SCADA Protocols Introduction

Kamjoo Bayat

Technical manager

www.pbscontrol.com

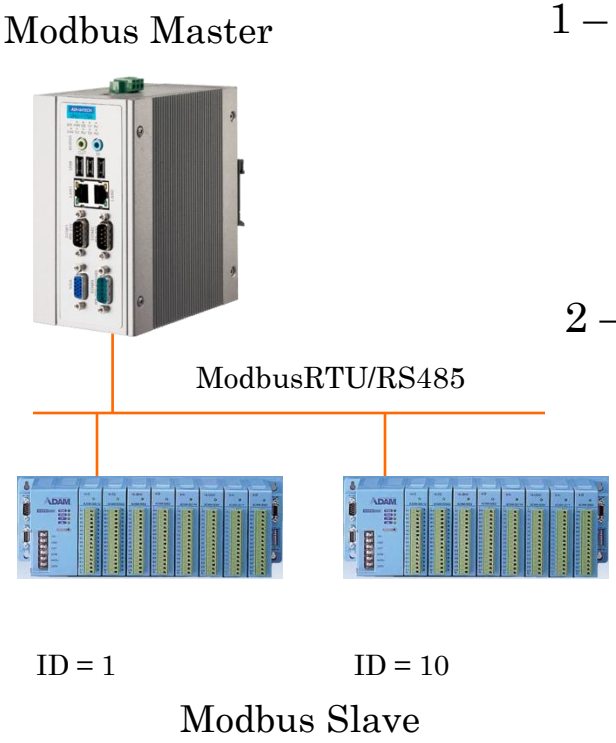
Agenda

- What is Modbus Protocol?
- What is IEC870-5-101/104 Protocol?
- What is DNP3 Protocol?
- What is OPC UA protocol ?

What is Modbus Protocol ?

- Modbus is one of the most simple protocols in industrial Automation
- Modbus is developed for Modicon PLCs in 1979 (35 Years before)
- Modbus RTU : Modbus Binary frame over RS232/RS485/RS422 (Serial)
- Modbus TCP : Modbus Binary Frame over TCP/IP
- Modbus Architecture : Master/Slave
- All Transactions always starts from Master

What is ModbusRTU protocol ?

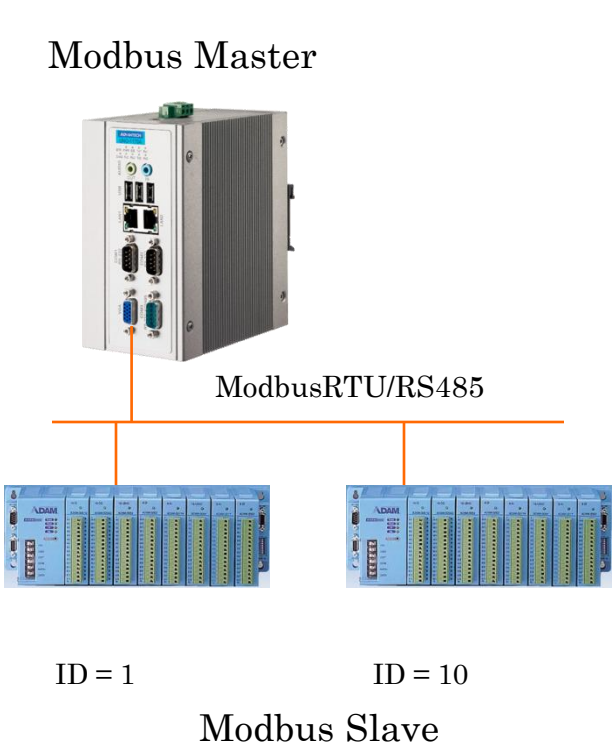


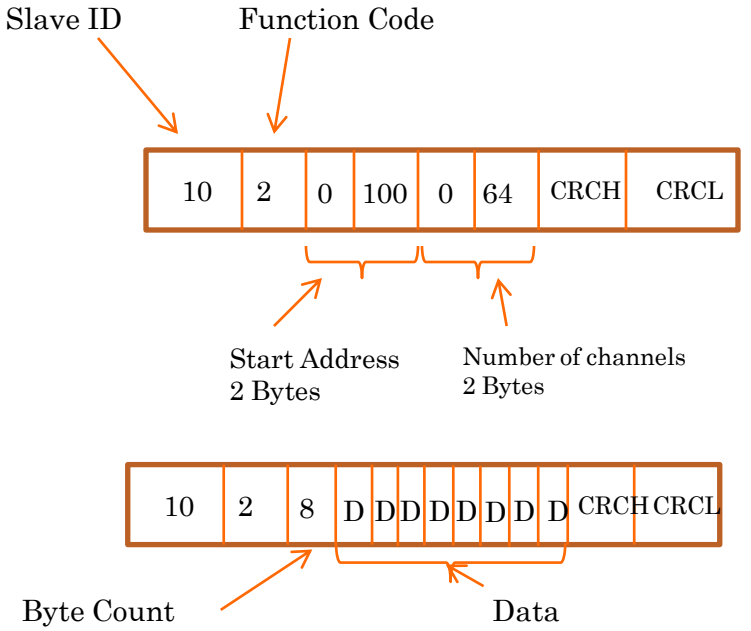
1 – Any Modbus Slave device has 4 Tables Inside
- Digital Input -Input Status 1 X reference
- Digital Output – Coil – 0 X reference
- Analog Input – Input Register – 3 X reference
- Analog Output – Holding register - 4 X reference

2 – Modbus Function codes
- 1 = Read Coil Status
- 2 = Read Input Status
- 3 = Read Holding register
- 4 = Read Input register
- 5 = Force Single Coil
- 6 = Preset Single holding register
- 15 = Force Multiple Coils
- 16 = Preset Multiple Holding register

3 – Any Modbus Slave must has unique ID in the network . Slave ID is 1 Byte in the Frame So maximum we can have 255 Slave Device on a network . First Byte in the frame.
4 – When Master Send a frame to network ,all Slaves will get Frame . But Slave that has Same ID number of frame will answer . Other will forget frame .

What is ModbusRTU protocol ?





Tag Address
2 Bytes in Frame

	1X	0X	3X	4X
	DI	DO	AI	AO
0	0			
	0			
	1			
100				
	1			
65535	0			

Modbus Limitations

- No Time label for signals
- No Time synchronization function
- Only simple data types are supported – No float – no long , …
- For float – long ,… data type you need to use Modbus registers and simulate as float , Long , …
- No powerful error detection mechanism
- No event buffering mechanism
- only one type of data can be transferred in a transaction – different commands for digitals and analogs
- Data Frame length is max 255 bytes : 127 AI or 63 Float
- Mostly use Modbus for local IO and Local HMI

What is IEC-870-5 ?

Selected application functions of IEC 60870-5-5	**User process**
Selected application information elements of IEC 60870-5-4	**Application (layer 7)**
Selected application service data units of IEC 60870-5-3	
Selected link transmission procedures of IEC 60870-5-2	**Link (layer 2)**
Selected transmission frame formats of IEC 60870-5-1	
Selected ITU-T recommendations	**Physical (layer 1)**

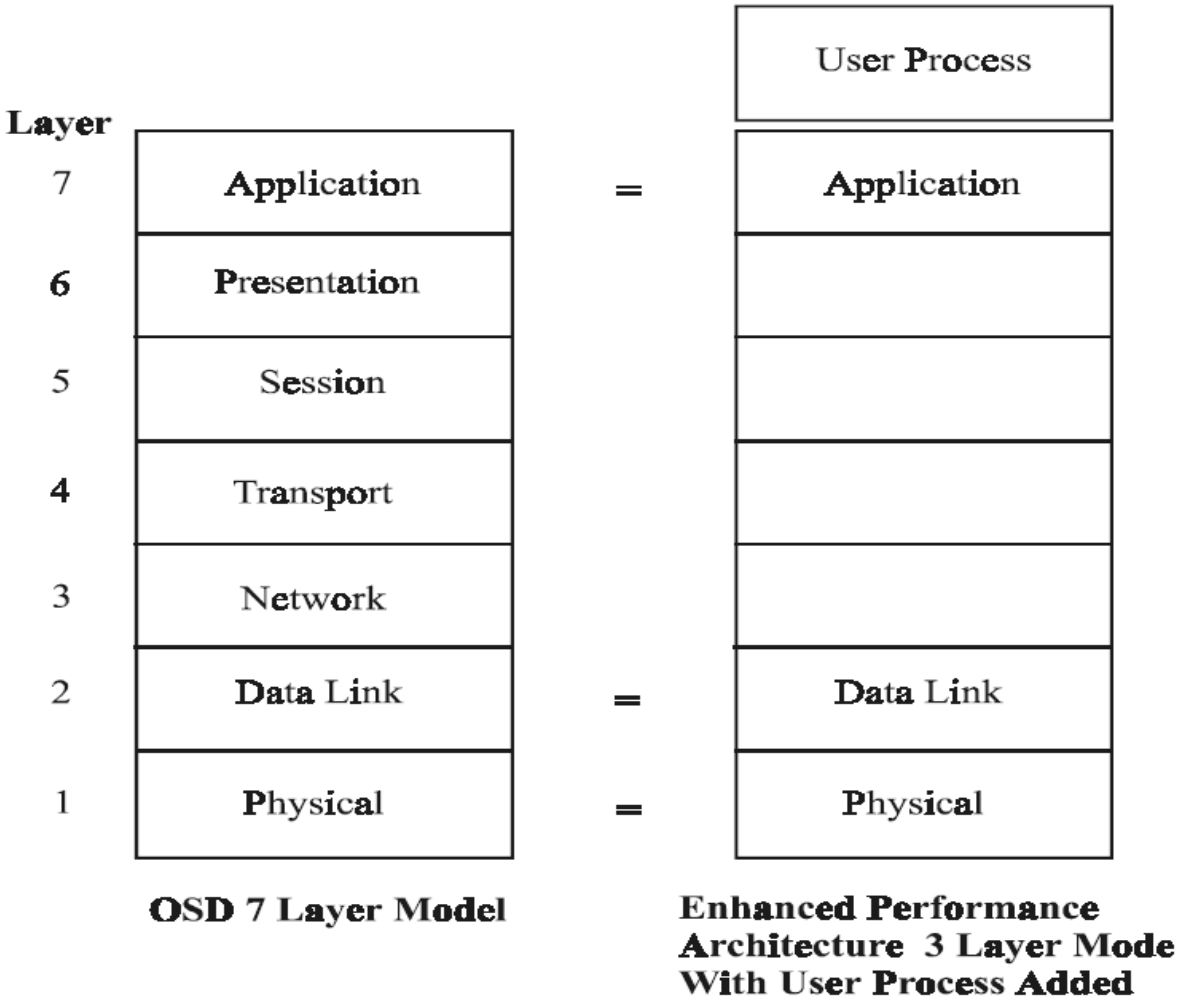
IEC 084/03

Figure 1 – Selected standard provisions of the defined telecontrol companion standard

What is IEC870-5 Protocol?

Reference	Description	Year
IEC 60870-5-101	Companion Standard for Basic Telecontrol Tasks	1995
IEC 60870-5-102	Companion Standard for Transmission of Integrated Totals	1996
IEC 60870-5-103	Companion Standard for Protection Communication	1997
IEC 60870-5-104	Network Access using Standard Transport Profiles	2000

What is EPA Structure?



IEC870-5-101 PHYSICAL LAYER

- IEC 870-5-101 specifies frame format **FT 1.2 .**
- IEC 870-5-101 is an asynchronous protocol with hamming distance = 4
- **Character format**
  - 1 Start bit
  - 1 Stop bit
  - 1 Parity bit (even)
  - 8 Data bits

The Hamming distance between:

- "karolin" and "kathrin" is 3.
- "karolin" and "kerstin" is 3.
- **1011101** and **1001001** is 2.
- **2173896** and **2233796** is 3.

From Wikipedia, the free encyclopedia

In information theory, the **Hamming distance** between two strings of equal length is the number of positions at which the corresponding symbols are different. In another way, it measures the minimum number of *substitutions* required to change one string into the other, or the minimum number of *errors* that could have transformed one string into the other.

FT1.2 Frame format

Figure shows the three frame formats in IEC 870-5-101 format class FT 1.2

Frame with variable length

is used for data transmission of user data between controlling and controlled station.

Start 68 H
L
L
Start 68 H
C
A
A
Link/ userdata
“
“
“
“
Checksum
End 16H

L

Frame with fixed length

is normally used for link layer services

Start 10H
C
A
Checksum
End 16H

- L Length field range 0 - 255
- L Specifies the number of subsequent user data octets including the control and the address fields
- C Control field
- A Address field (link)

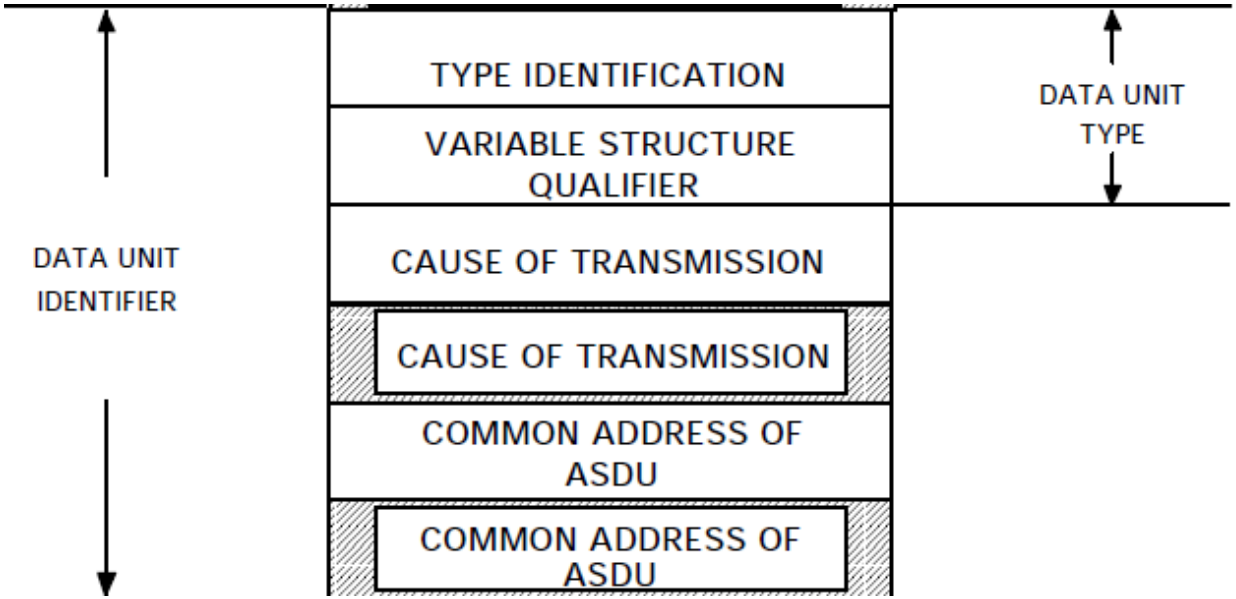
Single character

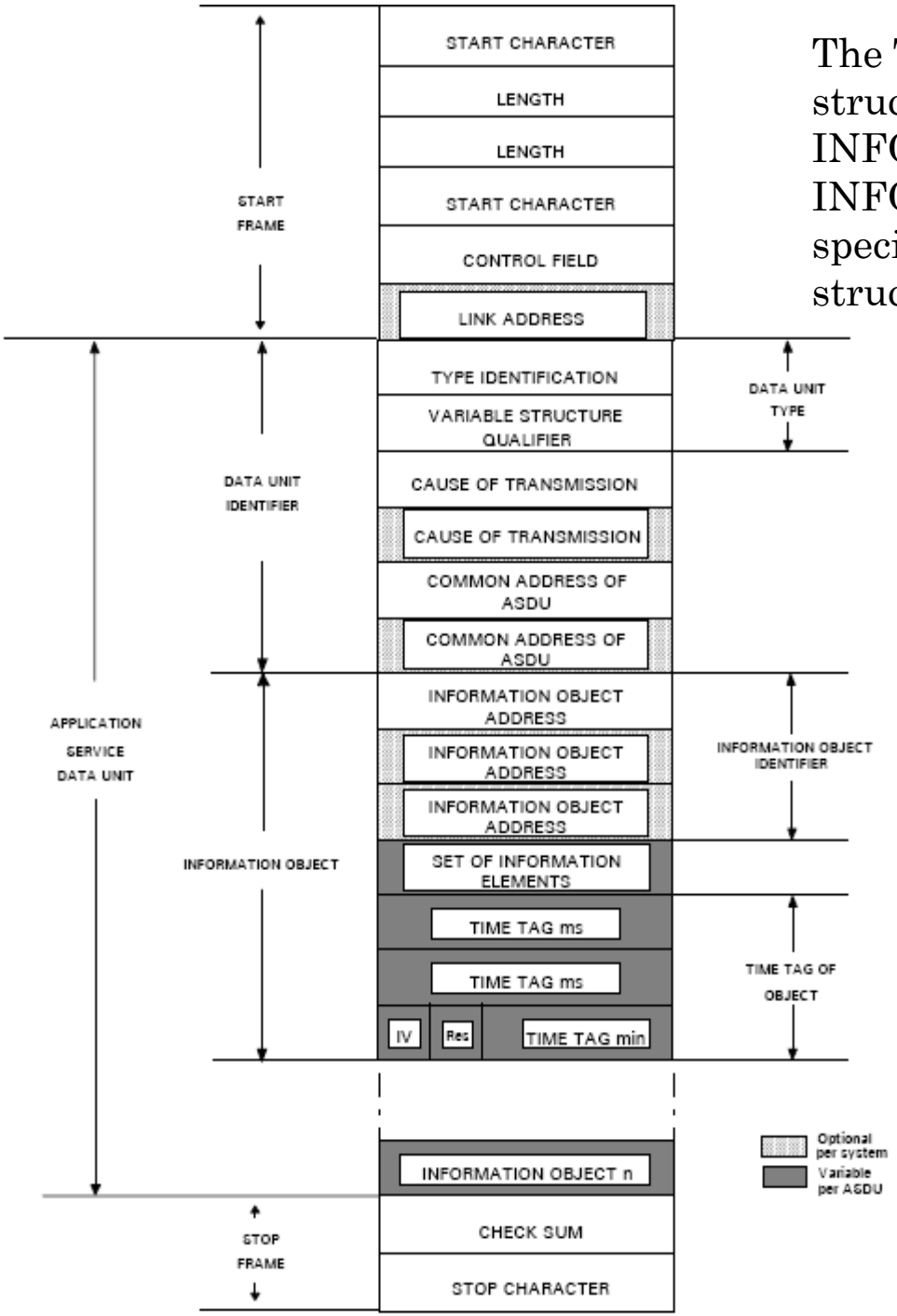
is normally used to confirm data on link services and to confirm user data

E 5 H

DATA UNIT IDENTIFIER

- The structure of the DATA UNIT IDENTIFIER is:
  - - one octet TYPE IDENTIFICATION
  - - one octet VARIABLE STRUCTURE QUALIFIER
  - - one or two octets CAUSE OF TRANSMISSION
  - - one or two octets COMMON ADDRESS OF ASDU



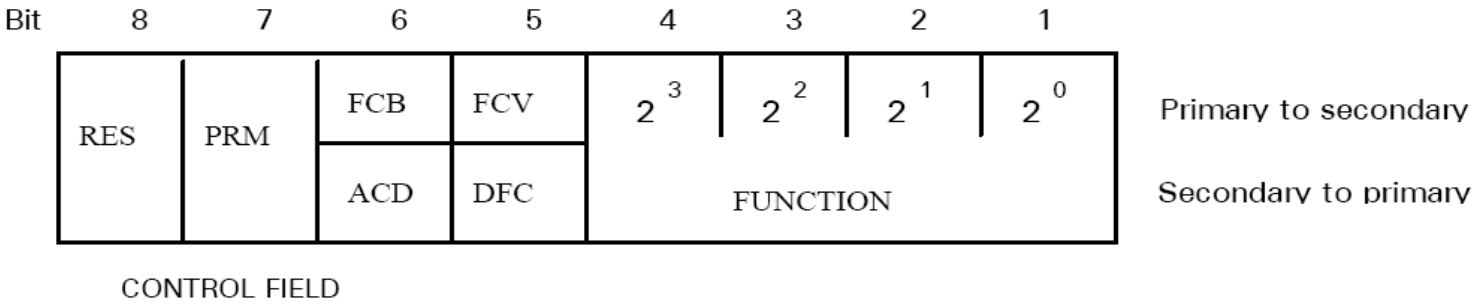


The TYPE IDENTIFICATION defines the structure, the type and the format of the INFORMATION OBJECT. All INFORMATION OBJECTs of a specific ASDU (telegrams) are of the same structure, type and format.

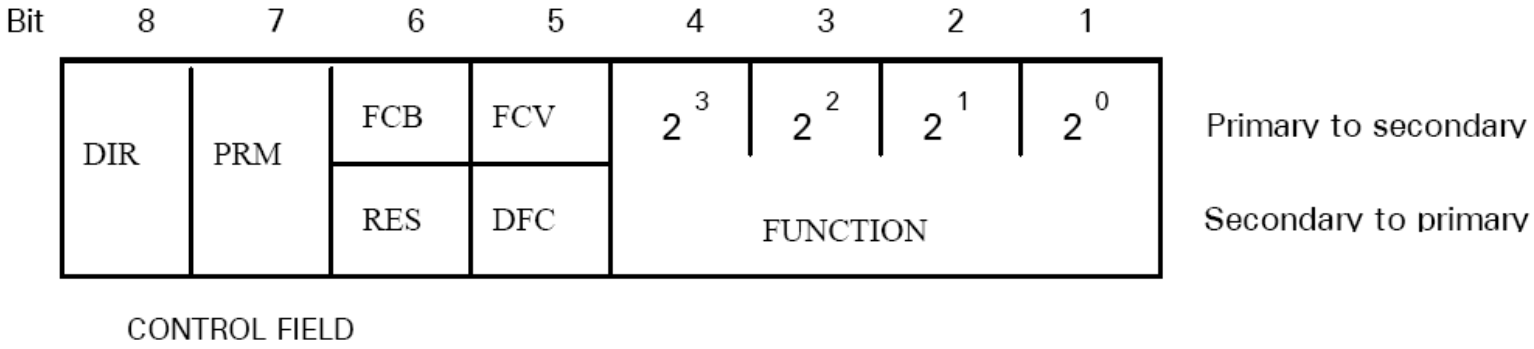
IEC 101 Frame Format, Variable length		
Data unit	**Name**	**Function**
Start Frame	Start Character	*Indicates start of Frame*
	Length Field (*2)	*Total length of Frame*
	Start Character (repeat)	*Repeat provided for reliability*
	Control Field	*Indicates control functions like message direction*
	Link Address (*0,1 or 2*)	*Normally used as the device / station address*
Data Unit Identifier	Type Identifier	*Defines the data type which contains specific format of information objects*
	Variable Structure Qualifier	*Indicates whether type contains multiple information objects or not*
	COT (*1 or 2*)	*Indicates causes of data transmissions like spontaneous or cyclic*
	ASDU Address (*1 or 2*)	*Denotes separate segments and its address inside a device*
Information Object	Information Object Address (*1 or 2 or 3*)	*Provides address of the information object element*
	Information Elements (n)	*Contains details of the information element depending on the type*
Information Object-2	-----	
-----	-----	
Information Object-m		
Stop Frame	Checksum	*Used for Error checks*
	Stop Char	*Indicates end of a frame*

Unbalanced and Balanced Communication

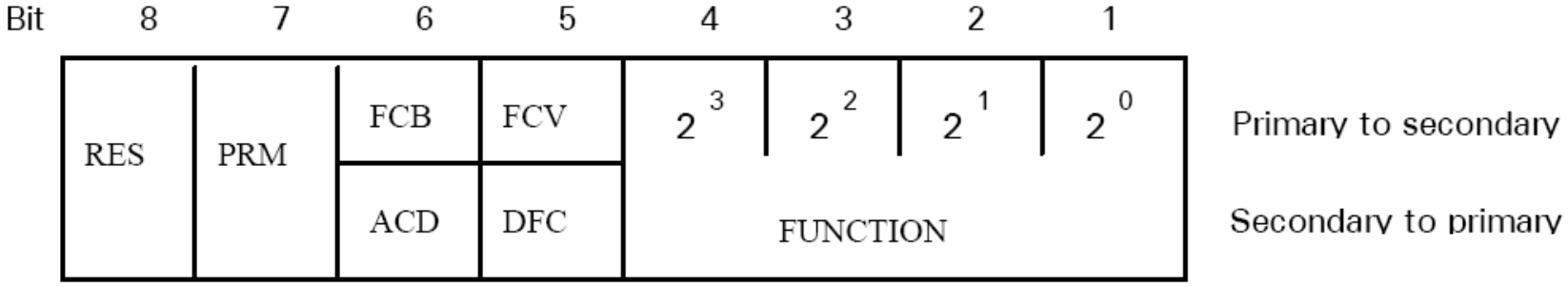
- Unbalanced : Master /Slave Master is always primary and slave is secondary . All Transactions Starting always from Master



- Balanced : Master and Slave can be primary (Point to Point)



CONTROL FIELD UNBALANCED



CONTROL FIELD

PRM Primary message
0 = message from secondary(responding) station.
1 = message from primary (initiating) station

FCB Frame count bit: 0 - 1 = alternating bit for successive SEND/CONFIRM or REQUEST/RESPOND services per station.
The frame count bit is used to delete losses and duplications of information transfers. The primary station alternate the FCB bit for each new SEND/CONFIRM or REQUEST/RESPOND transmission service directed to the same secondary station.
Thus the primary station keeps a copy of the frame count bit per secondary station.
If an expected reply is timed out (missing) or garbled, then the same SEND/CONFIRM and REQUEST/RESPOND service is repeated with the same frame count bit.
In case of reset commands the FCB bit is always zero, and upon receipt of these commands the secondary station will always be set to expect the next frame primary to secondary with FCV = valid (FCV = 1) to have the opposite setting of FCB, i.e. FCB equal to one

Control Field unbalanced

FCV Frame count bit valid.
0 = alternating function and FCB bit is invalid
1 = alternating function of FCB bit is valid
SEND/NO REPLY services, broadcast messages and other transmission services that
ignore the deletion of duplication or loss of information output do not alternate the FCB bit and indicates this by a cleared FCV bit

ACD Access demand.
There are two classes of message data provided, namely class 1 and 2.
0 = no access demand for class 1 data transmission
1 = access demand for class 1 data transmission
Class 1 data transmission is typically used for events or for messages with high priority.
Class 2 data transmission is typically used for cyclic transmission or for low priority messages.

