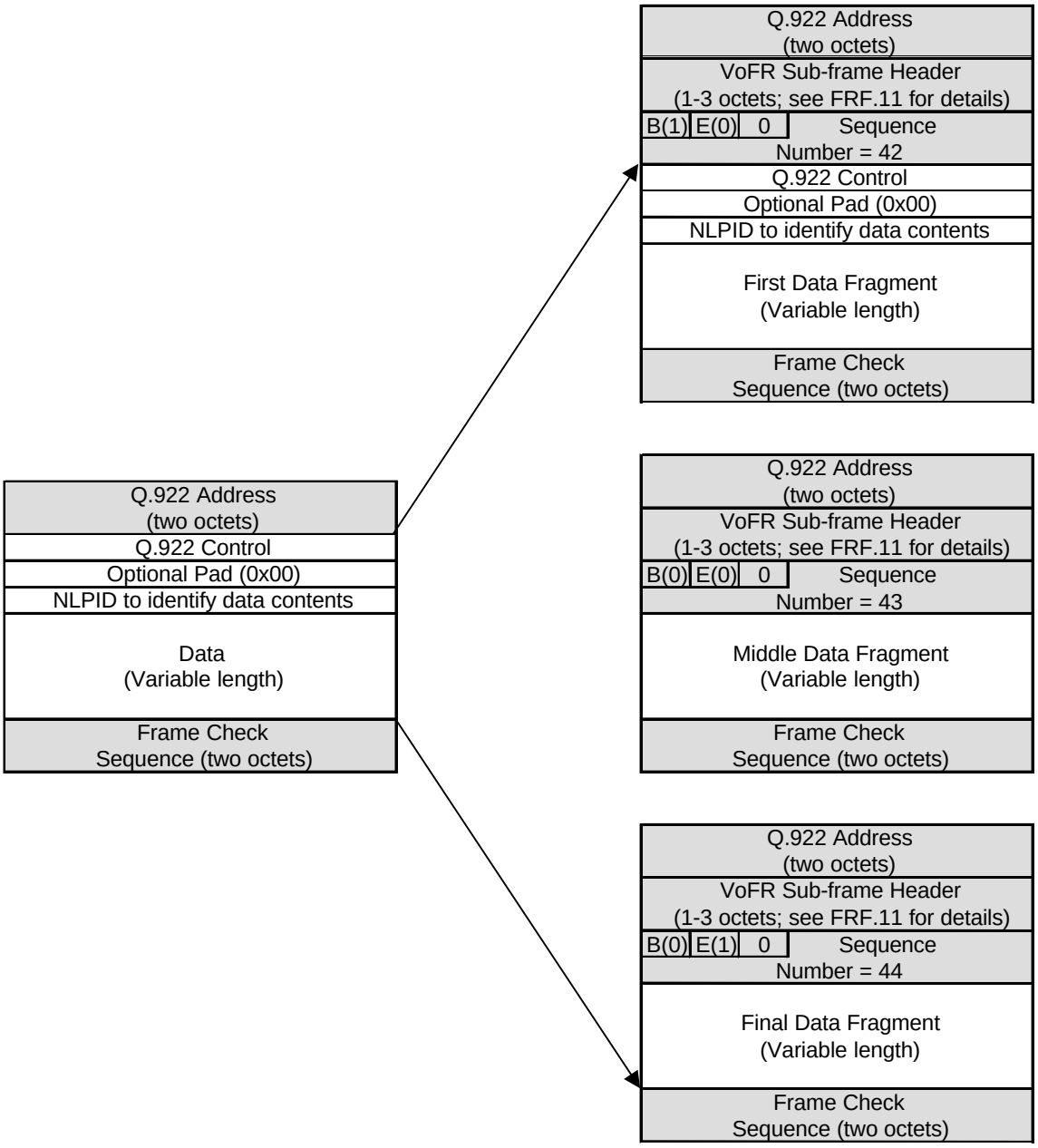

Figure 9-3 FRF.11 Fragmentation Example

Appendix A CONSIDERATIONS FOR CHOOSING THE END-TO-END FRAGMENTATION SIZE

(Informative)

Section 7 mentions that for the case of End-to-End fragmentation, one or more of the network(s) that interconnect the DTEs may use FRF.5 [4] FR/ATM Network Interworking and use ATM cell-based transport. In such cases, the choice of fragment size will have an effect on how efficiently the fragments may be packed into the ATM cells. Presented below is a figure that illustrates the efficiency of different choices for the fragment payload size for the case of networks implementing FRF.5. The efficiency is calculated as follows:

Length of PDU to convert to cells = [FR Header (2 octets)
+ Fragmentation Header (4 octets)
+ Fragmentation Payload Size (**P** octets)
+ AAL5 trailer (8 octets)]

Number of cells needed to carry PDU (**N** cells) = CEILING [Length of PDU /48 octets/cell]

Efficiency (%) = 100 * (**P** octets) / [**N** cells * 53 octets/cell]

Figure A-1 shows Efficiency (%) on the vertical axis versus Fragmentation Payload Size (**P** octets). This does not represent a requirement or a recommendation of FRF.12.

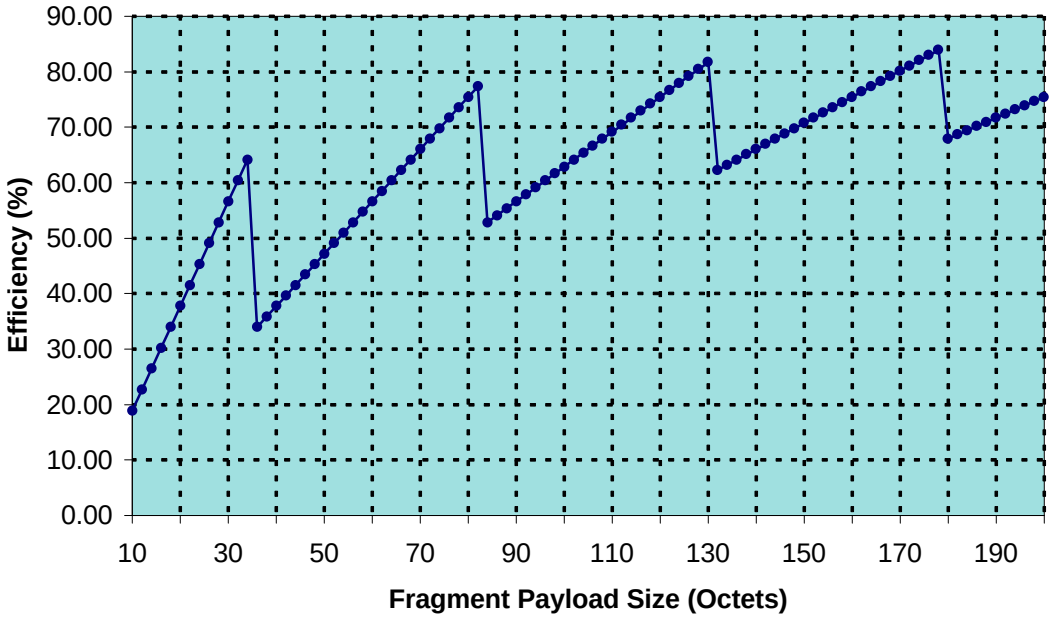

Figure A-1 Efficiency Of Different Fragment Payload Sizes When Used With FRF.5