DFC Data flow control
0 = further messages are acceptable
1 = further messages may cause data overflow
Secondary (responding) stations indicate to the message initiating (primary) station that an immediate secession of further message may cause a buffer overflow.

Function codes of control-field in messages sent from primary (PRM = 1)

Unbalanced mode

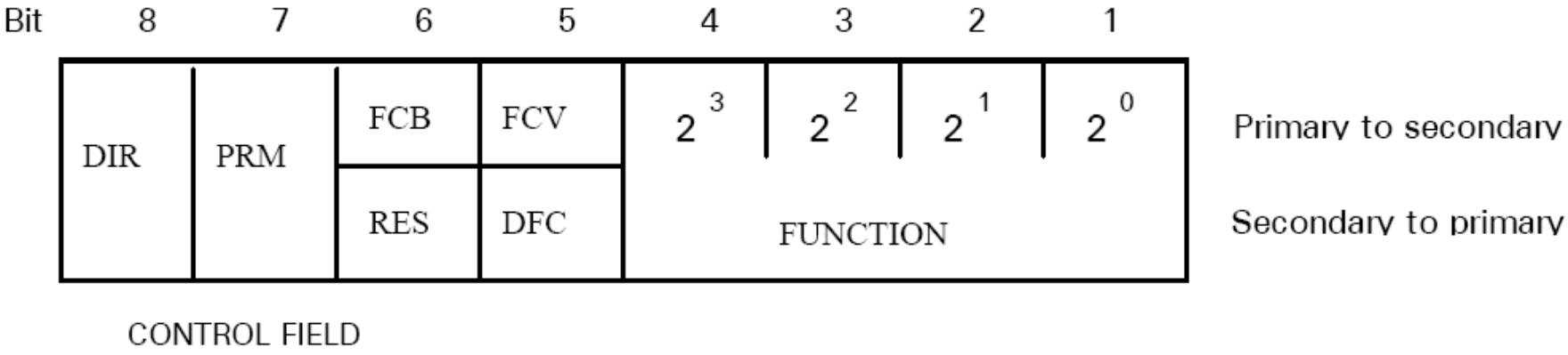
Function code no	Frame type	Service function	FCV
0	SEND/CONFIRM expected	Reset remote link	0
1	SEND/CONFIRM expected	Reset of user process	0
2	SEND/CONFIRM expected	Reserved for balanced mode	-
3	SEND/CONFIRM expected	User data	1
4	SEND/NO REPLY expected	User data	0
5		Reserved	-
6 - 7		Reserved for special use agreement	-
8	REQUEST for access demand	Expected response specifies access demand	0
9	REQUEST/RESPOND expected	Request status of link	0
10	REQUEST/RESPOND expected	Request user data class 1	1
11	REQUEST/RESPOND expected	Request user data class 2	1
12 - 13		Reserved	-
14 - 15		Reserved for special use by agreement	-

Function codes of control-field in messages sent from secondary (PRM = 0)

Unbalanced mode

Function code no	Frame type	Service function
0	CONFIRM	ACK: Positive acknowledgement
1	CONFIRM	NACK: message not accepted, link busy
2 - 5		Reserved
6 - 7		Reserved for special use by agreement
8	RESPOND	User data
9	RESPOND	NACK: requested data not available
10		Reserved
11	RESPOND	Status of link or access demand
12		Reserved
13		Reserved for special use by agreement
14		Link service not functioning
15		Link service not implemented

CONTROL FIELD BALANCED



DIR Physical transmission direction
DIR=1 Data from controlling to controlled station
DIR=0 Data from controlled to controlling station

PRM Primary message
0 =message from secondary (responding) station
1 = message from primary (initiating) station

Function codes of control-field in messages sent from primary (PRM = 1)

Balanced mode

Function code no	Frame type	Service function	FCV
0	SEND/CONFIRM expected	Reset remote link	0
1	SEND/CONFIRM expected	Reset of user process	0
2	SEND/CONFIRM expected	Test function for link	1
3	SEND/CONFIRM expected	User data	1
4	SEND/NO REPLY expected	User data	0
5		Reserved	-
6 - 7		Reserved for special use agreement	-
8		Reserved for unbalanced transmission procedure	-
9	REQUEST/RESPOND expected	Request status of link	0
10		Reserved for unbalanced transmission procedure	-
11		Reserved for unbalanced transmission procedure	-
12 - 13		Reserved	-
14 - 15		Reserved for special use by agreement	-

Function codes of control-field in messages sent from secondary (PRM = 0)

Balanced mode

Function code no	Frame type	Service function
0	CONFIRM	ACK: Positive acknowledgement
1	CONFIRM	NACK: message not accepted, link busy
2 - 5		Reserved
6 - 7		Reserved for special use by agreement
8		Reserved for unbalanced transmission procedure
9		Reserved for unbalanced transmission procedure
10		Reserved
11	RESPOND	Status of link
12		Reserved
13		Reserved for special use by agreement
14		Link service not functioning
15		Link service not implemented

Type Identification

7.2.1 Type identification

Octet 1, TYPE IDENTIFICATION defines structure, type and format of the following INFORMATION OBJECT(s).

TYPE IDENTIFICATION is defined as:

TYPE IDENTIFICATION := UI8[1..8]<1..255>

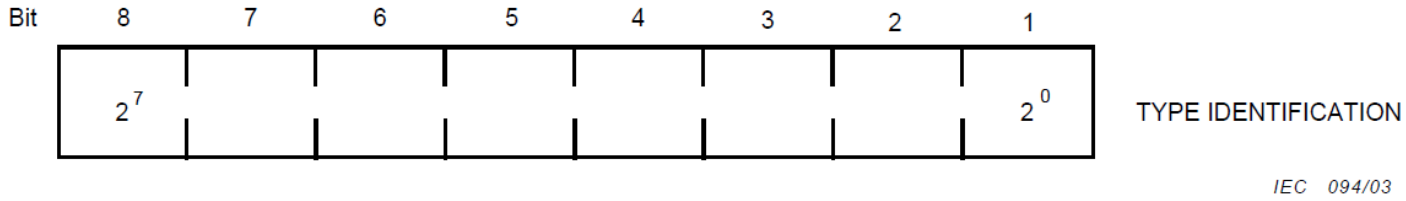


Figure 11 – Type identification

INFORMATION OBJECTS with or without TIME TAGS are distinguished with different numbers of the TYPE IDENTIFICATION.

Type Identification

TYPE IDENTIFICATION := UI8[1..8]<0..44>

<0>	:=	not defined	
<1>	:=	single-point information	M_SP_NA_1
<2>	:=	single-point information with time tag	M_SP_TA_1
<3>	:=	double-point information	M_DP_NA_1
<4>	:=	double-point information with time tag	M_DP_TA_1
<5>	:=	step position information	M_ST_NA_1
<6>	:=	step position information with time tag	M_ST_TA_1
<7>	:=	bitstring of 32 bit	M_BO_NA_1
<8>	:=	bitstring of 32 bit with time tag	M_BO_TA_1
<9>	:=	measured value, normalized value	M_ME_NA_1
<10>	:=	measured value, normalized value with time tag	M_ME_TA_1
<11>	:=	measured value, scaled value	M_ME_NB_1
<12>	:=	measured value, scaled value with time tag	M_ME_TB_1
<13>	:=	measured value, short floating point number	M_ME_NC_1
<14>	:=	measured value, short floating point number with time tag	M_ME_TC_1
<15>	:=	integrated totals	M_IT_NA_1
<16>	:=	integrated totals with time tag	M_IT_TA_1
<17>	:=	event of protection equipment with time tag	M_EP_TA_1
<18>	:=	packed start events of protection equipment with time tag	M_EP_TB_1
<19>	:=	packed output circuit information of protection equipment with time tag	M_EP_TC_1
<20>	:=	packed single-point information with status change detection	M_PS_NA_1
<21>	:=	measured value, normalized value without quality descriptor	M_ME_ND_1

Type Identifications

<30>	:= single-point information with time tag CP56Time2a	M_SP_TB_1
<31>	:= double-point information with time tag CP56Time2a	M_DP_TB_1
<32>	:= step position information with time tag CP56Time2a	M_ST_TB_1
<33>	:= bitstring of 32 bits with time tag CP56Time2a	M_BO_TB_1
<34>	:= measured value, normalized value with time tag CP56Time2a	M_ME_TD_1
<35>	:= measured value, scaled value with time tag CP56Time2a	M_ME_TE_1
<36>	:= measured value, short floating point number with time tag CP56Time2a	M_ME_TF_1
<37>	:= integrated totals with time tag CP56Time2a	M_IT_TB_1
<38>	:= event of protection equipment with time tag CP56Time2a	M_EP_TD_1
<39>	:= packed start events of protection equipment with time tag CP56Time2a	M_EP_TE_1
<40>	:= packed output circuit information of protection equipment with time tag CP56Time2a	M_EP_TF_1

Type Identifications

Table 9 – Semantics of TYPE IDENTIFICATION – Process information in control direction

TYPE IDENTIFICATION := UI8[1..8]<45..69>

CON	<45>	:= single command	C_SC_NA_1
CON	<46>	:= double command	C_DC_NA_1
CON	<47>	:= regulating step command	C_RC_NA_1
CON	<48>	:= set point command, normalized value	C_SE_NA_1
CON	<49>	:= set point command, scaled value	C_SE_NB_1
CON	<50>	:= set point command, short floating point number	C_SE_NC_1
CON	<51>	:= bitstring of 32 bits	C_BO_NA_1

Table 11 – Semantics of TYPE IDENTIFICATION – System information in control direction

TYPE IDENTIFICATION := UI8[1..8]<100..109>

CON	<100>	:= interrogation command	C_IC_NA_1
CON	<101>	:= counter interrogation command	C_CI_NA_1
CON	<102>	:= read command	C_RD_NA_1
CON	<103>	:= clock synchronization command	C_CS_NA_1
CON	<104>	:= test command	C_TS_NA_1
CON	<105>	:= reset process command	C_RP_NA_1
CON	<106>	:= delay acquisition command	C_CD_NA_1

7.2.2 Variable structure qualifier

Octet 2 of the DATA UNIT IDENTIFIER of the ASDU defines the VARIABLE STRUCTURE QUALIFIER which is specified in the following.

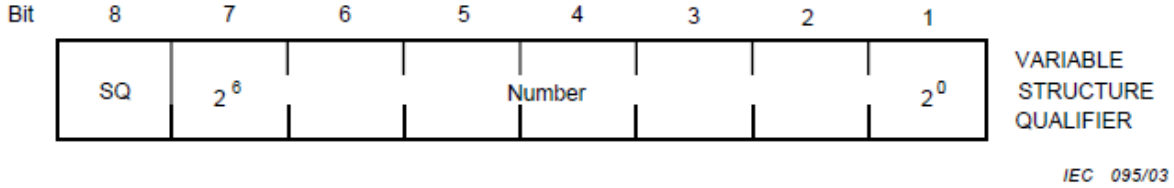


Figure 12 – VARIABLE STRUCTURE QUALIFIER

7.2.2.1 Definition of the semantics of the values of the VARIABLE STRUCTURE QUALIFIER field

VARIABLE STRUCTURE QUALIFIER := CP8{number, SQ}

number=N	:=	UI7[1..7]<0..127>
<0>	:=	ASDU contains no INFORMATION OBJECT
<1..127>	:=	number of INFORMATION OBJECTS or ELEMENTS (single elements or equal combinations of elements)
SQ=Single/sequence	:=	BS1[8] <0..1>
<0>	:=	addressing of individual SINGLE INFORMATION ELEMENTS or COMBINATION OF INFORMATION ELEMENTS in a number of INFORMATION OBJECTS of the same type
<1>	:=	addressing of a sequence of SINGLE INFORMATION ELEMENTS or equal COMBINATIONS OF INFORMATION ELEMENTS of a single object per ASDU
SQ<0>and N<0..127>	:=	number of INFORMATION OBJECTS i
SQ<1>and N<0..127>	:=	number of SINGLE INFORMATION ELEMENTS or COMBINATIONS OF INFORMATION ELEMENTS j

7.2.3 Cause of transmission

Octet 3 of the DATA UNIT IDENTIFIER of the ASDU defines the CAUSE OF TRANSMISSION field which is specified in the following.

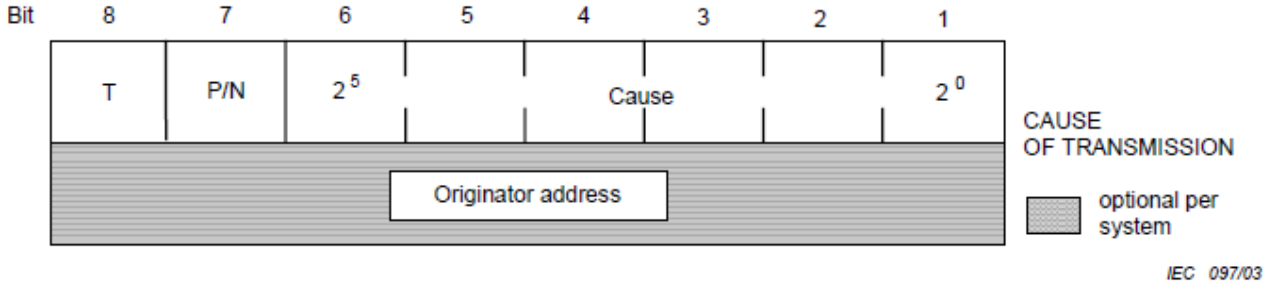


Figure 14 – CAUSE OF TRANSMISSION field

Definition of the semantics of the values of the CAUSE OF TRANSMISSION field

CAUSE OF TRANSMISSION	:=	CP16{Cause,P/N,T,Originator Address (opt)}
Cause	:=	UI6[1..6]<0..63>
<0>	:=	not defined
<1..63>	:=	number of cause
<1..47>	:=	for standard definitions of this companion standard (compatible range) see Table 14
<48..63>	:=	for special use (private range)
P/N	:=	BS1[7] <0..1>
<0>	:=	positive confirm
<1>	:=	negative confirm
T=test	:=	BS1[8] <0..1>
<0>	:=	no test
<1>	:=	test
Originator address	:=	UI8[9..16]<0..255>
<0>	:=	default
<1..255>	:=	number of originator address

Table 14 – Semantics of CAUSE OF TRANSMISSION

Cause	:=	UI6[1..6]<0..63>	
<0>	:=	not used	
<1>	:=	periodic, cyclic	per/cyc
<2>	:=	background scan[3]	back
<3>	:=	spontaneous	spont
<4>	:=	initialized	init
<5>	:=	request or requested	req
<6>	:=	activation	act
<7>	:=	activation confirmation	actcon
<8>	:=	deactivation	deact
<9>	:=	deactivation confirmation	deactcon
<10>	:=	activation termination	actterm
<11>	:=	return information caused by a remote command	retrem
<12>	:=	return information caused by a local command	retloc
<13>	:=	file transfer	file
<14..19>	:=	reserved for further compatible definitions	
<20>	:=	interrogated by station interrogation	inrogen
<21>	:=	interrogated by group 1 interrogation	inro1
<22>	:=	interrogated by group 2 interrogation	inro2
<23>	:=	interrogated by group 3 interrogation	inro3
<24>	:=	interrogated by group 4 interrogation	inro4
<25>	:=	interrogated by group 5 interrogation	inro5
<26>	:=	interrogated by group 6 interrogation	inro6
<27>	:=	interrogated by group 7 interrogation	inro7
<28>	:=	interrogated by group 8 interrogation	inro8
<29>	:=	interrogated by group 9 interrogation	inro9
<30>	:=	interrogated by group 10 interrogation	inro10
<31>	:=	interrogated by group 11 interrogation	inro11
<32>	:=	interrogated by group 12 interrogation	inro12
<33>	:=	interrogated by group 13 interrogation	inro13

7.2.4 COMMON ADDRESS OF ASDUs

Octet 4 and optionally 5 of the DATA UNIT IDENTIFIER of the ASDU define the station address which is specified in the following. The length of the COMMON ADDRESS (one or two octets) is a parameter which is fixed per system.

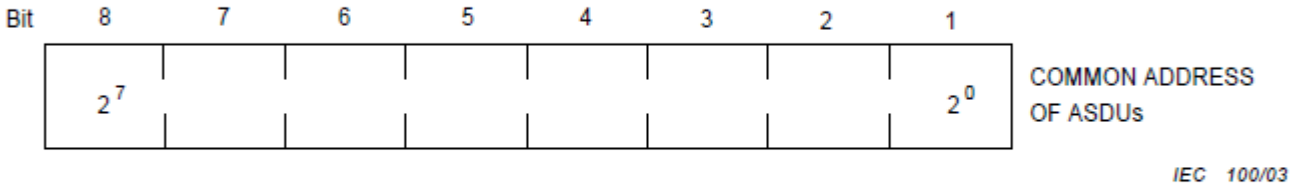


Figure 17 – COMMON ADDRESS of ASDUs (one octet)

COMMON ADDRESS := UI8[1..8]<0..255>

<0>	:=	not used
<1..254>	:=	station address
<255>	:=	global address

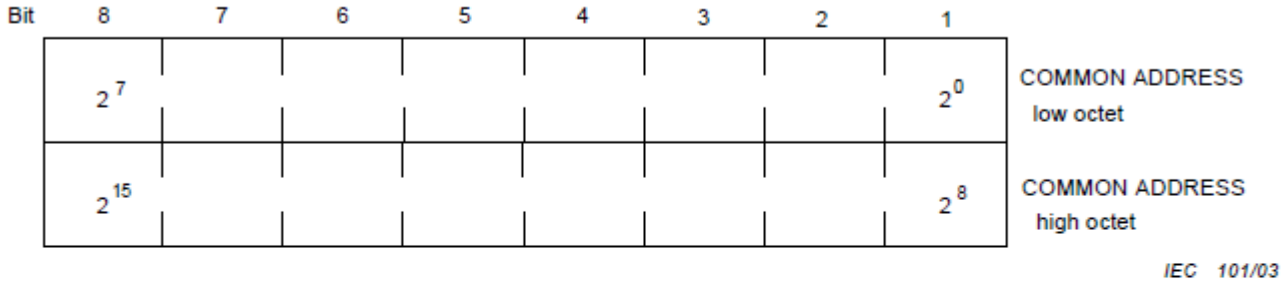


Figure 18 – COMMON ADDRESS of ASDUs (two octets)

COMMON ADDRESS	:=	UI16[1..16]<0..65535>
<0>	:=	not used
<1..65534>	:=	station address
<65535>	:=	global address

7.2.5 INFORMATION OBJECT ADDRESS

Octet 1, optionally 2 and optionally 3 of the INFORMATION OBJECT are defined in the following. The length of the INFORMATION OBJECT ADDRESS (one, two or three octets) is a parameter which is fixed per system.

The INFORMATION OBJECT ADDRESS is used as a destination address in control direction and a source address in the monitor direction.

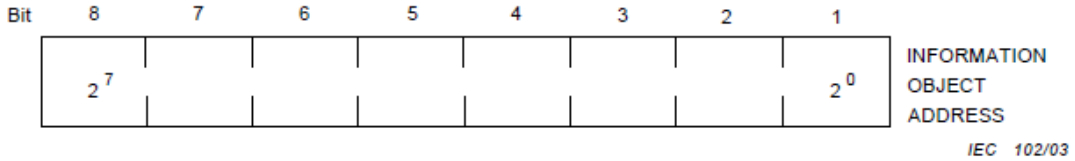


Figure 19 – INFORMATION OBJECT ADDRESS (one octet)

INFORMATION OBJECT ADDRESS := UI8[1..8]<0..255>

<0> := INFORMATION OBJECT ADDRESS is irrelevant

<1..255> := INFORMATION OBJECT ADDRESS

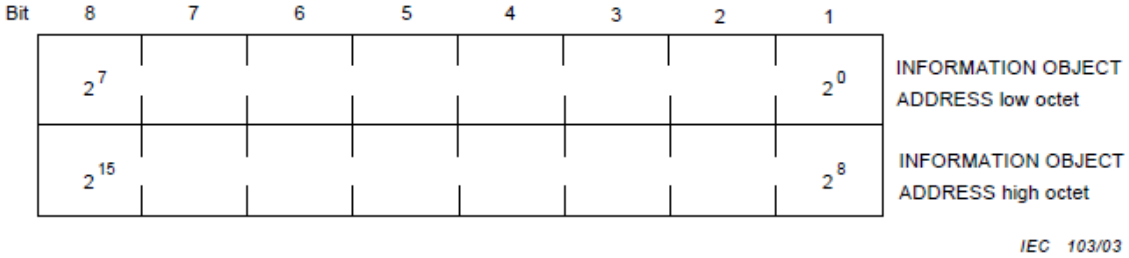


Figure 20 – INFORMATION OBJECT ADDRESS (two octets)

INFORMATION OBJECT ADDRESS := UI16[1..16]<0..65535>

<0>	:=	INFORMATION OBJECT ADDRESS is irrelevant
<1..65535>	:=	INFORMATION OBJECT ADDRESS

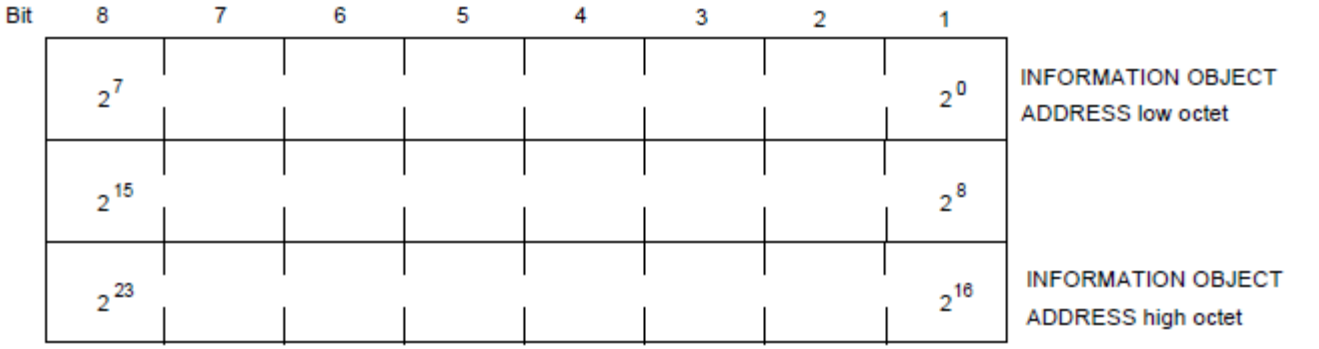


Figure 21 – INFORMATION OBJECT ADDRESS (three octets)

INFORMATION OBJECT ADDRESS := UI24[1..24]<0..16777215>

<0>	:=	INFORMATION OBJECT ADDRESS is irrelevant
<1..16777215>	:=	INFORMATION OBJECT ADDRESS

7.3.1 ASDUs for process information in monitor direction

7.3.1.1 TYPE IDENT 1: M_SP_NA_1

Single-point information without time tag

Sequence of information objects (SQ = 0)

0 0 0 0 0 0 0 1	TYPE IDENTIFICATION	DATA UNIT IDENTIFIER Defined in 7.1
0 Number i of objects	VARIABLE STRUCTURE QUALIFIER	
Defined in 7.2.3	CAUSE OF TRANSMISSION	
Defined in 7.2.4	COMMON ADDRESS OF ASDU	
Defined in 7.2.5	INFORMATION OBJECT ADDRESS	INFORMATION OBJECT 1
IV NT SB BL 0 0 0 SPI	SIQ = Single-point information with quality descriptor, defined in 7.2.6.1	
Defined in 7.2.5	INFORMATION OBJECT ADDRESS	INFORMATION OBJECT i
IV NT SB BL 0 0 0 SPI	SIQ = Single-point information with quality descriptor, defined in 7.2.6.1	

IEC 106/03

7.2.6.1 Single-point information (IEV 371-02-07) with quality descriptor

SIQ	:=	CP8{SPI,RES,BL,SB,NT,IV}	
SPI	:=	BS1[1]<0..1>	(Type 6)
<0>	:=	OFF	
<1>	:=	ON	
RES = RESERVE	:=	BS3[2..4]<0>	(Type 6)
BL	:=	BS1[5]<0..1>	(Type 6)
<0>	:=	not blocked	
<1>	:=	blocked	
SB	:=	BS1[6]<0..1>	(Type 6)
<0>	:=	not substituted	
<1>	:=	substituted	
NT	:=	BS1[7]<0..1>	(Type 6)
<0>	:=	topical	
<1>	:=	not topical	
IV	:=	BS1[8]<0..1>	(Type 6)
<0>	:=	valid	
<1>	:=	invalid	

Object Status

- OV = OVERFLOW/NO OVERFLOW The value of the INFORMATION OBJECT is beyond a predefined range of value (mainly applicable to analog values).
- BL = BLOCKED/NOT BLOCKED The value of the INFORMATION OBJECT is blocked for transmission; the value remains in the state that was acquired before it was blocked. Blocking and deblocking may be initiated for example by a local lock or a local automatic cause.
- SB = SUBSTITUTED/NOT SUBSTITUTED The value of the INFORMATION OBJECT is provided by the input of an operator (dispatcher) or by an automatic source.
- NT = NOT TOPICAL/TOPICAL A value is topical if the most recent update was successful. It is not topical if it was not updated successfully during a specified time interval or if it is unavailable.
- IV = INVALID/VALID A value is valid if it was correctly acquired. After the acquisition function recognizes abnormal conditions of the information source (missing or non-operating updating devices) the value is then marked invalid. The value of the INFORMATION OBJECT is not defined under this condition. The mark INVALID is used to indicate to the destination that the value may be incorrect and cannot be used.

Sequence of information elements in a single information object (SQ = 1)

0	0	0	0	0	0	1	1	TYPE IDENTIFICATION	DATA UNIT IDENTIFIER Defined in 7.1
1	Number j of elements							VARIABLE STRUCTURE QUALIFIER	
Defined in 7.2.3								CAUSE OF TRANSMISSION	
Defined in 7.2.4								COMMON ADDRESS OF ASDU	
Defined in 7.2.5								INFORMATION OBJECT ADDRESS A	INFORMATION OBJECT
IV	NT	SB	BL	0	0	DPI	1	DIQ = Double-point information with quality descriptor, defined in 7.2.6.2 Belongs to information object address A	
IV	NT	SB	BL	0	0	DPI	j	DIQ = Double-point information with quality descriptor, defined in 7.2.6.2 Belongs to information object address A+j–1	

IEC 110/03

Figure 26 – ASDU: M_DP_NA_1 Sequence of double-point information without time tag

7.2.6.2 Double-point information (IEV 371-02-08) with quality descriptor

DIQ		:=	CP8{DPI,RES,BL,SB,NT,IV}	
DPI		:=	UI2[1..2]<0..3>	(Type 1.1)
	<0>	:=	indeterminate or intermediate state	
	<1>	:=	determined state OFF	
	<2>	:=	determined state ON	
	<3>	:=	indeterminate state	
RES = RESERVE		:=	BS2[3..4]<0>	(Type 6)
BL		:=	BS1[5]<0..1>	(Type 6)
	<0>	:=	not blocked	
	<1>	:=	blocked	
SB		:=	BS1[6]<0..1>	(Type 6)
	<0>	:=	not substituted	
	<1>	:=	substituted	
NT		:=	BS1[7]<0..1>	(Type 6)
	<0>	:=	topical	
	<1>	:=	not topical	
IV		:=	BS1[8]<0..1>	(Type 6)
	<0>	:=	valid	
	<1>	:=	invalid	

7.3.1.9 TYPE IDENT 9: M_ME_NA_1
Measured value, normalized value

Sequence of information objects (SQ = 0)

0 0 0 0 1 0 0 1	TYPE IDENTIFICATION	DATA UNIT IDENTIFIER Defined in 7.1
0 Number i of objects	VARIABLE STRUCTURE QUALIFIER	
Defined in 7.2.3	CAUSE OF TRANSMISSION	
Defined in 7.2.4	COMMON ADDRESS OF ASDU	
Defined in 7.2.5	INFORMATION OBJECT ADDRESS	INFORMATION OBJECT 1
Value	NVA = Normalized value, defined in 7.2.6.6	
S Value		
IV NT SB BL 0 0 0 OV	QDS = Quality descriptor, defined in 7.2.6.3	
Defined in 7.2.5	INFORMATION OBJECT ADDRESS	INFORMATION OBJECT i
Value	NVA = Normalized value, defined in 7.2.6.6	
S Value		
IV NT SB BL 0 0 0 OV	QDS = Quality descriptor, defined in 7.2.6.3	

IEC 118/03

7.3.1.10 TYPE IDENT 10: M_ME_TA_1
Measured value, normalized value with time tag

Sequence of information objects (SQ = 0)

0 0 0 0 1 0 1 0	TYPE IDENTIFICATION	DATA UNIT IDENTIFIER Defined in 7.1
0 Number i of objects	VARIABLE STRUCTURE QUALIFIER	
Defined in 7.2.3	CAUSE OF TRANSMISSION	
Defined in 7.2.4	COMMON ADDRESS OF ASDU	
Defined in 7.2.5	INFORMATION OBJECT ADDRESS	INFORMATION OBJECT 1
Value	NVA = Normalized value, defined in 7.2.6.6	
S Value		
IV NT SB BL 0 0 0 OV	QDS = Quality descriptor, defined in 7.2.6.3	
CP24Time2a Defined in 7.2.6.19	Three octet binary time	
Defined in 7.2.5	INFORMATION OBJECT ADDRESS	INFORMATION OBJECT i
Value	NVA = Normalized value, defined in 7.2.6.6	
S Value		
IV NT SB BL 0 0 0 OV	QDS = Quality descriptor, defined in 7.2.6.3	
CP24Time2a Defined in 7.2.6.19	Three octet binary time	

IEC 120/03

Application Layer Functions

7.4 Selections from IEC 60870-5-5: Basic application functions

The following basic application functions, defined in IEC 60870-5-5 are used:

Station initialization (IEC 60870-5-5, 6.1)

Data acquisition by polling (IEC 60870-5-5, 6.2)

Cyclic data transmission (IEC 60870-5-5, 6.3)

Acquisition of events (IEC 60870-5-5, 6.4)

General interrogation (IEC 60870-5-5, 6.6)

Clock synchronization (IEC 60870-5-5, 6.7)

Command transmission (IEC 60870-5-5, 6.8)

Transmission of integrated totals (IEC 60870-5-5, 6.9)

Parameter loading (IEC 60870-5-5, 6.10)

Test procedure (IEC 60870-5-5, 6.11)

File transfer (IEC 60870-5-5, 6.12)

Acquisition of transmission delay (IEC 60870-5-5, 6.13)

If a controlled station has data for more than one of the following ASDU types ready for transmission at the same time, they must be sent in the following order regardless of which data was generated first. Table 16 does not define the order in which the controlling station must request the data or require that the controlled station not transmit data until another type of data becomes available. ASDU type identifications within the same row may be sent in any order. The chronological reporting requirements defined in 7.2.2.2 are valid.

Table 16 – Respond priorities of the controlled station

Request ASDU	Description	Comment
70	End of initialization	In monitor direction
45 to 69	Command transmission	Mirrored ASDUs
1 to 44 103 106	Event reporting Clock synchronization Acquisition of transmission delay	Event reporting: In monitor direction with COT = 3 Sequence of events and clock synchronization (See 6.7 of IEC 60870-5-5)
102, 104, 105, 110 to 113	Read command, test procedure, reset process, parameter loading,	
100, 101	Station interrogation, transmission of integrated totals	
9, 11, 13, 21 120 to 127	Cyclic data transmission (in monitor direction with COT = 1), file transfer	

7.4.2 Selections from data acquisition by polling

The complete function, defined in IEC 60870-5-5, 6.2, is used.

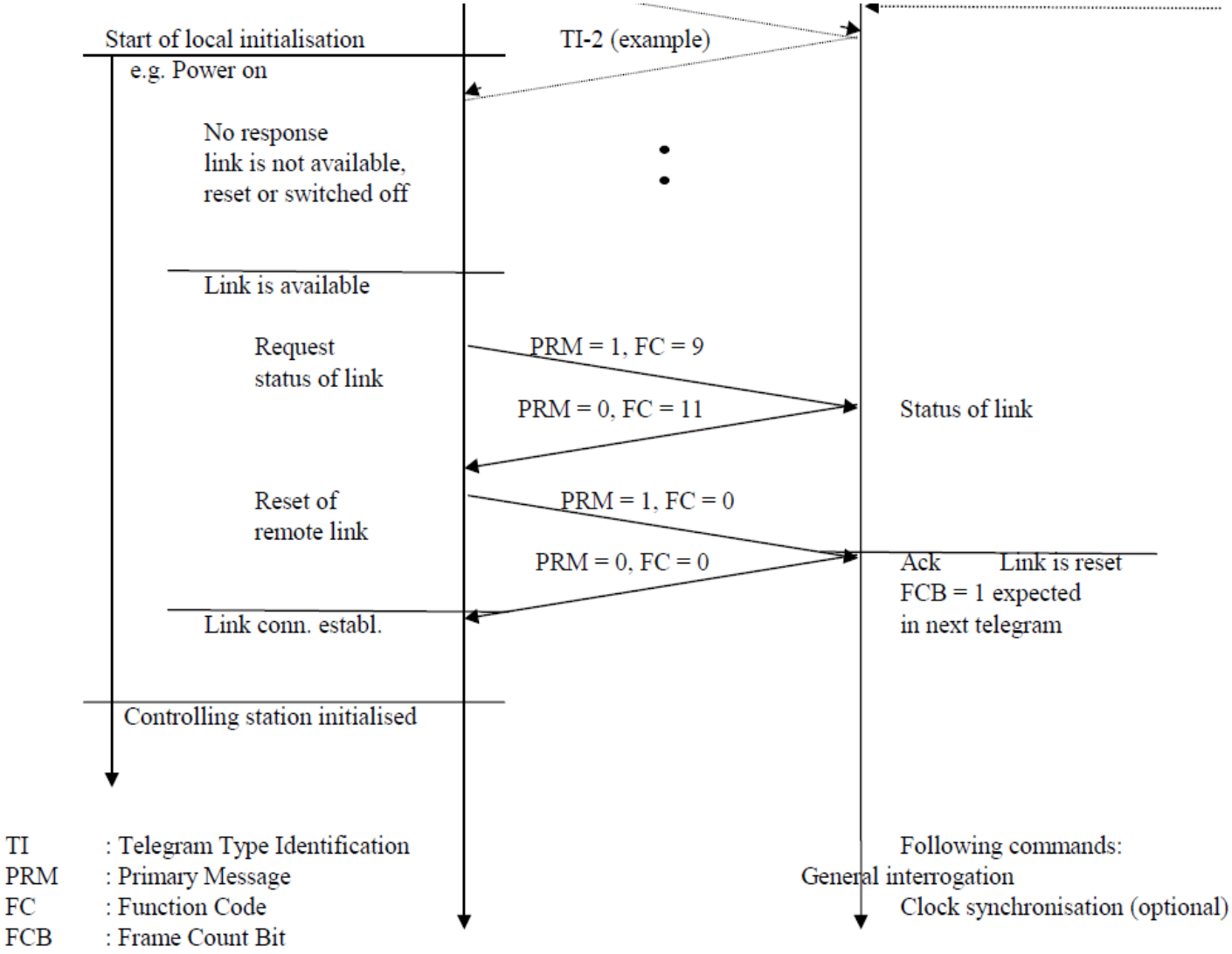
The polling procedure is supported by the link layer which requests user data of classes 1 and 2. In general, ASDUs containing the causes of transmission periodic/cyclic are assigned to be transmitted with the link layer data class 2 and all time tagged or spontaneously transmitted ASDUs are assigned to be transmitted with the link layer data class 1. Other ASDUs with other causes of transmission of low priority such as background scan may also be assigned to data class 2 and must be listed in the interoperability document.

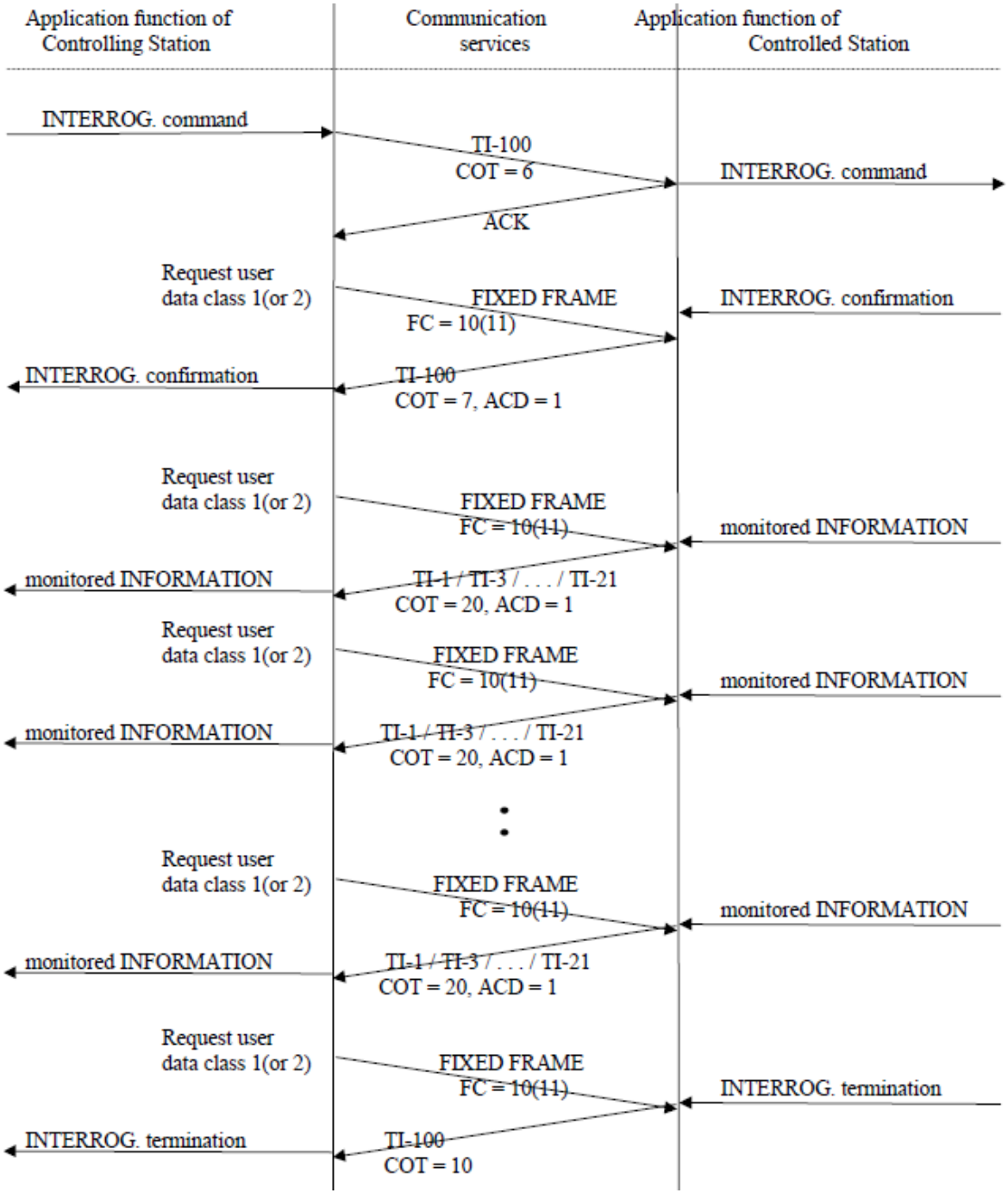
In this case, it has to be considered that the link request of class 1 occurs at a different point of time (to or from) the link request of class 2, which may influence the correct sequence of the ASDUs delivered to the application layer of the controlling station.

In response to a class 2 poll, a controlled station may respond with class 1 data when there is no class 2 data available.

When using the read command, specific information objects may be requested by interrogating their respective information object addresses. The requested information objects are returned with the cause of transmission <5> requested. Normally, these requested objects do not include the time tag.

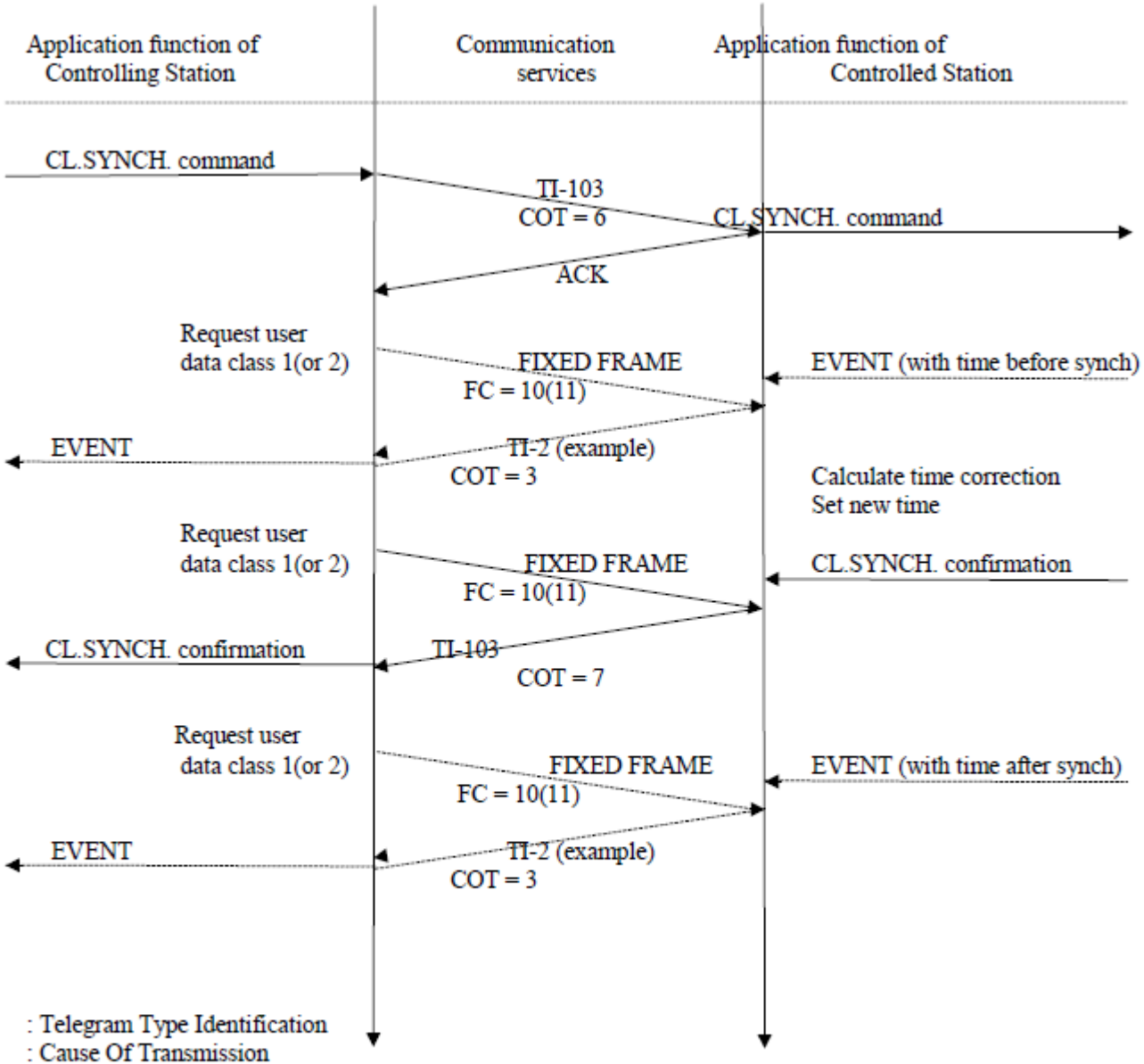
Local initialization of controlling station - unbalanced systems





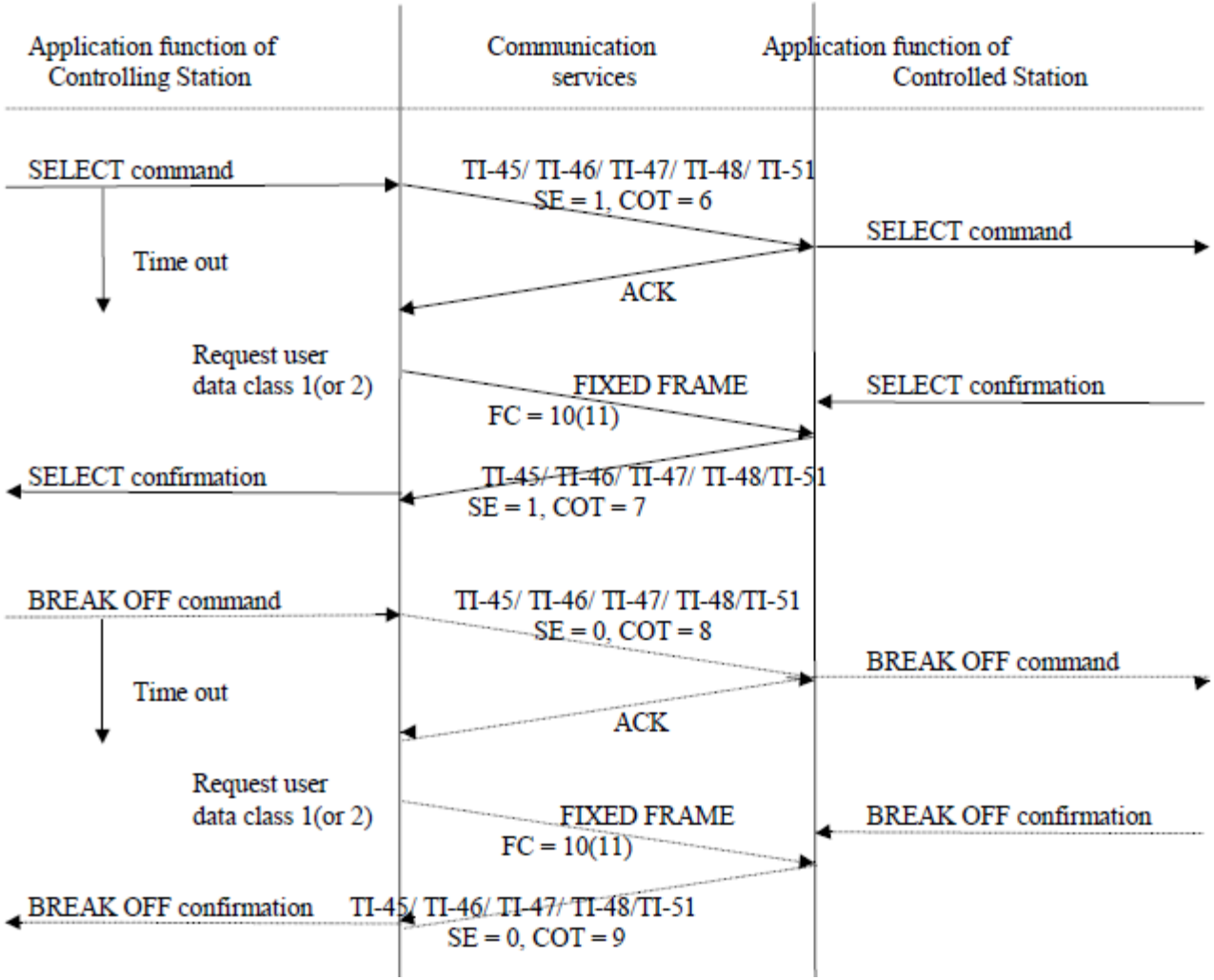
TI : Telegram Type Identification
COT : Cause Of Transmission
FC : Function Code
ACD : Access Demand bit
ACK : Single character or fixed frame ACK

Fig 6.10 *Interrogation procedure - unbalanced systems*



TI : Telegram Type Identification
COT : Cause Of Transmission
FC : Function Code
ACD : Access Demand bit
ACK : Single character or fixed frame ACK

Fig. 6.13 *Clock synchronisation procedure - unbalanced systems*


Application function of Controlling Station
Communication services
Application function of Controlled Station
SELECT command
TI-45/ TI-46/ TI-47/ TI-48/ TI-51
SE = 1, COT = 6
SELECT command
Time out
ACK
Request user data class 1(or 2)
FIXED FRAME
FC = 10(11)
SELECT confirmation
SELECT confirmation
TI-45/ TI-46/ TI-47/ TI-48/TI-51
SE = 1, COT = 7
BREAK OFF command
TI-45/ TI-46/ TI-47/ TI-48/TI-51
SE = 0, COT = 8
BREAK OFF command
Time out
ACK
Request user data class 1(or 2)
FIXED FRAME
FC = 10(11)
BREAK OFF confirmation
BREAK OFF confirmation
TI-45/ TI-46/ TI-47/ TI-48/TI-51
SE = 0, COT = 9

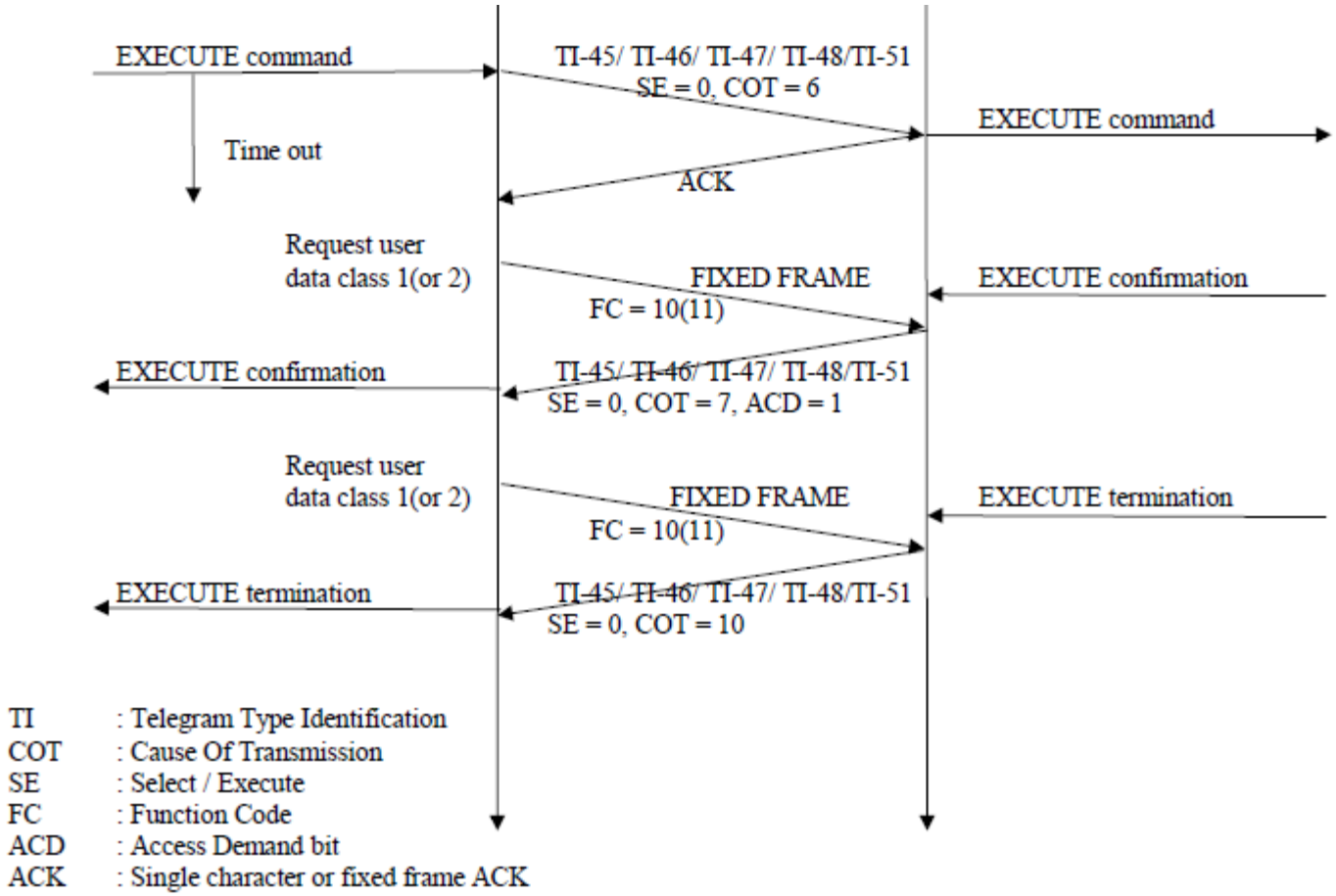
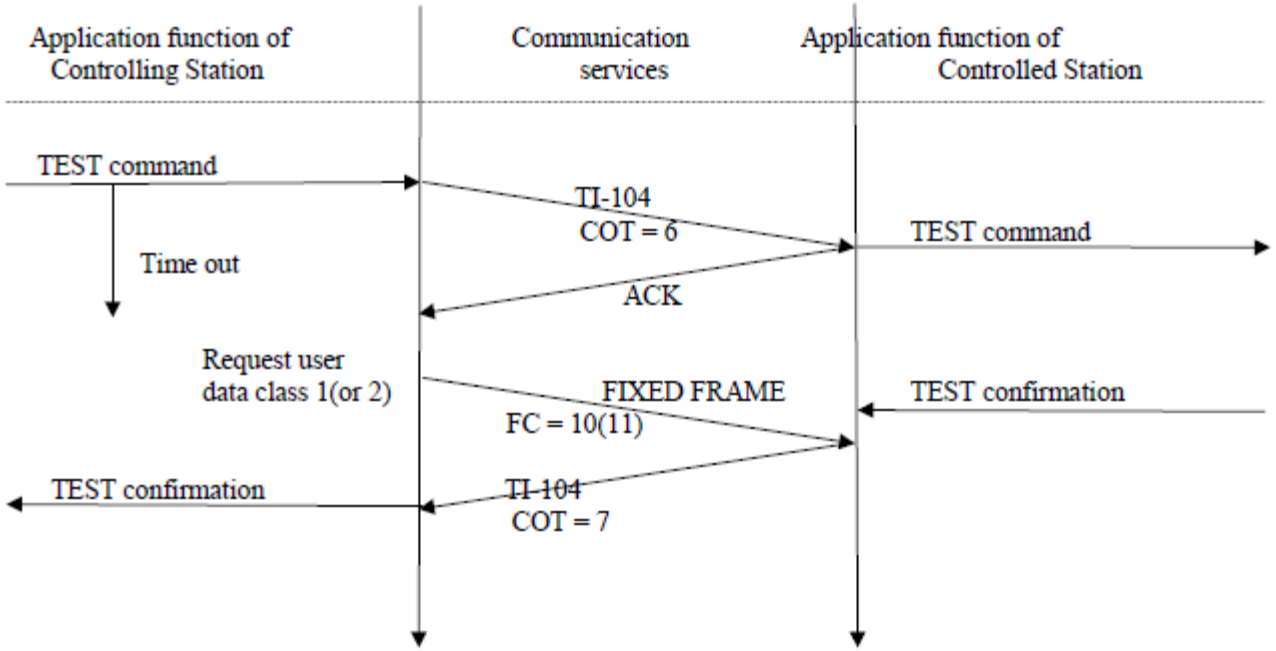


Fig. 6.15 *Command transmission procedure - unbalanced systems*



TI : Telegram Type Identification
COT : Cause Of Transmission
FC : Function Code
ACD : Access Demand bit

Fig. 6.21 *Test procedure - unbalanced systems*

IEC870-5-104

APCI Application Protocol Control Information
ASDU Application Service Data Unit
APDU Application Protocol Data Unit

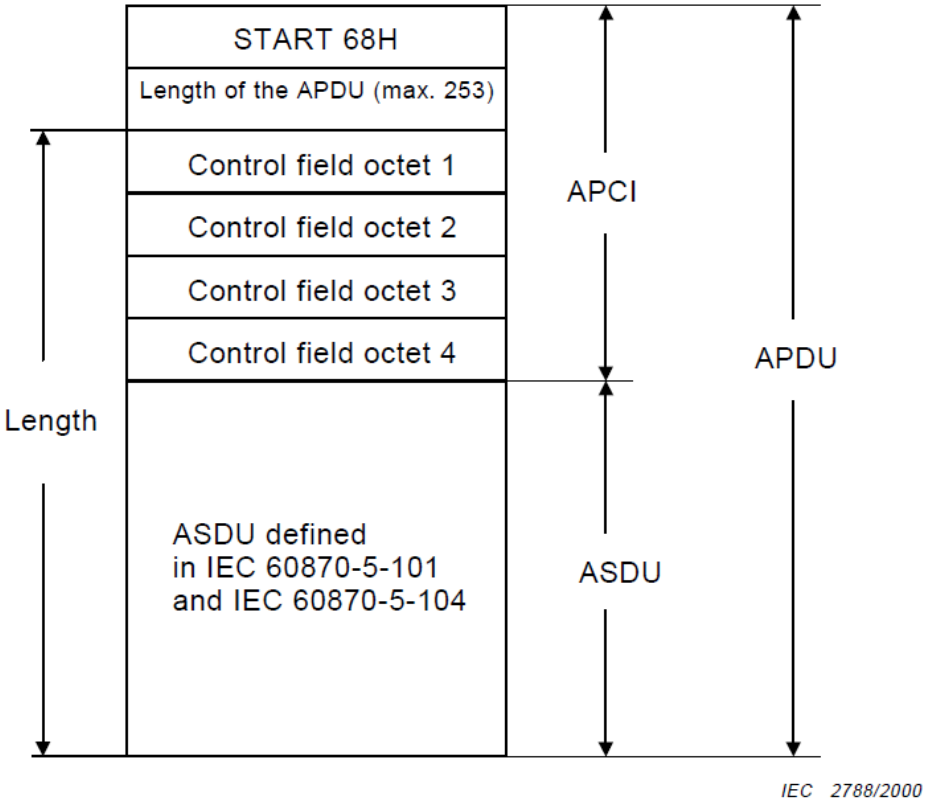


Figure 4 – APDU of the defined telecontrol companion standard

Control Field types

- Three types of control field formats are used to perform
  - numbered information transfer (I format),
  - numbered supervisory functions (S format)
  - unnumbered control functions (U format).

I Format Control Field

Control field octet 1 bit 1 = 0 defines the I format. I format APDUs always contain an ASDU. The control information of an I format is shown in figure 6.

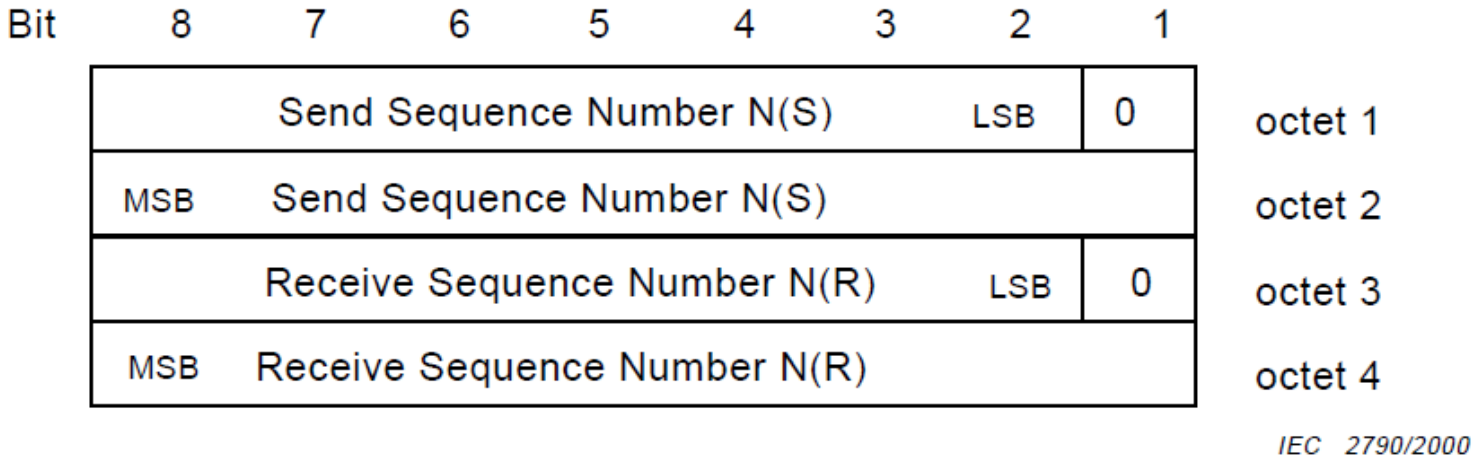


Figure 6 – Control field of type Information transfer format (I format)

S Format Control Field

Control field octet 1 bit 1 = 1 and bit 2 = 0 defines the S format. S format APDUs consist of the APCI only. The control information of an S format is shown in figure 7.

Bit	8	7	6	5	4	3	2	1	
	0						0	1	octet 1
	0								octet 2
	Receive Sequence Number N(R)						LSB	0	octet 3
	MSB	Receive Sequence Number N(R)							octet 4

IEC 2791/2000

Figure 7 – Control field of type numbered supervisory functions (S format)

U Format Control Field

Control field octet 1 bit 1 = 1 and bit 2 = 1 defines the U format. U format APDUs consist of the APCI only. The control information of a U format is shown in figure 8. Only one function – TESTFR, STOPDT or STARTDT – may be active at the same time.

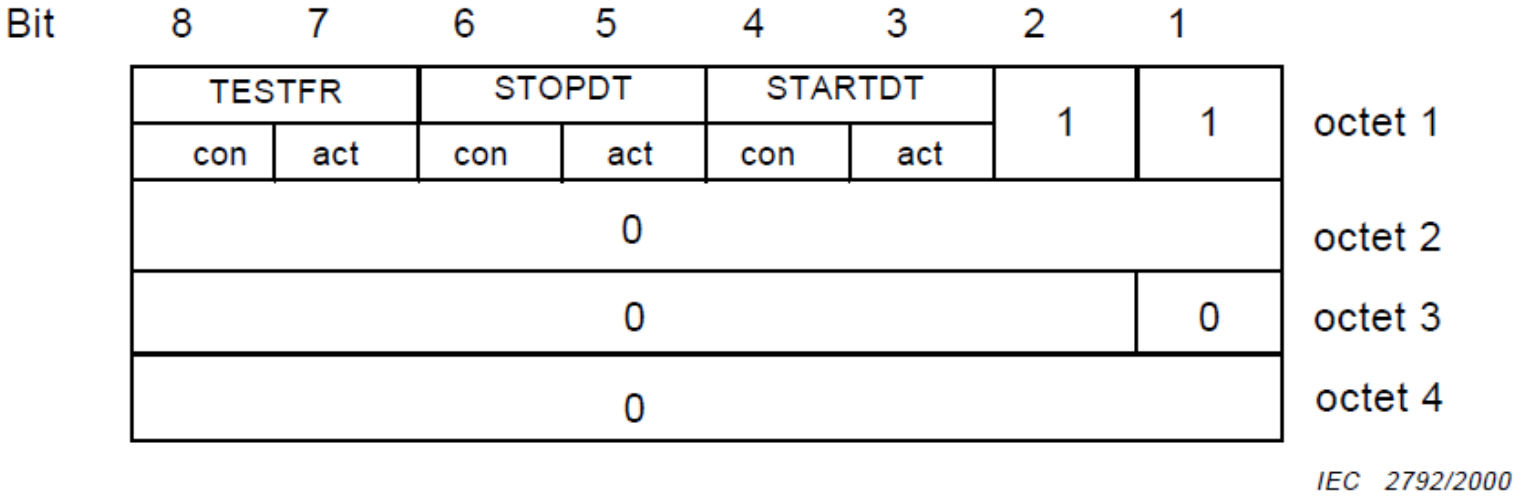
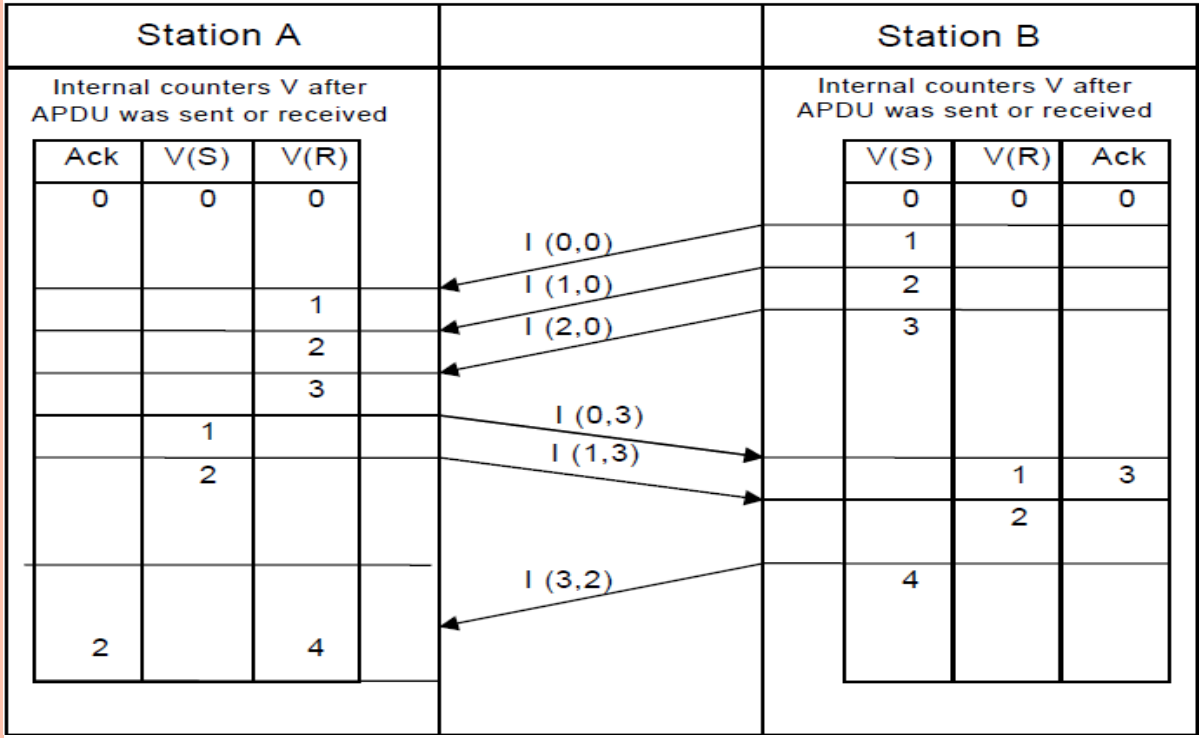


Figure 8 – Control field of type unnumbered control functions (U format)

N(S) , N(R)

Both sequence numbers are sequentially increased by one for each APDU and each direction. The transmitter increases the Send Sequence Number N(S) and the receiver increases the Receive Sequence Number N(R). The receiving station acknowledges each APDU or a number of APDUs when it returns the Receive Sequence Number up to the number whose APDUs are properly received. The sending station holds the APDU or APDUs in a buffer until it receives back its own Send Sequence Number as a Receive Sequence Number which is a valid acknowledge for all numbers <= the received number. Then it may delete the correctly transmitted APDUs from the buffer. In case of longer data transmission in one direction only, an S format has to be sent in the other direction to acknowledge the APDUs before buffer overflow or time out. This method should be used in both directions. After the establishment of a TCP connection, the send and receive sequence numbers are set to zero.

UNDISTURBED SEQUENCES OF NUMBERED I FORMAT APDUS



V(S) = Send state variable (see ITU-T X.25);

V(R) = Receive state variable (see ITU-T X.25);

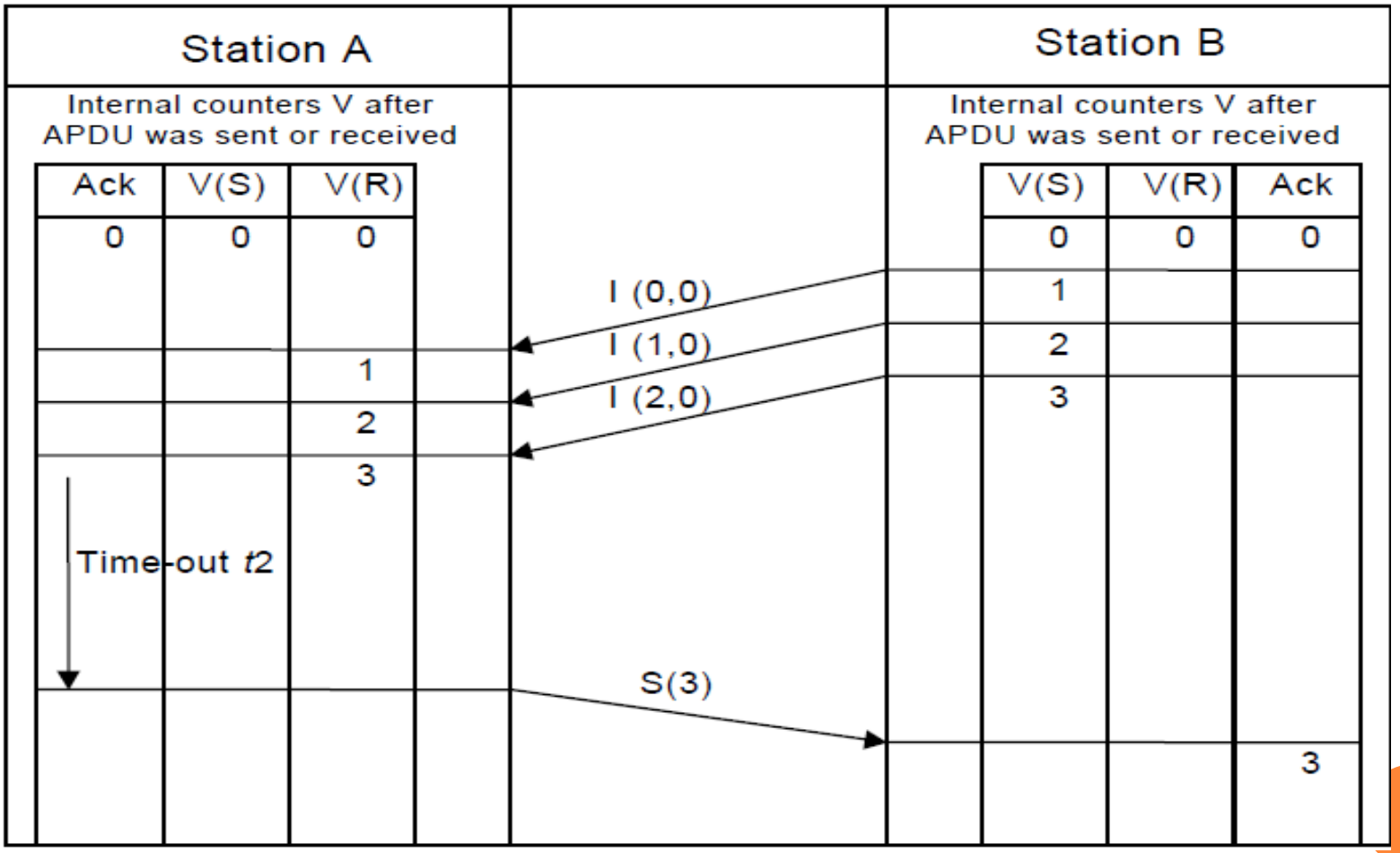
Ack = Indicates that the DTE has received correctly all I format APDUs numbered up to and including this number;

I(a,b) = Information format APDU with a = send sequence number and b = receive sequence number;

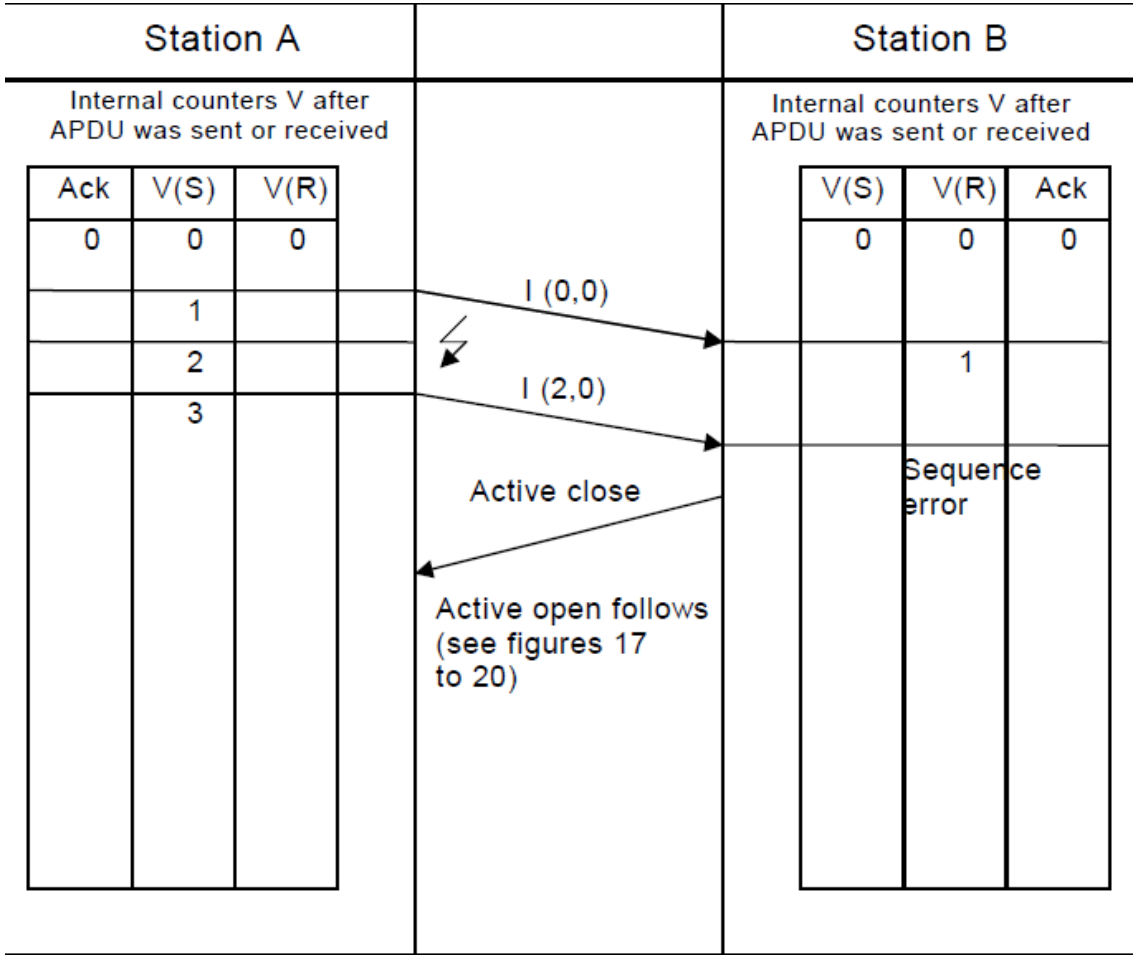
S(b) = Supervisory format APDU with b = receive sequence number;

U = Unnumbered control function APDU.

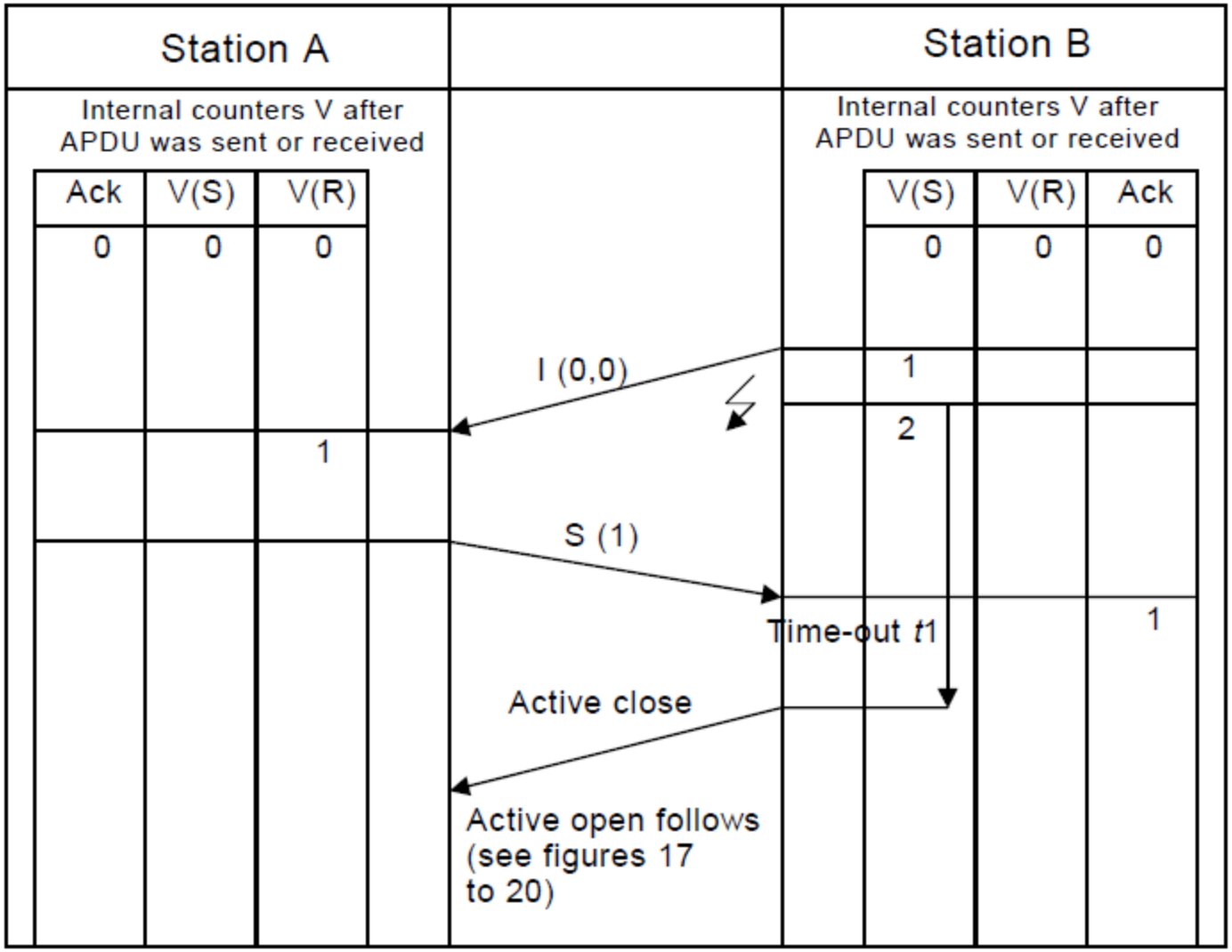
Undisturbed sequences of numbered I format APDUs acknowledged by an S format APDU



Disturbed sequence of numbered I format APDUs



Time-out in case of a not acknowledged last I format APDU



APCI Control Field

Control field formats

Two types of control field formats: I-Format, S-Format are used to perform numbered information transfer.

The third: U-Format control field is used to perform unnumbered link layer control functions.

I-Format

byte\bit	7	6	5	4	3	2	1	0
0	Send sequence number N(S) LSB							0
1	Send sequence number N(S) MSB							
2	Receive sequence number N(R) LSB							0
3	Receive sequence number N(R) MSB							

S-Format

byte\bit	7	6	5	4	3	2	1	0
0	0						0	1
1	0							
2	Receive sequence number N(R) LSB							0
3	Receive sequence number N(R) MSB							

U-Format

byte\bit	7	6	5	4	3	2	1	0
0	TESTFR		STOPDT		STARTDT		1	1
1	0							
2	0							
3	0							

STARTDT , STOPDT

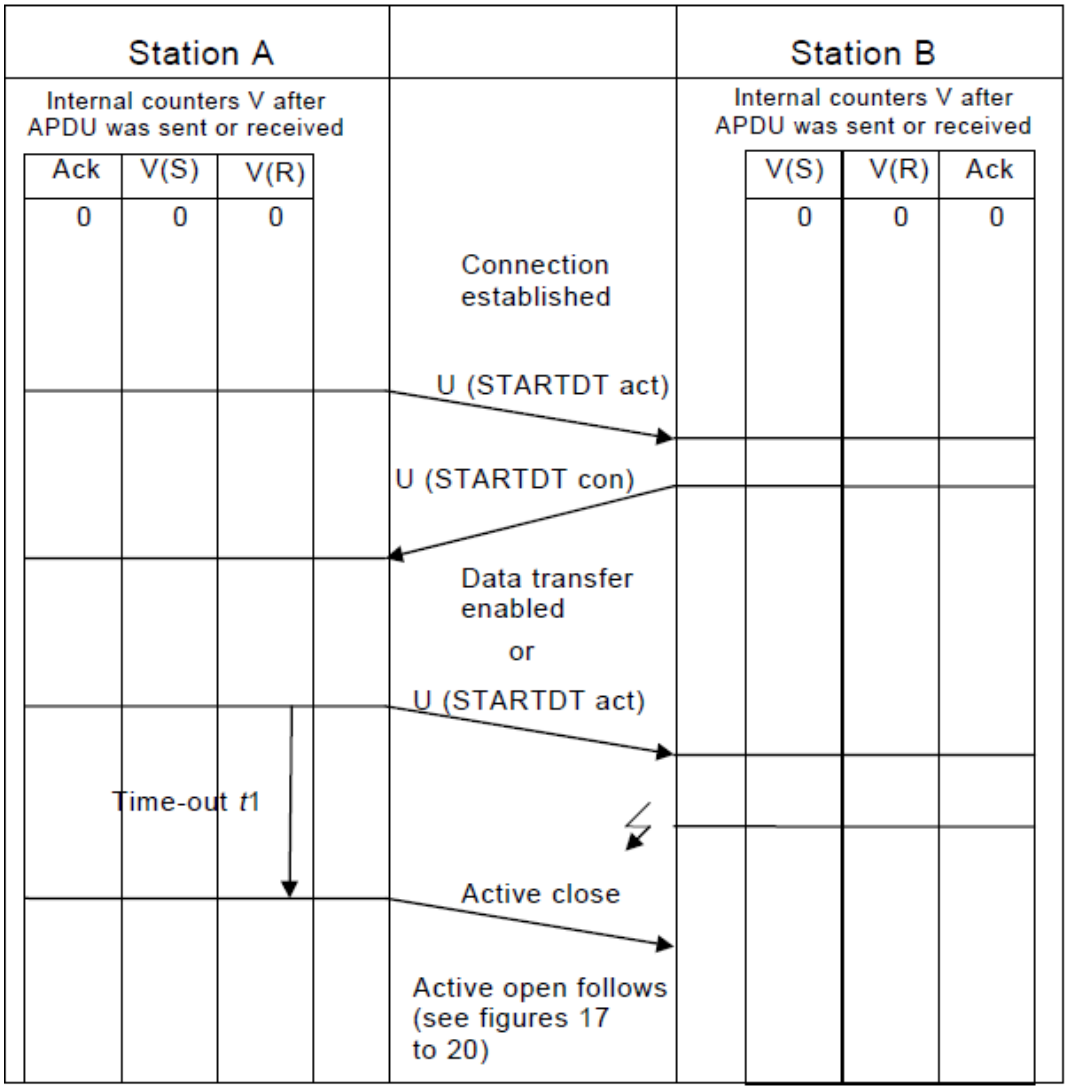
- STARTDT (Start Data Transfer) and STOPDT (Stop Data Transfer) are used by the controlling station (for example, Station A), to control the data transfer from a controlled station (Station B).

- When the connection is established, user data transfer is not automatically enabled from the controlled station on that connection, i.e. STOPDT is the default state when a connection is established. In this state, the controlled station does not send any data via this connection, except unnumbered control functions and confirmations to such functions. The controlling station must activate the user data transfer on a connection by sending a STARTDT act via this connection. The controlled station responds to this command with a STARTDT con. If the STARTDT is not confirmed, the connection is closed by the controlling station. This implies that after station initialization (see 7.1) STARTDT must always be sent before any user data transfer from the controlled station (for example, general interrogated information) is initiated. Any pending user data in the controlled station is sent only after the STARTDT con.

STARTDT , STOPDT (START/STOP DATA TRANSFER)

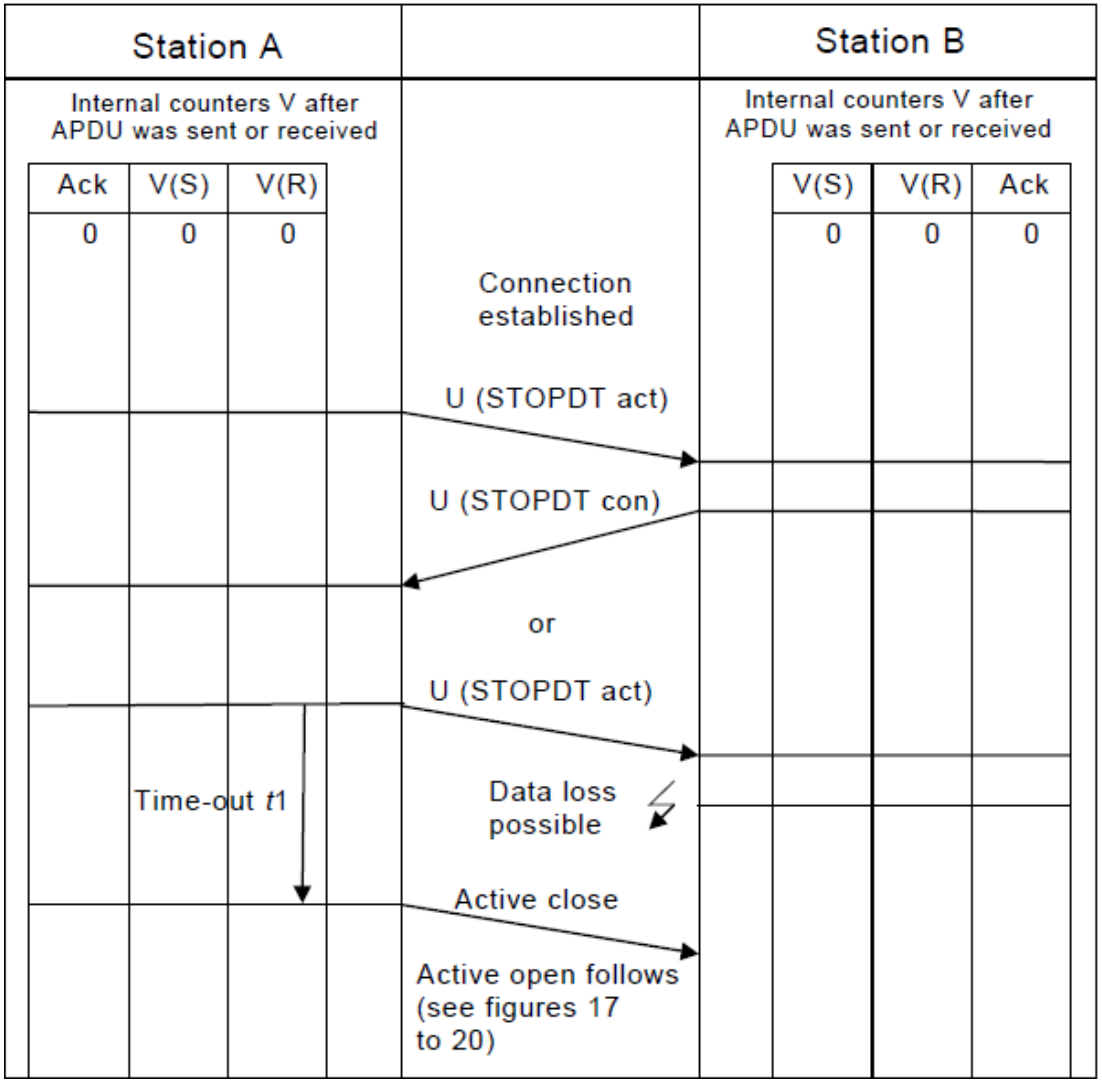
- Refer to IEC 60870-5-104 clause 5.3. Only the controlling station sends the STARTDT. The expected mode of operation is that the STARTDT is sent only once after the initial establishment of the connection (or re-establishment of a connection). The connection then operates with both controlled and controlling stations permitted to send any message at any time until the controlling station decides to close the connection with a STOPDT command (or the connection fails and is automatically closed after the timeouts expire).

START DATA TRANSFER PROCEDURE



STARTDT/STOPDT is a mechanism for the controlling station to activate/deactivate the monitoring direction. The controlling station may send commands or setpoints even if it has not
yet received the activation confirmation. Send and receive counters continue their functionality
independent of the use of STARTDT/STOPDT

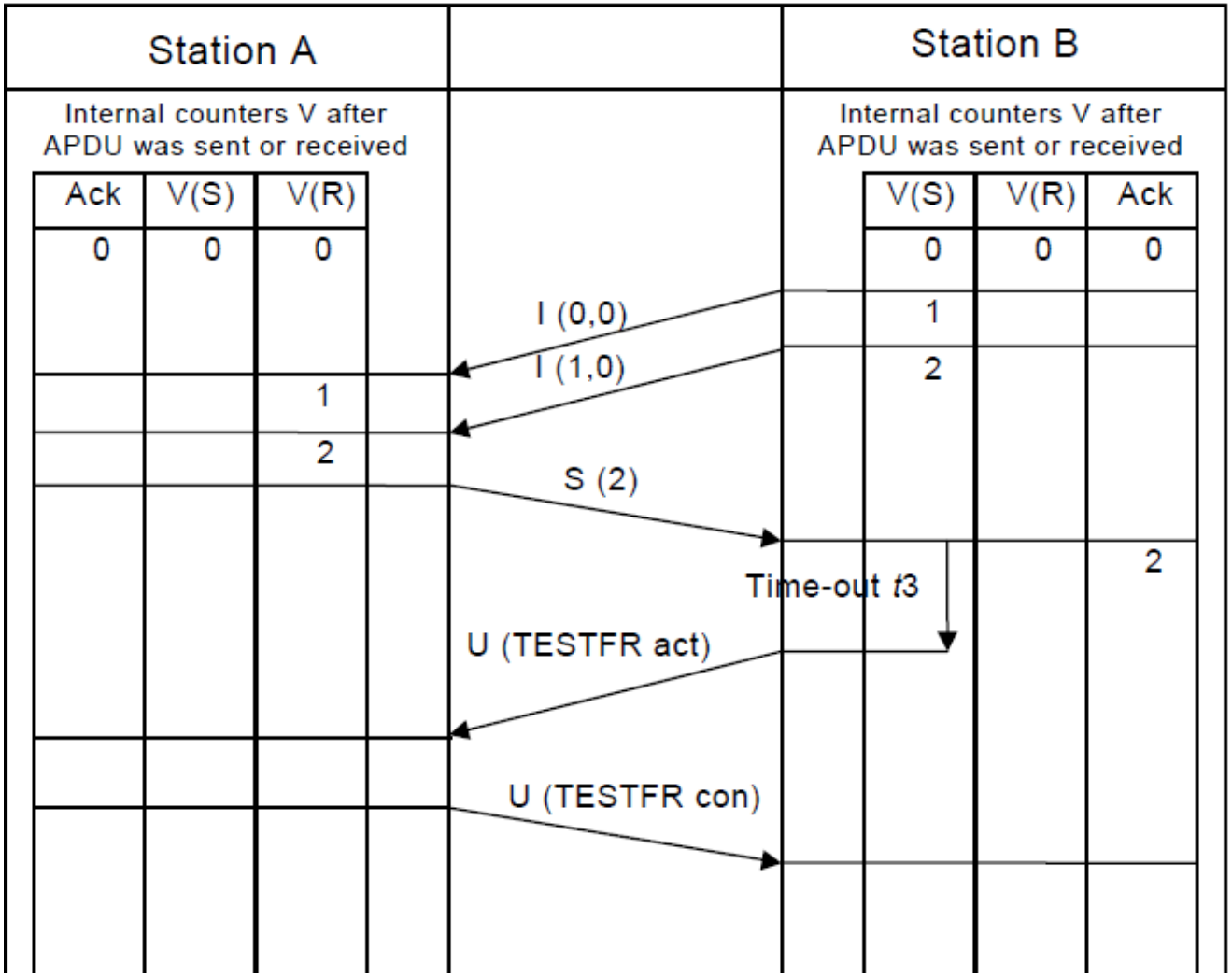
STOP DATA TRANSFER PROCEDURE



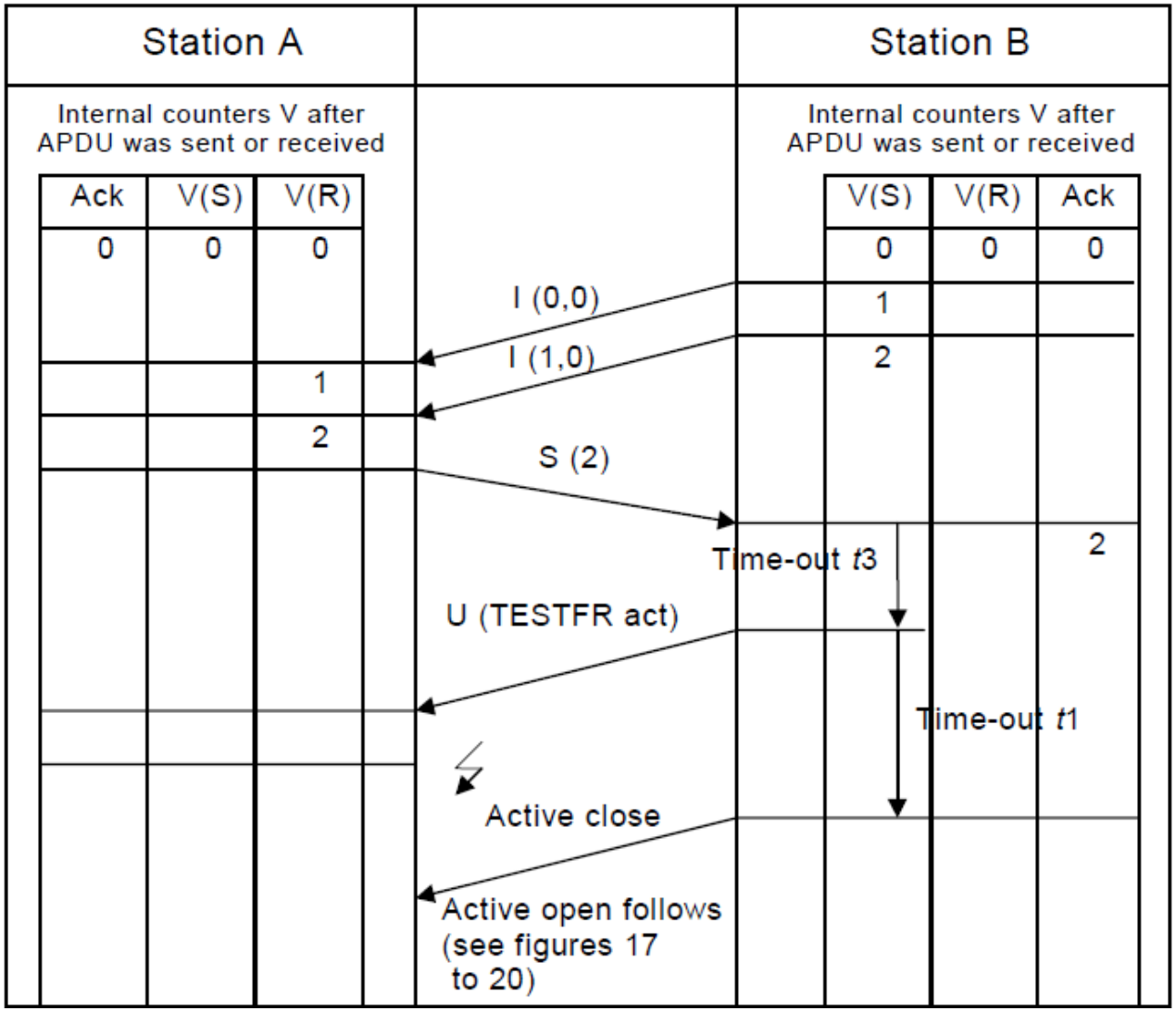
TESTFR

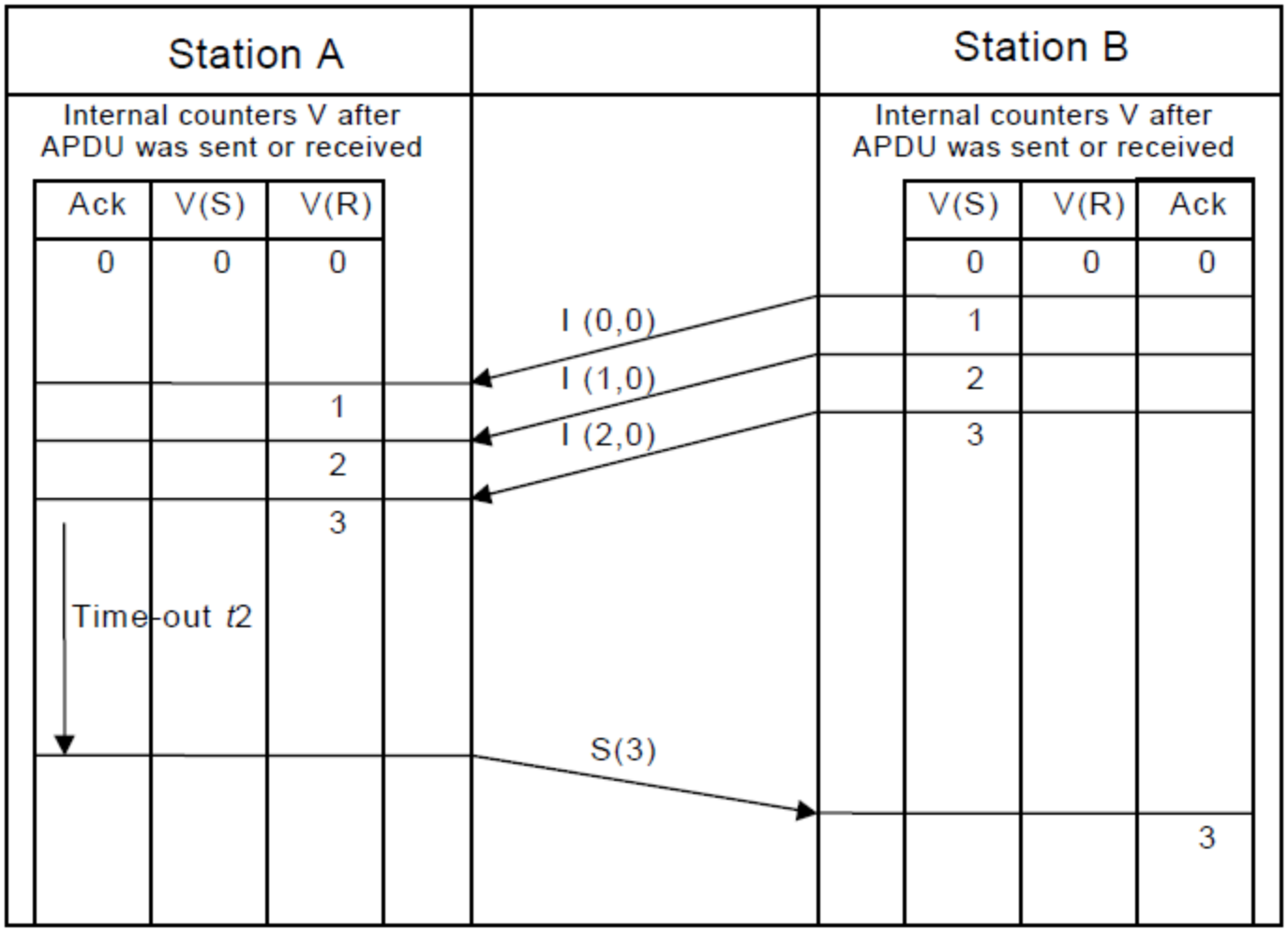
- The controlling and/or controlled station must regularly check the status of all established connections to detect any communication problems as soon as possible. This is done by sending TESTFR frames
- Unused, but open, connections may be periodically tested in both directions by sending test APDUs (TESTFR = act) which are confirmed by the receiving station sending TESTFR = con. Both stations may initiate the test procedure after a specified period of time in which no data transfers occur (time out). The reception of every frame – I frame, S frame or U frame – retriggers timer t3.

UNDISTURBED TEST PROCEDURE



UNCONFIRMED TEST PROCEDURE





IEC 2794/2000

Figure 10 – Undisturbed sequences of numbered I format APDUs acknowledged by an S format APDU

Port Number

Every TCP address consists of an IP address and a portnumber. Every equipment connected to the TCP-LAN has its particular IP address, while the same portnumber is defined for the complete system (see RFC 1700). For use in this standard, the portnumber

2404

is defined and has been confirmed by IANA (Internet Assigned Numbers Authority).

Maximum number of outstanding I format APDUs (k)

The value of *k* shall indicate the maximum number of sequentially numbered I format APDUs that the DTE may have outstanding (i.e. unacknowledged) at a given time. Each I frame is sequentially numbered and may have the value 0 through modulus *n* minus 1, where "modulus" is the modulus of the sequence numbers which is defined by the parameter *n*. The value of *k* shall never exceed *n* – 1 for modulo *n* operation (see 2.3.2.2.1 and 2.4.8.6 of the ITU-T X.25 recommendation).

- The transmitter stops the transmission at *k* unacknowledged I format APDUs.
- The receiver acknowledges at the latest after receiving *w* = I format APDUs*
- The maximum number of *k* is *n* – 1 for modulo *n* operation.

Maximum range of values of *k*: 1 to 32767 (2^{15}–1) APDUs, accuracy 1 APDU.

Maximum range of values of *w*: 1 to 32767 APDUs, accuracy 1 APDU (recommendation: *w* should not exceed two-thirds of *k*).

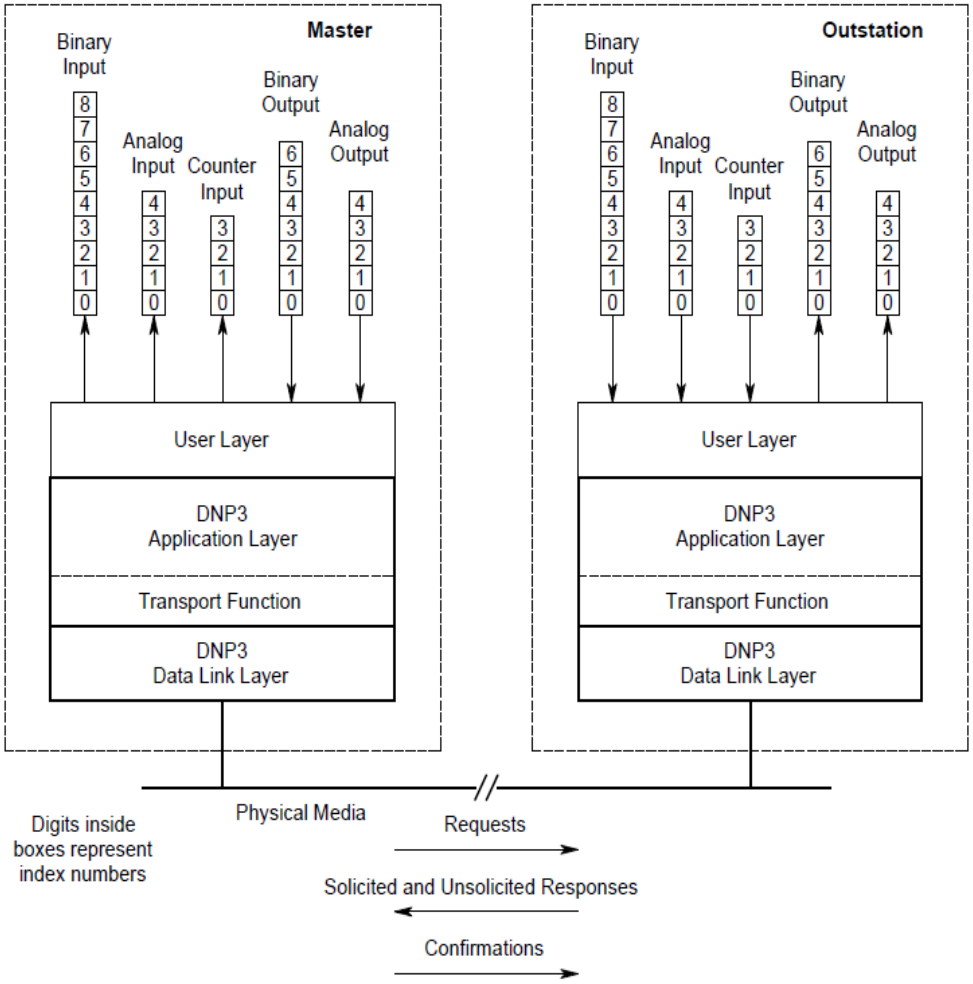
Synchronization Mechanisms

- Control field data of IEC104 contains various types of formats /mechanisms for effective handling of network data synchronization
- 1. I Format – It is used to perform numbered information transfer. It contains send-sequence number and receive-sequence number. The transmitter station increases send-sequence number when it sends any data and receiver increases receive-sequence number when it receives any data. The sending station has to hold the send APDUs in the buffer until it receives back the send sequence numbers as the receive sequence number from destination station.
- 2. S Format – It is used to perform numbered supervisory functions. In any cases where the data transfer is only in a single direction, S-format APDUs has to be send in other direction before timeout (t2), buffer overflow or when it has crossed maximum number of allowed I format APDUs without acknowledgement (w).
- 3. U Format – It is used to perform unnumbered control functions. This is used for activation and confirmation mechanisms of STARTDT (start data transfer) & STOPDT (stop data transfer) & TESTFR (test APDU).
- 4. Test Procedure – Open but unused connections must be tested periodically (when it has crossed 't3' after the last message) by sending TESTFR frames, which need to be acknowledged, by the destination station. The connection needs to be closed when there is no reply for the test message after timeout (t1) or when there are more numbers of I-format APDUs than the specified 'k'.

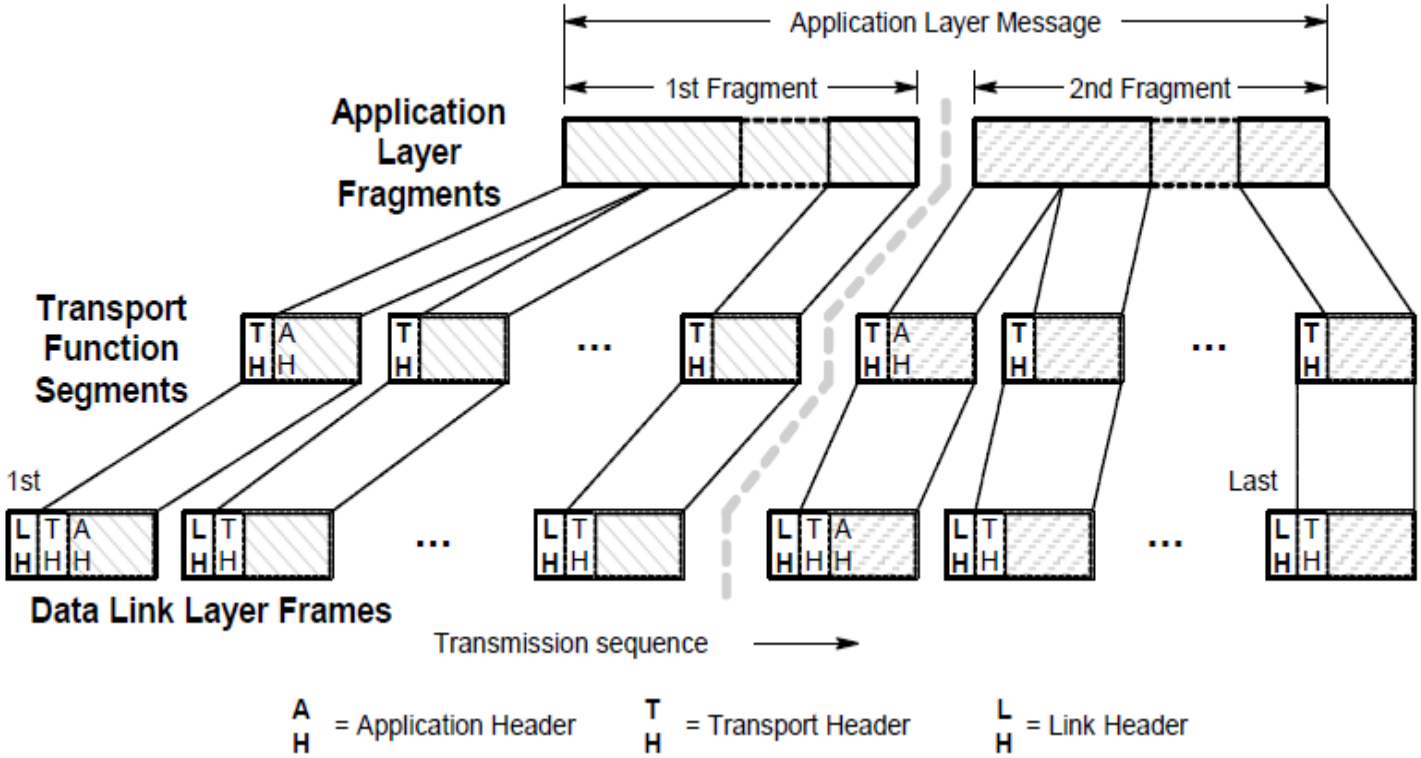
DNP3 Protocol

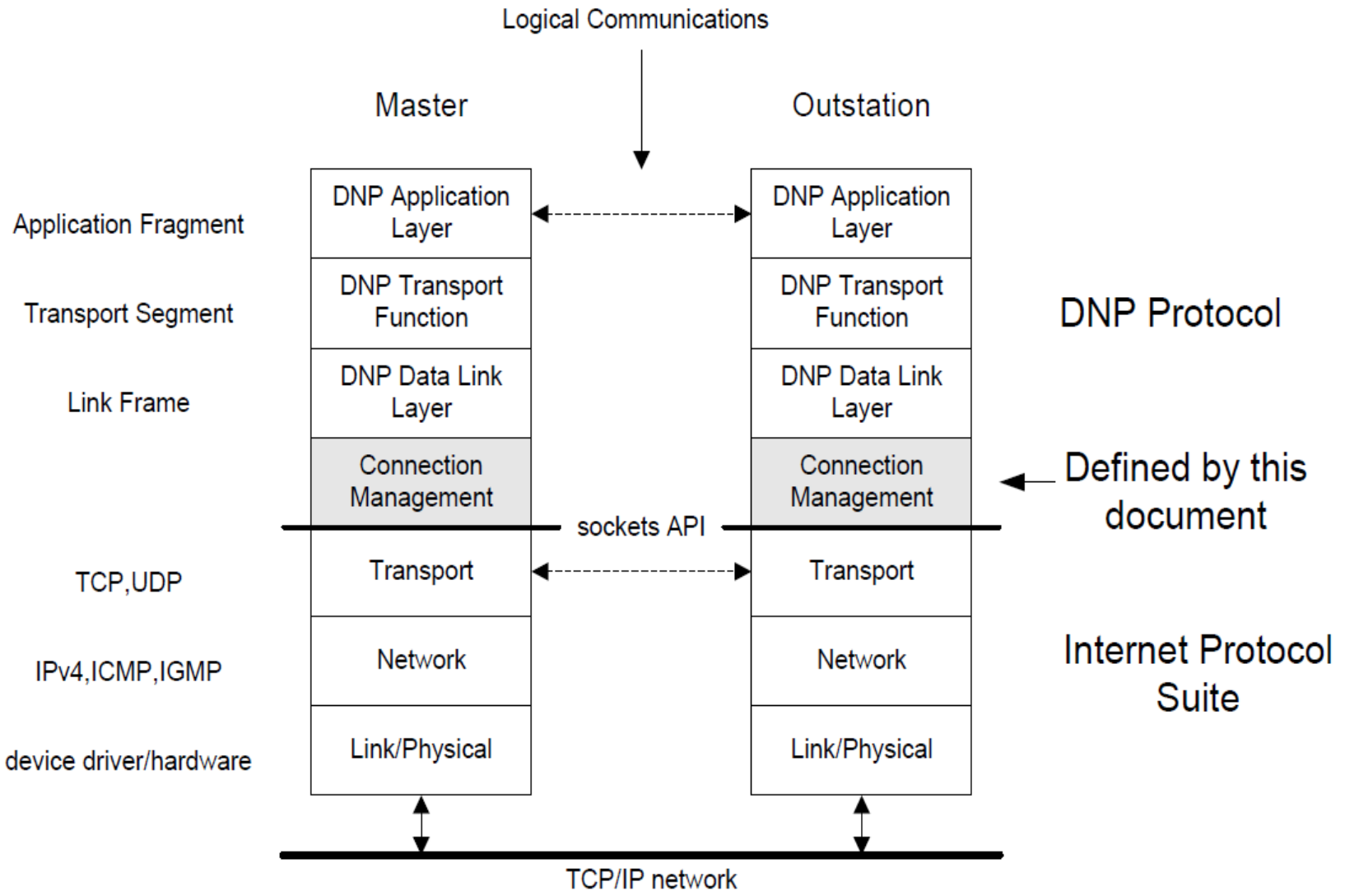
- Westronic Incorporated developed DNP3 between 1992 and 1994
- Time Label for signals at RTU Side
- Master /Slave and Unsolicited Communication Supported - RTU can start Data Communication without master request
- Different type of Data Types
- Event Buffering and data backfilling
- Transferring different type of data in one frame
- Time synchronization
- 4 Layer structure (from 7 layer of OSI)
- Powerful error detecting mechanism
- SBO(Select Before Operate) , Freeze operation

4 Layer structure – Enhanced Performance Architecture - EPA



Fragments , Segments , Frames




Logical Communications
Master
Outstation
Application Fragment
Transport Segment
Link Frame
TCP,UDP
IPv4,ICMP,IGMP
device driver/hardware
DNP Application Layer
DNP Transport Function
DNP Data Link Layer
Connection Management
sockets API
Transport
Network
Link/Physical
DNP Application Layer
DNP Transport Function
DNP Data Link Layer
Connection Management
Transport
Network
Link/Physical
TCP/IP network
DNP Protocol
Defined by this document
Internet Protocol Suite

Data Modeling

- DNP3 Data Modeling is based on Data Type groups , Variations , Address and Class
- Variation is different presentation of a Tag In a group
- Event groups Shows Buffered data with time
- Frozen Counter shows Freeze value of a Counter at specific time
- Example : DI Tag with address 10 (G1) and G2 tag with Address 10 refer to same point . G1 shows Current value of Tag and g2 Shows Buffered data

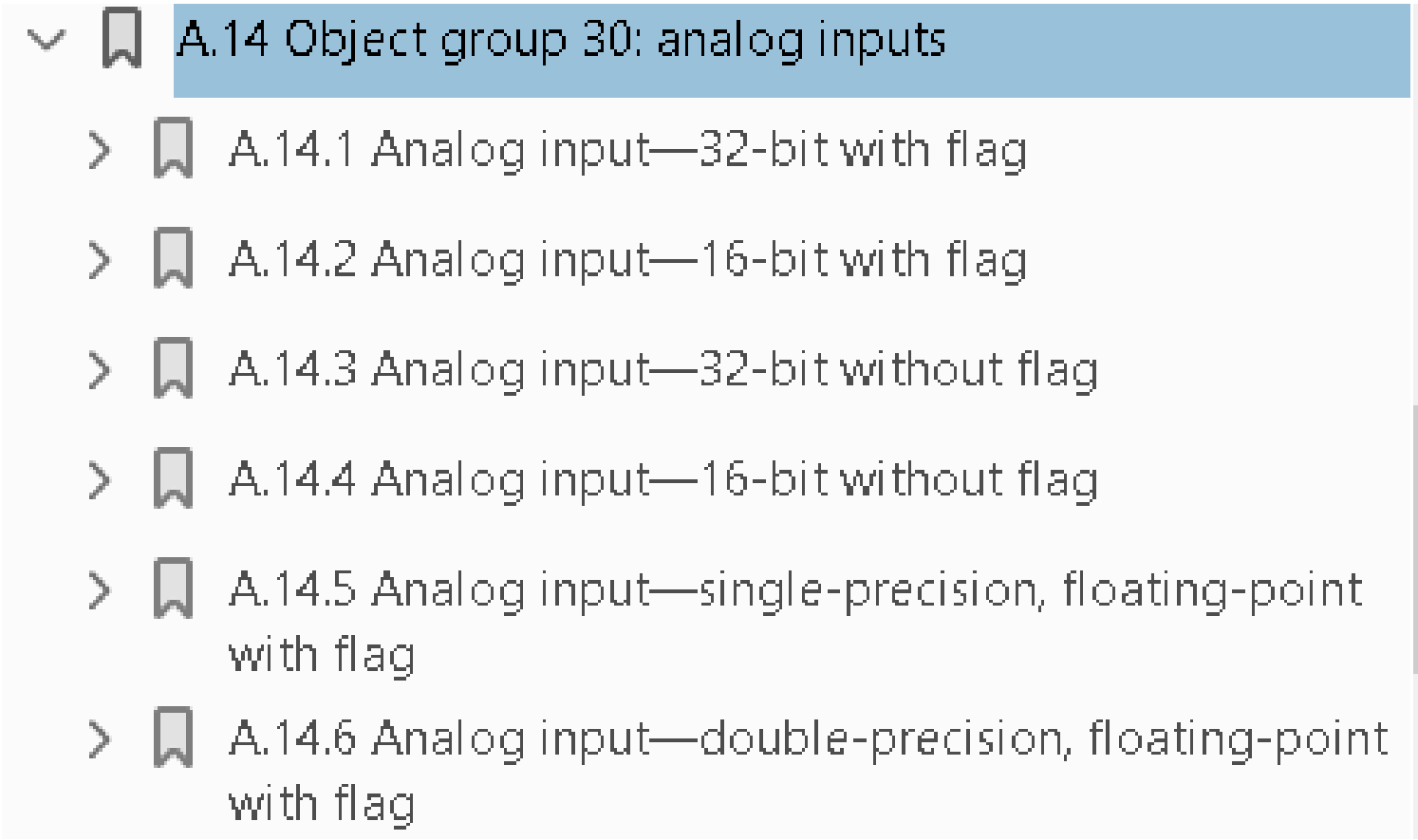
Most Important groups

- Group1 = Digital input
- Group2 = Digital Input Event
- Group3 = Double Bit Input
- Group4 = Double Bit Input Event
- Group 10 = Digital output Status
- Group 11 = Digital output Status Event
- Group 12 = Digital Output Command
- Group 20 = Counters
- Group 21 = Frozen Counters
- Group 22 = Counters Event
- Group 23 = Frozen Counters Event
- Group 30 = Analog Input
- Group 32 = Analog Input Event
- Group 40 = Analog Output Status
- Group 41 = Analog Output Command
- Group 42 = Analog output Status Event
- Group 50 = Date Time
- Group 51 = Common Time of occurrence
- Group 60 = Class Object
- Group 80 = Internal Indication
-

Variations

- Variation for a Group is different presentation for data
- Master can read with any variation from Slave
- When Master is asking for Class 0 , 1,2,3 then Slave should send by its default variation
- When Slave is sending data by unsolicited communication , then Slave should use its default variations
- Default variations can be set in Slave device as parameters

Group 30 Variations



GROUP 60 CLASS

- DNP Tags has Class .
  - Class 0 = Static value . Current value of Tag
  - Class 1 = Event Value . Class 1
  - Class 2 = Event Value . Class 2
  - Class 3 = Event Value . Class 3

- There is no priority for class 1 , 2, 3 . It is only a logical grouping of tags
- For example you can set All Digital Input tags in class 1 and all Analog Inputs in class 2 . Then master can send class 1 request every sec and class 2 request every 10 sec .

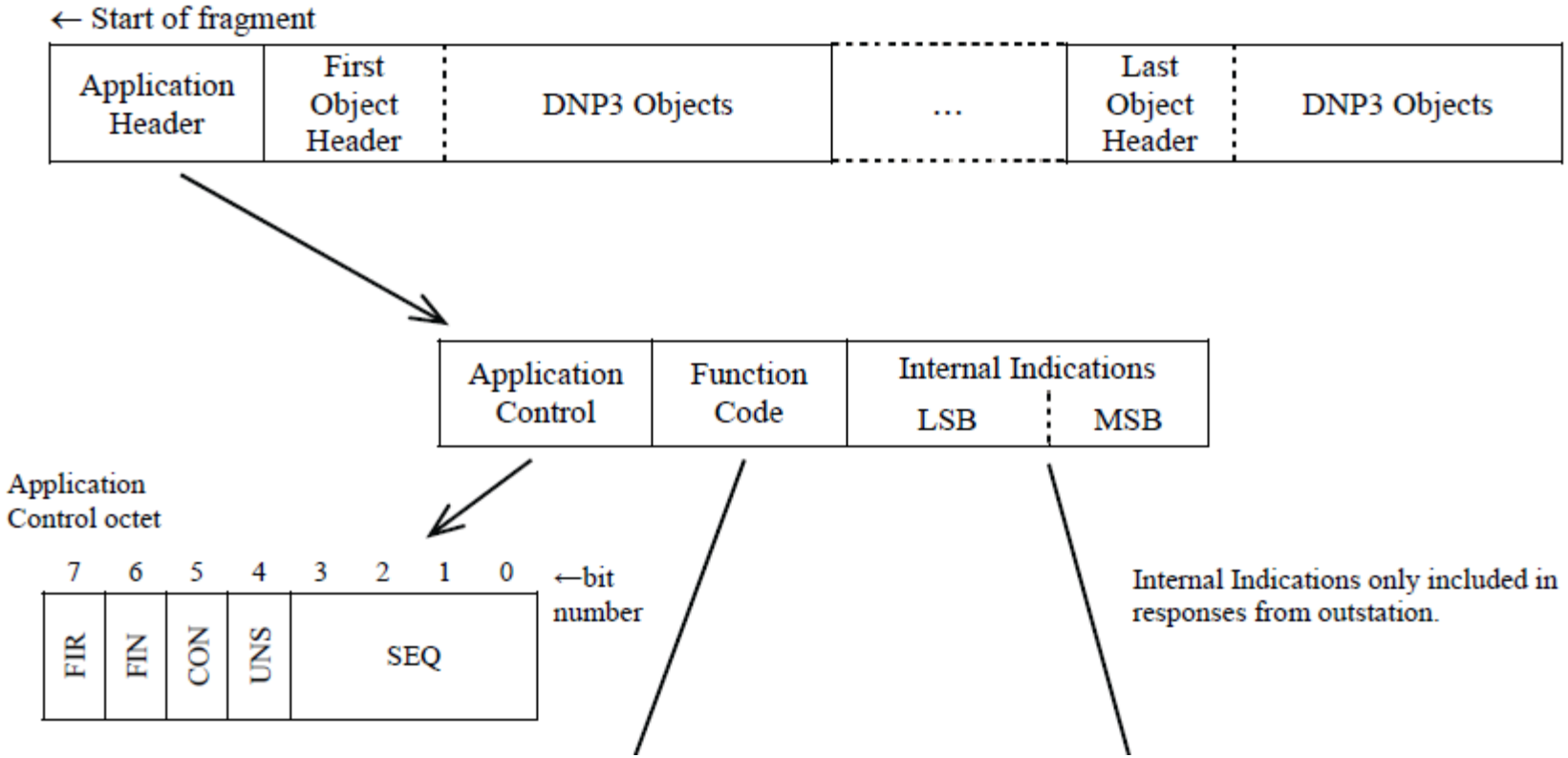
1	1	Binary Input – Packed format	0.02
1	2	Binary Input – With flags	0.02
2	1	Binary Input Event – Without time	0.02
2	2	Binary Input Event – With absolute time	0.02
2	3	Binary Input Event – With relative time	0.02
3	1	Double-bit Binary Input – Packed format	1.00
3	2	Double-bit Binary Input – With flags	1.00
4	1	Double-bit Binary Input Event – Without time	1.00
4	2	Double-bit Binary Input Event – With absolute time	1.00
4	3	Double-bit Binary Input Event – With relative time	1.00
10	1	Binary Output – Packed format	0.02
10	2	Binary Output – Output status with flags	0.02
11	1	Binary Output Event – Status without time	1.01
11	2	Binary Output Event – Status with time	1.00
12	1	Binary Command – Control relay output block (CROB)	0.02
12	2	Binary Command – Pattern control block (PCB)	0.02
12	3	Binary Command – Pattern mask	0.02
13	1	Binary Output Command Event – Command status without time	1.00
13	2	Binary Output Command Event – Command status with time	1.00
20	1	Counter – 32-bit with flag	0.02
20	2	Counter – 16-bit with flag	0.02
20	3	Counter – 32-bit with flag, delta	0.02
20	4	Counter – 16-bit with flag, delta	0.02
20	5	Counter – 32-bit without flag	0.02
20	6	Counter – 16-bit without flag	0.02
20	7	Counter – 32-bit without flag, delta	0.02
20	8	Counter – 16-bit without flag, delta	0.02
21	1	Frozen Counter – 32-bit with flag	0.02
21	2	Frozen Counter – 16 bit with flag	0.02
21	3	Frozen Counter – 32-bit with flag, delta	0.02
21	4	Frozen Counter – 16-bit with flag, delta	0.02

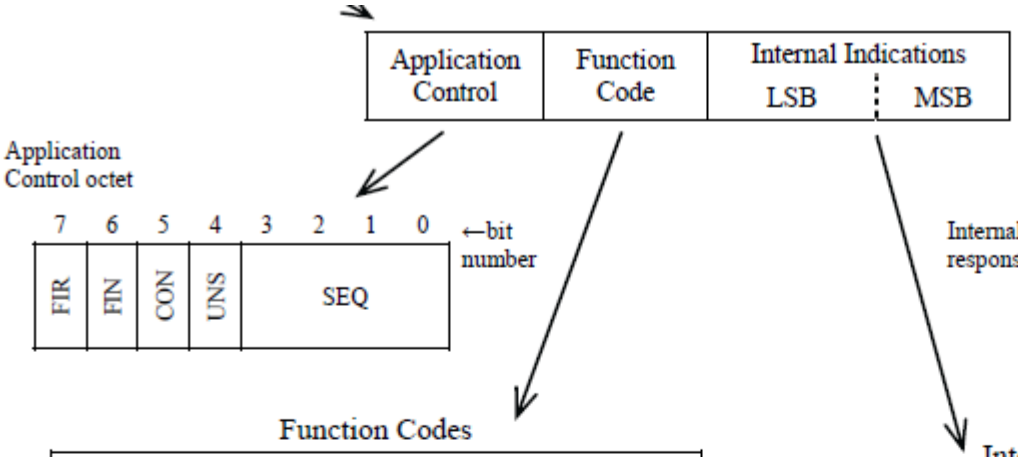
20	5	Counter – 32-bit without flag	0.02
20	6	Counter – 16-bit without flag	0.02
20	7	Counter – 32-bit without flag, delta	0.02
20	8	Counter – 16-bit without flag, delta	0.02
21	1	Frozen Counter – 32-bit with flag	0.02
21	2	Frozen Counter – 16 bit with flag	0.02
21	3	Frozen Counter – 32-bit with flag, delta	0.02
21	4	Frozen Counter – 16-bit with flag, delta	0.02
21	5	Frozen Counter – 32-bit with flag and time	0.02
21	6	Frozen Counter – 16-bit with flag and time	0.02
21	7	Frozen Counter – 32-bit with flag and time, delta	0.02
21	8	Frozen Counter – 16-bit with flag and time, delta	0.02
21	9	Frozen Counter – 32-bit without flag	0.02
21	10	Frozen Counter – 16-bit without flag	0.02
21	11	Frozen Counter – 32-bit without flag, delta	0.02
21	12	Frozen Counter – 16-bit without flag, delta	0.02
22	1	Counter Event – 32-bit with flag	0.02
22	2	Counter Event – 16-bit with flag	0.02
22	3	Counter Event – 32-bit with flag, delta	0.02
22	4	Counter Event – 16-bit with flag, delta	0.02
22	5	Counter Event – 32-bit with flag and time	0.02
22	6	Counter Event – 16-bit with flag and time	0.02
22	7	Counter Event – 32-bit with flag and time, delta	0.02
22	8	Counter Event – 16-bit with flag and time, delta	0.02
23	1	Frozen Counter Event – 32-bit with flag	0.02
23	2	Frozen Counter Event – 16-bit with flag	0.02
23	3	Frozen Counter Event – 32-bit with flag, delta	0.02
23	4	Frozen Counter Event – 16-bit with flag, delta	0.02

30	1	Analog Input – 32-bit with flag	0.02
30	2	Analog Input – 16-bit with flag	0.02
30	3	Analog Input – 32-bit without flag	0.02
30	4	Analog Input – 16-bit without flag	0.02
30	5	Analog Input – Single-prec flt-pt with flag	0.02
30	6	Analog Input – Double-prec flt-pt with flag	0.02
31	1	Frozen Analog Input – 32-bit with flag	0.02
31	2	Frozen Analog Input – 16-bit with flag	0.02
31	3	Frozen Analog Input – 32-bit with time-of-freeze	0.02
31	4	Frozen Analog Input – 16-bit with time-of-freeze	0.02
31	5	Frozen Analog Input – 32-bit without flag	0.02
31	6	Frozen Analog Input – 16-bit without flag	0.02
31	7	Frozen Analog Input – Single-prec flt-pt with flag	0.02
31	8	Frozen Analog Input – Double-prec flt-pt with flag	0.02
32	1	Analog Input Event – 32-bit without time	0.02
32	2	Analog Input Event – 16-bit without time	0.02
32	3	Analog Input Event – 32-bit with time	0.02
32	4	Analog Input Event – 16-bit with time	0.02
32	5	Analog Input Event – Single-prec flt-pt without time	0.02
32	6	Analog Input Event – Double-prec flt-pt without time	0.02
32	7	Analog Input Event – Single-prec flt-pt with time	0.02
32	8	Analog Input Event – Double-prec flt-pt with time	0.02
33	1	Frozen Analog Input Event – 32-bit without time	0.02
33	2	Frozen Analog Input Event – 16-bit without time	0.02
33	3	Frozen Analog Input Event – 32-bit with time	0.02
33	4	Frozen Analog Input Event – 16-bit with time	0.02
33	5	Frozen Analog Input Event – Single-prec flt-pt without time	0.02
33	6	Frozen Analog Input Event – Double-prec flt-pt without time	0.02
33	7	Frozen Analog Input Event – Single-prec flt-pt with time	0.02
33	8	Frozen Analog Input Event – Double-prec flt-pt with time	0.02
34	1	Analog Input Deadband – 16-bit	0.01
34	2	Analog Input Deadband – 32-bit	0.01
34	3	Analog Input Deadband – Single-prec flt-pt	0.01

43	1	Analog Output Command Event – 32-bit without time	1.00
43	2	Analog Output Command Event – 16-bit without time	1.00
43	3	Analog Output Command Event – 32-bit with time	1.00
43	4	Analog Output Command Event – 16-bit with time	1.00
43	5	Analog Output Command Event – Single-prec flt-pt without time	1.00
43	6	Analog Output Command Event – Double-prec flt-pt without time	1.00
43	7	Analog Output Command Event – Single-prec flt-pt with time	1.00
43	8	Analog Output Command Event – Double-prec flt-pt with time	1.00
50	1	Time and Date – Absolute time	0.02
50	2	Time and Date – Absolute time and interval	0.02
50	3	Time and Date – Absolute time at last recorded time	1.00
51	1	Time and Date CTO – Absolute time, synchronized	0.02
51	2	Time and Date CTO – Absolute time, unsynchronized	0.02
52	1	Time Delay – Coarse	0.02
52	2	Time Delay – Fine	0.02
60	1	Class Objects – Class 0 data	1.00
60	2	Class Objects – Class 1 data	1.00
60	3	Class Objects – Class 2 data	1.00
60	4	Class Objects – Class 3 data	1.00

Application Layer





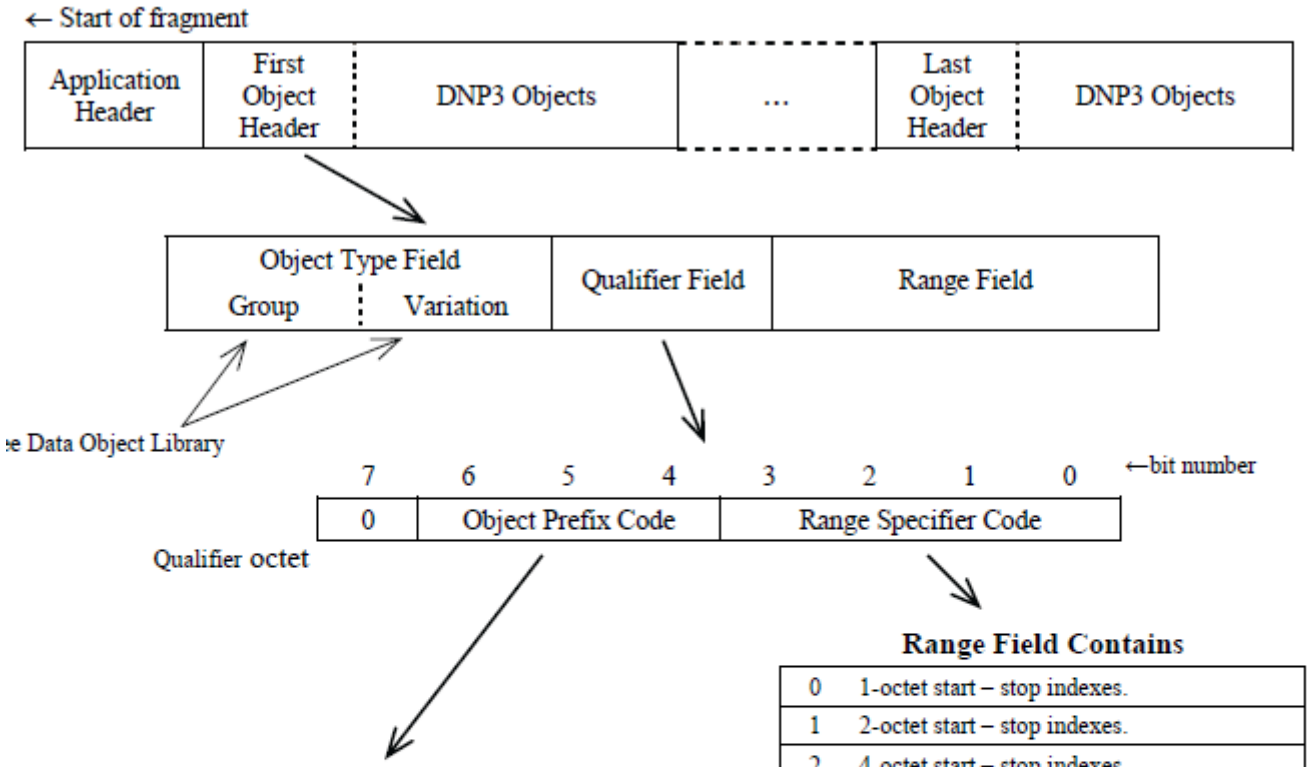
Function Codes

Requests (Hex)			
0	Confirm	11	Start application
1	Read	12	Stop application
2	Write	13	Save configuration
3	Select	14	Enable unsolicited
4	Operate	15	Disable unsolicited
5	Dir operate	16	Assign class
6	Dir operate – No resp	17	Delay measurement
7	Freeze	18	Record current time
8	Freeze – No resp	19	Open file
9	Freeze clear	1A	Close file
A	Freeze clear – No resp	1B	Delete file
B	Freeze at time	1C	Get file information
C	Freeze at time – No resp	1D	Authenticate file
D	Cold restart	1E	Abort file
E	Warm restart	1F	Activate config
F	Initialize data	20	Authentication request
10	Initialize application	21	Authentication error
Responses (Hex)			
81	Response	83	Authentication response
82	Unsolicited response		

Internal Indications

LSB	
IIN1.0	All stations
IIN1.1	Class 1 events
IIN1.2	Class 2 events
IIN1.3	Class 3 events
IIN1.4	Need time
IIN1.5	Local control
IIN1.6	Device trouble
IIN1.7	Device restart
MSB	
IIN2.0	Function code not supported
IIN2.1	Object unknown
IIN2.2	Parameter error
IIN2.3	Event buffer overflow
IIN2.4	Already executing
IIN2.5	Configuration corrupt
IIN2.6	Reserved
IIN2.7	Reserved

Application Layer



Object Prefix

0	Objs packed without a prefix.
1	Objs prefixed with 1-octet index.
2	Objs prefixed with 2-octet index.
3	Objs prefixed with 4-octet index.
4	Objs prefixed with 1-octet object size.
5	Objs prefixed with 2-octet object size.
6	Objs prefixed with 4-octet object size.
7	Reserved.

Range Field Contains

0	1-octet start – stop indexes.
1	2-octet start – stop indexes.
2	4-octet start – stop indexes.
3	1-octet start – stop virtual addresses.
4	2-octet start – stop virtual addresses.
5	4-octet start – stop virtual addresses.
6	No range field used. Implies all values.
7	1-octet count of objects.
8	2-octet count of objects.
9	4-octet count of objects.
A	Reserved.
B	1-octet count of objects (variable format).
C	Reserved.
D	Reserved.
E	Reserved.
F	Reserved.

EX 4-9	This example shows a request for the static analog values from indexes 4 through 7 returned as a 16-bit value with a flag octet.

►►► Request Message

C3	01	1E	02	00	04	07
AC	FC	Grp	Var	Qual	Range	

◄◄◄ Response Message (beginning)

C3	81	00	00	1E	02	00	04	07	01	88
AC	FC	IIN_1	IIN_2	Grp	Var	Qual	Range		Flg_4 ←	LSB_4 DIO_4

◄◄◄ Continuation of Response Message

13	01	20	4E	01	50	FB	01	60	00
MSB_4 →	Flg_5 ←	LSB_5 DIO_5	MSB_5 →	Flg_6 ←	LSB_6 DIO_6	MSB_6 →	Flg_7 ←	LSB_7 DIO_7	MSB_7 →

EX 4-10	This example shows a request for all of the static binary inputs. Assume there are 18 binary inputs.

►►► Request Message

C3	01	01	01	06
AC	FC	Grp	Var	Qual

Note that no range field appears when the qualifier 06 is specified; it means values from all points.

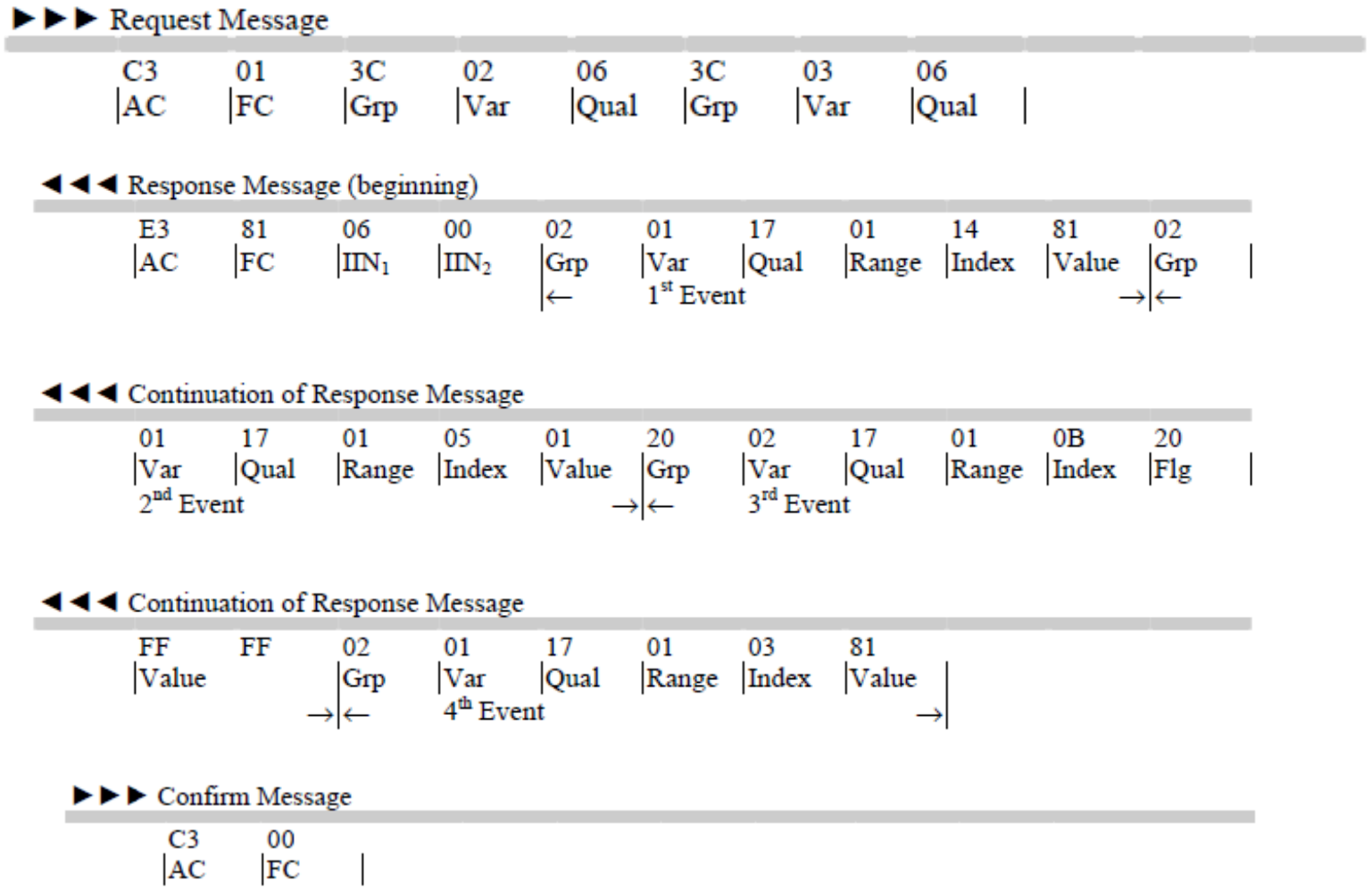
◄◄◄ Response Message (beginning)

C3	81	00	00	01	01	01	00	11	0F	AA
AC	FC	IIN_1	IIN_2	Grp	Var	Qual	Range		←	States

◄◄◄ Continuation of Response Message

03
→

EX 4-11	This example illustrates a request to read Class 1 and Class 2 event data. Event data is requested with object group 60.



EX 4-12	This example shows the master requesting a maximum of 20 binary events. When a master requests event data using a group number other than 60 (class data), as this example does, it expects to receive events from the respective point type without regard to the event class assignments for any of those points.

►►► Request Message

C3	01	02	00	07	14
AC	FC	Grp	Var	Qual	Range

The request uses variation 0. Variation 0 has special meaning, and only master requests may use it. Variation 0 indicates that the master does not have a preferred format for the outstation to use in its response—the outstation determines which object variations to return.

◄◄◄ Response Message (beginning)

E3	81	06	00	02	01	17	03	14	81	05
AC	FC	IIN_1	IIN_2	Grp	Var	Qual	Range	Index 1st Event	Value	Index 2nd

◄◄◄ Continuation of Response Message

01	03	81
Value Event	Index 3rd Event	Value

►►► Confirm Message

C3	00
AC	FC

EX 4-13	This example illustrates returning what is called a "null response" because no data objects are included. Outstations send null responses when the master requests events and no events qualify for reporting back in a response.

►►► Request Message

C3	01	02	00	07	14
AC	FC	Grp	Var	Qual	Range

This request is identical to the previous example's request.

◄◄◄ Response Message

C3	81	04	00
AC	FC	IIN_1	IIN_2

EX 4-14	This example shows a master clearing an outstation's IIN1.7 [DEVICE_RESTART] bit.

►►► Request Message

C3	02	50	01	00	07	07	00
AC	FC	Grp	Var	Qual	Start Range	Stop	Value

◄◄◄ Response Message

C3	81	00	00
AC	FC	IIN_1	IIN_2

EX 4-15	This example shows setting analog deadbands for indexes 6, 8, and 20.

►►► Request Message

C3	02	22	01	17	03	06	12	00	08	4A
AC	FC	Grp	Var	Qual	Range	Index	Value		Index	Value
						← DIO_6			→ ← DIO_8	

►►► Continuation of Request Message

00	14	FF	FF
Value	Index	Value	
→	← DIO_{20}		→

◄◄◄ Response Message

C3	81	00	00
AC	FC	IIN_1	IIN_2

EX 4-16	This example shows setting the time.

►►► Request Message

C3	02	32	01	07	01	AC	E9	00	40	08
AC	FC	Grp	Var	Qual	Range	Time				

►►► Continuation of Request Message

01
Time

◄◄◄ Response Message

C3	81	00	00
AC	FC	IIN_1	IIN_2

EX 4-17	This example shows *select–operate* messages for a CROB (g12v1) issued to point 10. The command is to close the point one time for 250 milliseconds.

In the first exchange, the master sends a request with the SELECT function code.

►►► **Select** Request Message (beginning)

C3	03	0C	01	17	01	0A	41	01	FA	00
AC	FC	Grp	Var	Qual	Range	Prefix	←	CROB		

►►► Continuation of **Select** Request Message

00	00	00	00	00	00	00
CROB continued						→

◄◄◄ **Select** Response Message (beginning)

C3	81	00	00	0C	01	17	01	0A	41	01
AC	FC	IIN_1	IIN_2	Grp	Var	Qual	Range	Prefix	←	

◄◄◄ Continuation of **Select** Response Message

FA	00	00	00	00	00	00	00	00
CROB								→

►►► **Operate** Request Message (beginning)

C4	04	0C	01	17	01	0A	41	01	FA	00
|AC	|FC	|Grp	|Var	|Qual	|Range	|Prefix	|←	CROB		

►►► Continuation of **Operate** Request Message

00	00	00	00	00	00	00	
CROB continued						→|	

◄◄◄ **Operate** Response Message (beginning)

C4	81	00	00	0C	01	17	01	0A	41	01
|AC	|FC	|IIN_1	|IIN_2	|Grp	|Var	|Qual	|Range	|Prefix	|←	

◄◄◄ Continuation of **Operate** Response Message

FA	00	00	00	00	00	00	00	00	
CROB								→|	

EX 4-20	In this example, a master requests the outstation to freeze all of its counters.

►►► Request Message

C3	07	14	00	06	
AC	FC	Grp	Var	Qual	

◄◄◄ Response Message

C3	81	00	00	
AC	FC	IIN_1	IIN_2	

EX 4-21	This is an example of freezing all counters beginning on 15 June 2001 at 14 hours, 2 minutes, 0 seconds, and 0 milliseconds, and every 15 minutes thereafter.

►►► Request Message (beginning)

C3	0B	32	02	07	01	C0	5F	63	1C	E7
|AC	|FC	|Grp	|Var	|Qual	|Range	|←	Time & Date w/ Interval Object			

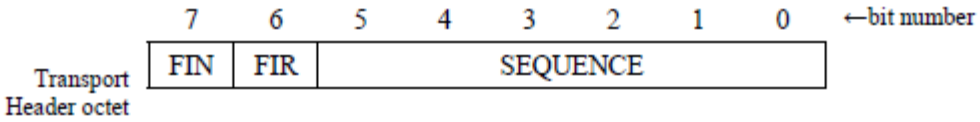
►►► Continuation of Request Message

00	A0	BB	0D	00	14	00	06
T & D w/ Intvl Obj Continued					→|Grp	|Var	|Qual |

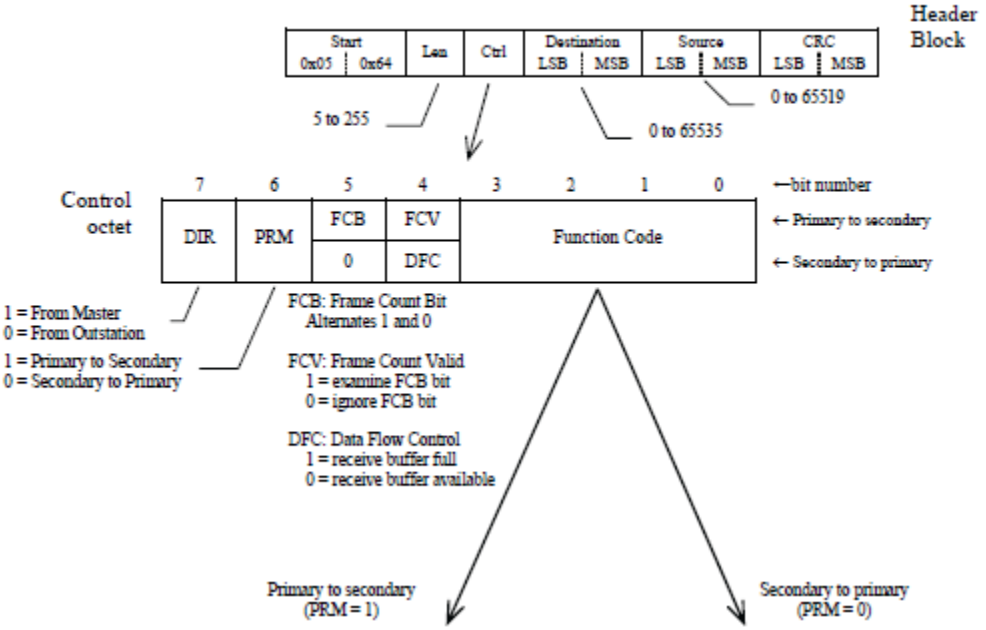
◄◄◄ Response Message

C3	81	00	00
|AC	|FC	|IIN_1	|IIN_2 |

Transport Function



Data Link Layer



Primary Function Code	Function Code Name	FCV Bit
0	RESET_LINK_STATES	0
1	–	–
2	TEST_LINK_STATES	1
3	CONFIRMED_USER_DATA	1
4	UNCONFIRMED_USER_DATA	0
5	–	–
6	–	–
7	–	–
8	–	–
9	REQUEST_LINK_STATUS	0
A	–	–
B	–	–
C	–	–
D	–	–
E	–	–
F	–	–

Secondary Function Code	Function Code Name
0	ACK
1	NACK
2	–
3	–
4	–
5	–
6	–
7	–
8	–
9	–
A	–
B	LINK_STATUS
C	–
D	–
E	–
F	NOT_SUPPORTED

Transaction Diagram

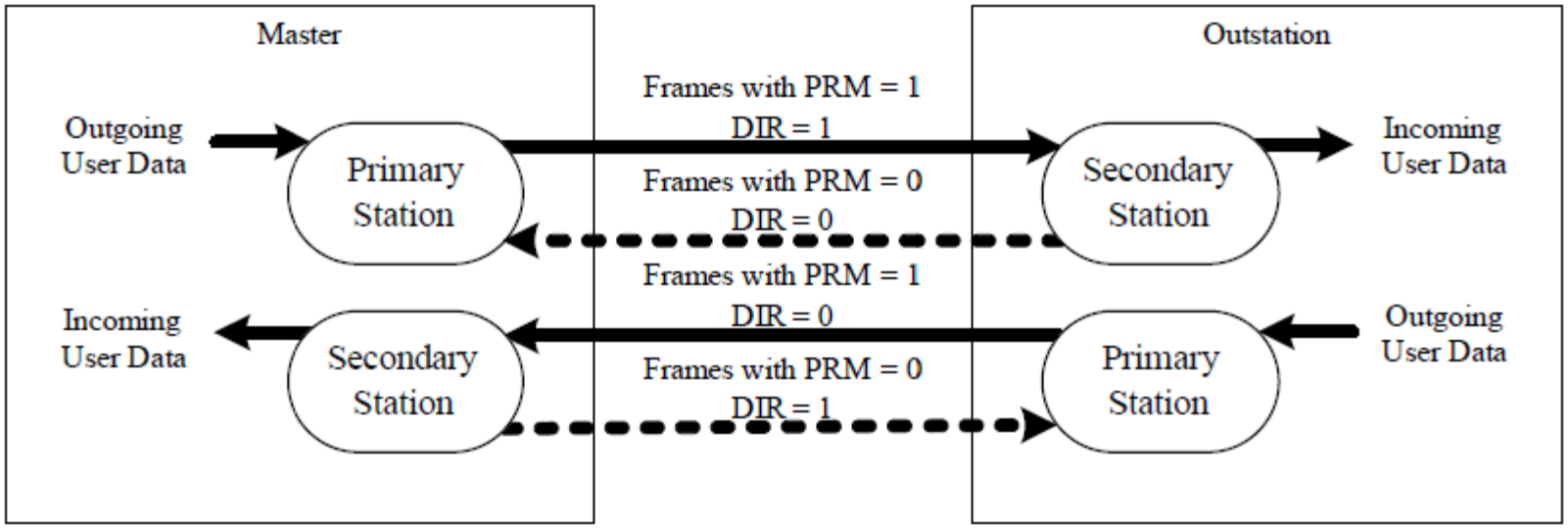


Figure 9-2—Transaction diagram

DLL Frame format

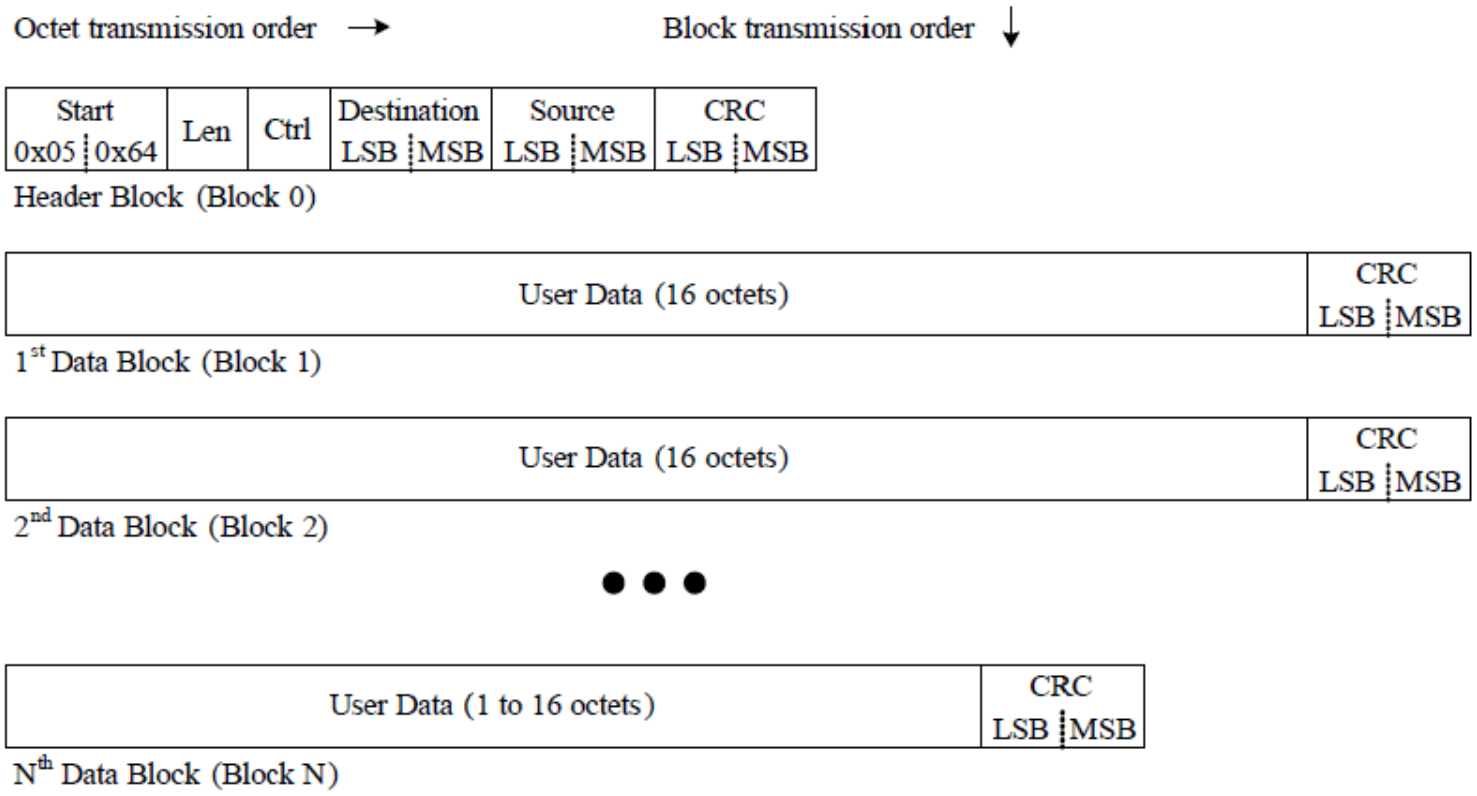


Figure 9-3—DNP3 frame format

Valid Data Link Layer Control Codes

Outstation to Master	Master to Outstation	Function Code Name	Type	Comment
00	**80**	**ACK**	Sec-to-Pri	
01	**81**	**NACK**		Link reset required
0B	8B	LINK_STATUS		
0F	8F	NOT_SUPPORTED		
10	90	ACK		Receive buffers full
11	91	NACK		Receive buffers full
1B	9B	LINK_STATUS		Receive buffers full
1F	9F	NOT_SUPPORTED		Receive buffers full
40	**C0**	**RESET_LINK_STATES**	Pri-to-Sec	FCB = 0 (secondary ignores FCB)
44	**C4**	**UNCONFIRMED_USER_DATA**		FCB = 0 (secondary ignores FCB)
49	C9	REQUEST_LINK_STATUS		FCB = 0 (secondary ignores FCB)
52	D2	TEST_LINK_STATES		FCB = 0
53	**D3**	**CONFIRMED_USER_DATA**		FCB = 0
60	E0	RESET_LINK_STATES		FCB = 1 (secondary ignores FCB)
64	E4	UNCONFIRMED_USER_DATA		FCB = 1 (secondary ignores FCB)
69	E9	REQUEST_LINK_STATUS		FCB = 1 (secondary ignores FCB)
72	F2	TEST_LINK_STATES		FCB = 1
73	**F3**	**CONFIRMED_USER_DATA**		FCB = 1

Table 9-1—Primary-to-secondary (PRM = 1) function codes

Primary function code	Function code name	Service function	FCV bit	Response function codes permitted from Secondary Station
0	RESET_LINK_STATES	Reset of remote link	0	0 or 1
1	—	Obsolete	—	15 or no response
2	TEST_LINK_STATES	Test function for link	1	0 or 1 (no response is acceptable if the link states are UnReset)
3	CONFIRMED_USER_DATA	Deliver application data, confirmation requested	1	0 or 1
4	UNCONFIRMED_USER_DATA	Deliver application data, confirmation not requested	0	No Secondary Station Data Link response
5	—	Reserved	—	15 or no response
6	—	Reserved	—	15 or no response
7	—	Reserved	—	15 or no response
8	—	Reserved	—	15 or no response
9	REQUEST_LINK_STATUS	Request status of link	0	11
10	—	Reserved	—	15 or no response
11	—	Reserved	—	15 or no response
12	—	Reserved	—	15 or no response
13	—	Reserved	—	15 or no response
14	—	Reserved	—	15 or no response
15	—	Reserved	—	15 or no response

Table 9-2—Secondary-to-primary (PRM = 0) function codes

Secondary function code	Function code name	Service function
0	ACK	Positive acknowledgment
1	NACK	Negative acknowledgment
2	—	Reserved
3	—	Reserved
4	—	Reserved
5	—	Reserved
6	—	Reserved
7	—	Reserved
8	—	Reserved
9	—	Reserved
10	—	Reserved
11	LINK_STATUS	Status of the link
12	—	Reserved
13	—	Reserved
14	—	Obsolete
15	NOT_SUPPORTED	Link service not supported

Table 9-3—Special use addresses

Address	Special use
0xFFFF	Broadcast, Application Layer Confirmation to clear IIN1.0 [BROADCAST] is optional
0xFFFE	Broadcast, Application Layer Confirmation to clear IIN1.0 [BROADCAST] is mandatory
0xFFFD	Broadcast, Application Layer Confirmation shall not be required to clear IIN1.0 [BROADCAST]
0xFFFC	Self-address
0xFFF0 to 0xFFFB	Reserved

Table 9-4—Primary Station variables

Variable	Description
SecondaryStationIsReset	This is a boolean variable that indicates that the Primary Station sent a RESET_LINK_STATES frame to the Secondary Station and received an ACK confirmation back.
SoftwareOperatingState	This variable indicates the current state in which the software is operating. The states are shown in column A of Table 9-6.
NFCB (Next FCB)	The 1 or 0 state of the next FCB bit to be included when transmitting a new primary-to-secondary frame with the FCV bit set.

Table 9-5—Secondary Station variables

Variable	Description
LinkIsReset	This is a boolean variable that indicates whether the link was reset via receipt of a RESET_LINK_STATES frame from the Primary Station.
SoftwareOperatingState	This variable indicates the current state in which the software is operating. The states are shown in column A of Table 9-6.
EFCB (Expected FCB)	The state of the FCB bit expected in the next frame from the Primary Station with the FCV bit set.

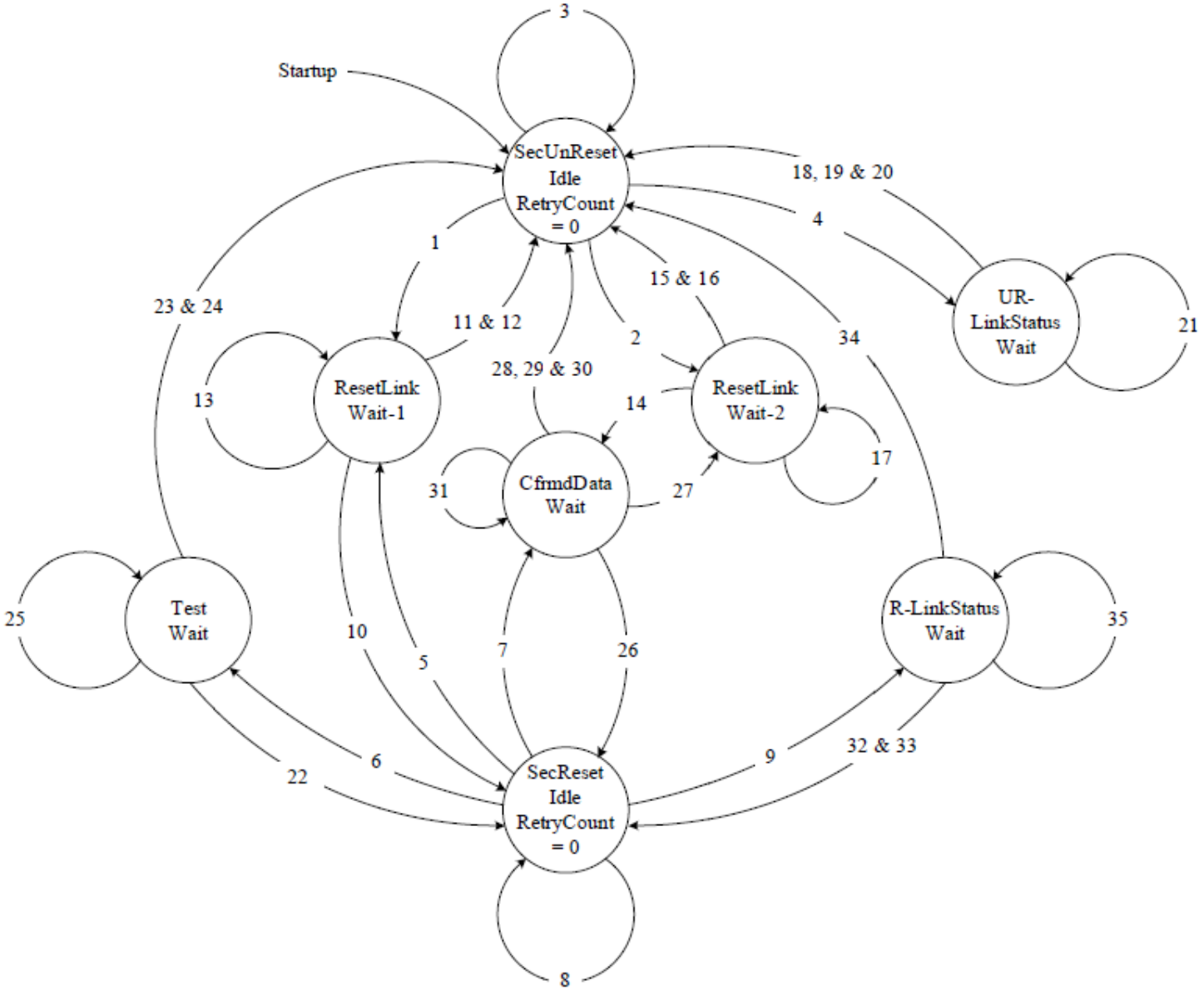


Figure 9-8—Primary Station state diagram

Current state	Event that triggers an action and possible transition	Action			Transition to state	
A	B	C	D		E	
If the software state is	Cause for action	Perform this action[a]	and transmit this function code (with user data if appropriate)		and then go to this state	
SecUnResetIdle	Implementation dependent	Set retry count = 0	0	RESET_LINK_STATES	ResetLinkWait-1	1
	User request to send confirmed data	Set retry count = 0	0	RESET_LINK_STATES	ResetLinkWait-2	2
	User request to send unconfirmed data	—	4	UNCONFIRMED_USER_DATA	SecUnResetIdle	3
	Keep-alive or implementation dep.	Set retry count = 0	9	REQUEST_LINK_STATUS	UR–LinkStatusWait	4
SecResetIdle	Implementation dependent	Set retry count = 0	0	RESET_LINK_STATES	ResetLinkWait-1	5
	User request to test link	Set retry count = 0	2	TEST_LINK_STATES	TestWait	6
	User request to send confirmed data	Set retry count = 0	3	CONFIRMED_USER_DATA	CfmdDataWait	7
	User request to send unconfirmed data	—	4	UNCONFIRMED_USER_DATA	SecResetIdle	8
	Keep-alive or implementation dep.	Set retry count = 0	9	REQUEST_LINK_STATUS	R–LinkStatusWait	9
ResetLinkWait-1	Function code 0 (ACK) received	▲ Set NFCB = 1.		—	SecResetIdle	10
	Non-0 function code received	▼		—	SecUnResetIdle	11
	Timeout, no response, retry count exceeded	▼ Act on failure (implementation dependent)		—	SecUnResetIdle	12
	Timeout, no response, retries remaining	Increment retry count	0	RESET_LINK_STATES	ResetLinkWait-1	13
ResetLinkWait-2	Function code 0 (ACK) received	▲ Set NFCB = 1.	3	CONFIRMED_USER_DATA	CfmdDataWait	14
	Non-0 function code received	▼		—	SecUnResetIdle	15
	Timeout, no response, retry count exceeded	▼ Act on failure (implementation dependent)		—	SecUnResetIdle	16
	Timeout, no response, retries remaining	Increment retry count	0	RESET_LINK_STATES	ResetLinkWait-2	17
UR–LinkStatusWait	Function code 11 (LINK_STATUS) received	Implementation dependent		—	SecUnResetIdle	18
	Non-11 function code received	Note error (implementation dependent)		—	SecUnResetIdle	19

Current state	Event that triggers an action and possible transition	Action			Transition to state	
A	B	C	D		E	
If the software state is	Cause for action	Perform this action[a]	and transmit this function code (with user data if appropriate)		and then go to this state	
	Timeout, no response, retry count exceeded	Act on failure (implementation dependent)	—		SecUnResetIdle	20
	Timeout, no response, retries remaining	Increment retry count	9	REQUEST_LINK_STATUS	UR–LinkStatusWait	21
TestWait	Function code 0 (ACK) received	Toggle NFCB.	—		SecResetIdle	22
	Non-0 function code received	▼	—		SecUnResetIdle	23
	Timeout, no response, retry count exceeded	▼ Act on failure (implementation dependent)	—		SecUnResetIdle	24
	Timeout, no response, retries remaining	Increment retry count	2	TEST_LINK_STATES	TestWait	25
CfmdDataWait	Function code 0 (ACK) received	Toggle NFCB.	—		SecResetIdle	26
	Function code 1 (NACK) received, DFC = 0	▼ Set retry count = 0.	0	RESET_LINK_STATES	ResetLinkWait-2	27
	Function code 1 (NACK) received, DFC = 1	▼	—		SecUnResetIdle	28
	Non-0, non-1 function code received	▼	—		SecUnResetIdle	29
	Timeout, no response, retry count exceeded	▼ Act on failure (implementation dependent)	—		SecUnResetIdle	30
	Timeout, no response, retries remaining	Increment retry count	3	CONFIRMED_USER_DATA	CfmdDataWait	31
R–LinkStatusWait	Function code 11 (LINK_STATUS) received	Implementation dependent	—		SecResetIdle	32
	Non-11 function code received	Note error (implementation dependent)	—		SecResetIdle	33
	Timeout, no response, retry count exceeded	▼ Act on failure (implementation dependent)	—		SecUnResetIdle	34
	Timeout, no response, retries remaining	Increment retry count	9	REQUEST_LINK_STATUS	R–LinkStatusWait	35

[a] The ▲ symbol indicates that variable SecondaryStationIsReset is set to true. The ▼ symbol indicates that the variable SecondaryStationIsReset is set to false.

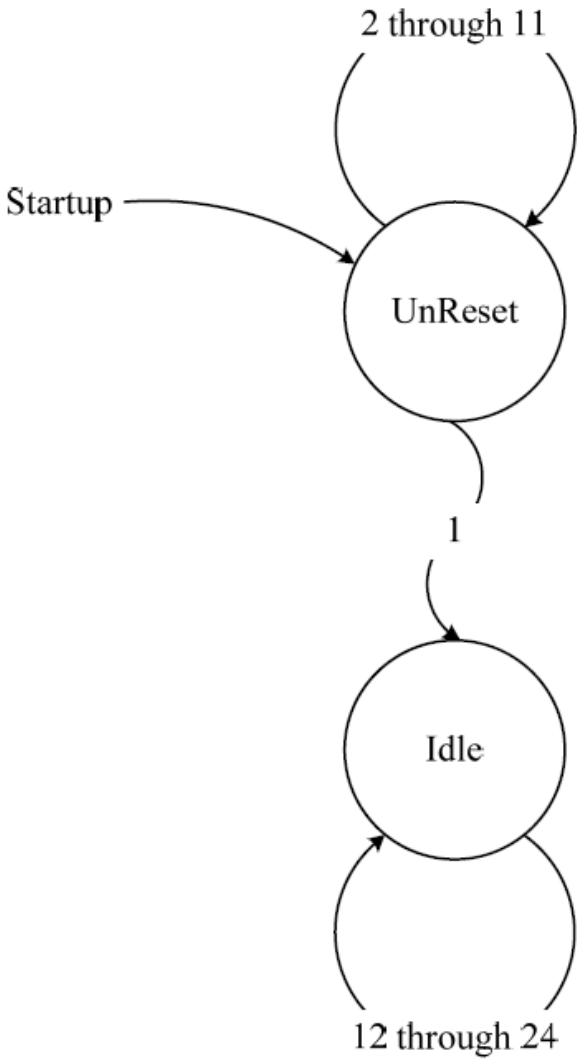


Figure 9-9—Secondary Station state diagram

Current state	Event that triggers an action and possible transition			Action			Transition to state	
A	B			C	D		E	
If the software state is	and a frame is received with link control octet fields			then perform this action[a]	and send this reply		and go to this state	
	FCB	FCV	Func		Func	Comments		
UnReset	X	0	0	▲ Set EFCB = 1	0	Ack.	Idle	1
	X	1		Do nothing.	15	May optionally not transmit anything.	UnReset	2
	X	X	1, 5, … 8, 10, … 15	Do nothing.	15	May optionally not transmit anything.	UnReset	3
	X	0	2	Do nothing.	15	May optionally not transmit anything.	UnReset	4
	X	1		Do nothing.	1	May optionally not transmit anything.	UnReset	5
	X	0	3	Do nothing.	15	May optionally not transmit anything.	UnReset	6
	X	1		Do nothing.	1	May optionally not transmit anything.	UnReset	7
	X	0	4	Send request to application parsing routine.	Do not transmit anything		UnReset	8
	X	1		Do nothing.			UnReset	9
	X	0	9	Do nothing.	11	Status of link.	UnReset	10
	X	1		Do nothing.	15	May optionally not transmit anything.	UnReset	11
Idle	X	0	0	Set EFCB = 1.	0	Ack.	Idle	12
	X	1		Do nothing.	15	May optionally not transmit anything.	Idle	13
	X	X	1, 5, … 8, 10, … 15	Do nothing.	15	May optionally not transmit anything.	Idle	14
	X	0	2	Do nothing.	15	May optionally not transmit anything.	Idle	15
	== EFCB	1		Toggle EFCB	0	Ack.	Idle	16
	!=EFCB			Do nothing.	–	Re-transmit most recent response that contained function code 0 (ACK) or 1 (NACK).	Idle	17
	X	0	3	Do nothing.	15	May optionally not transmit anything.	Idle	18
	== EFCB	1		Send request to application parsing routine. Toggle EFCB.	0	Ack.	Idle	19
	!=EFCB			Do nothing.	0	Ack.	Idle	20

Current state	Event that triggers an action and possible transition			Action			Transition to state	
A	B			C	D		E	
If the software state is	and a frame is received with link control octet fields			then perform this action[a]	and send this reply		and go to this state	
	FCB	FCV	Func		Func	Comments		
Idle	X	0	4	Send request to application parsing routine.	Do not transmit anything		Idle	21
	X	1		Do nothing.			Idle	22
	X	0	9	Do nothing.	11	Status of link.	Idle	23
	X	1		Do nothing.	15	May optionally not transmit anything.	Idle	24

[a] The ▲ symbol indicates that variable SecondaryStationIsReset is set to true.

DNP3 Exchange Samples

Reset Link Example		
--►	05 64 05 C0 01 00 00 04 E9 21	Reset link states
◄--	05 64 05 00 00 04 01 00 19 A6	Ack

Integrity Poll Example		
--►	05 64 14 F3 01 00 00 04 0A 3B C0 C3 01 3C 02 06 3C 03 06 3C 04 06 3C 01 06 9A 12	Request class 1, 2, 3 and 0 data
◄--	05 64 05 00 00 04 01 00 19 A6	Link layer confirm
◄--	05 64 05 40 00 04 01 00 A3 96	Reset link states
--►	05 64 05 80 01 00 00 04 53 11	Ack
◄--	05 64 53 73 00 04 01 00 03 FC C1 E3 81 96 00 02 01 28 01 00 00 00 01 02 01 28 05 24 01 00 01 00 01 02 01 28 01 00 02 00 01 02 01 28 B4 77 01 00 03 00 01 20 02 28 01 00 00 00 01 00 00 20 A5 25 02 28 01 00 01 00 01 00 00 01 01 01 00 00 03 00 2F AC 00 1E 02 01 00 00 01 00 01 00 00 01 00 00 16 ED	Response. IIN = device restart, need time, class 1 & 2 events. 4 binary input events, 2 analog input events, 4 binary inputs and 2 analog inputs.
--►	05 64 05 80 01 00 00 04 53 11	Link layer confirm
--►	05 64 08 C4 01 00 00 04 A4 CF C1 C3 00 20 3F	Application layer confirm

Reset Restart IIN Bit		
--►	05 64 0E C4 01 00 00 04 7D A4 C0 C4 02 50 01 00 07 07 00 64 11	Request write IIN1.7 = 0
◄--	05 64 0A 44 00 04 01 00 59 5E C2 C4 81 10 00 93 AD	Null response

Set Time and Date		
--►	05 64 12 C4 01 00 00 04 0E 0B C0 C5 02 32 01 07 01 F8 B8 6C AA F0 00 98 98	Request write time and date
◄--	05 64 0A 44 00 04 01 00 59 5E C3 C5 81 00 00 55 93	Null response

Key: --► Master station transmissions (Address 1024 decimal).
◄-- Outstation transmissions (Address 1).

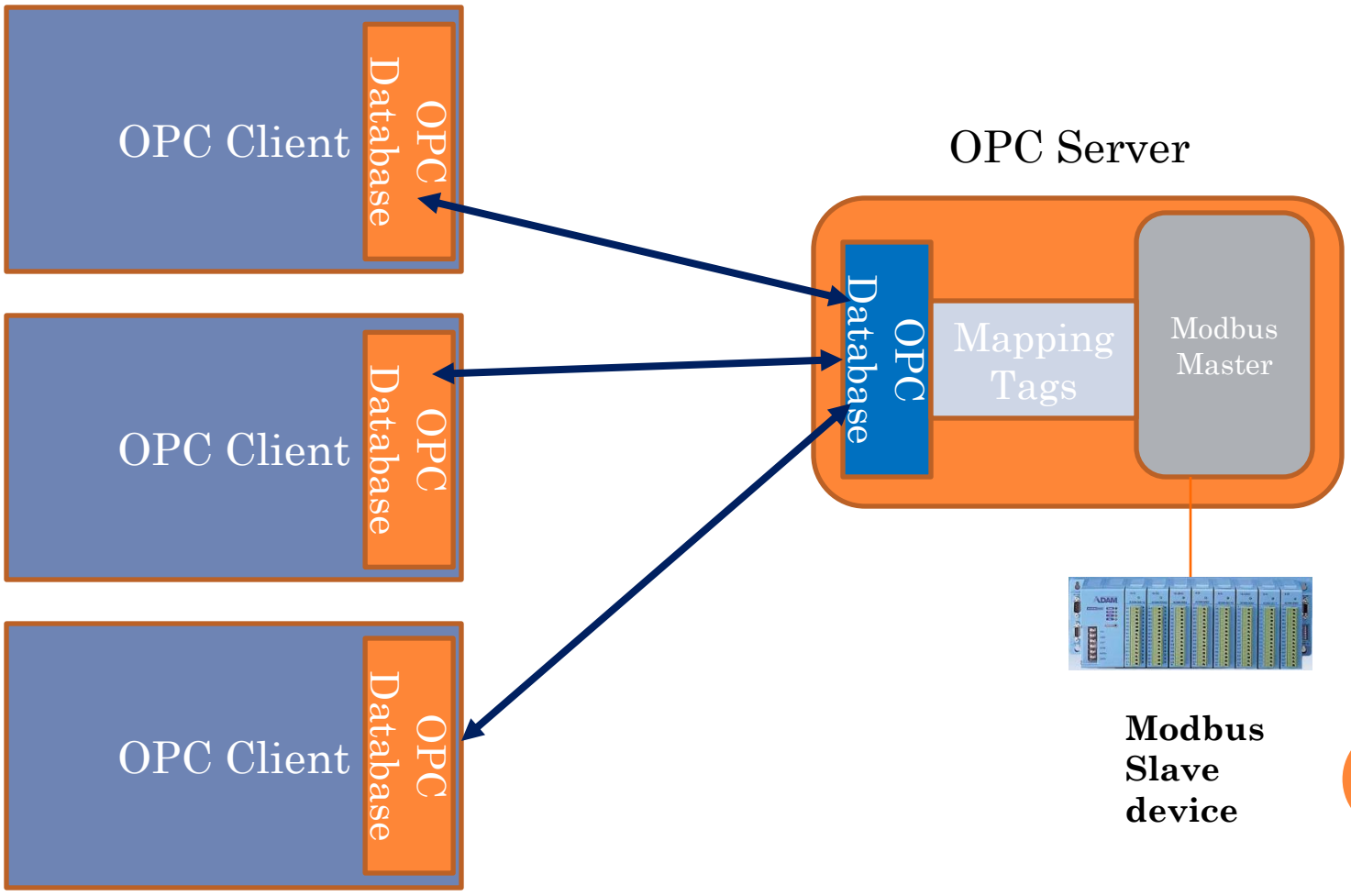
What is OPC UA Protocol ?

- At 1993 Microsoft released COM/DCOM technologies .
- COM is used for real time data transfer between two or more Windows Applications.
- DCOM is used for communication over Network .
- COM is base technology for many other Microsoft technologies like ActiveX and OLE .
- COM is based on Interface concept , Means software components are talking with no knowledge of their internal implementation.

OPC Classic (OLE for Process Control)

- OPC foundation at 1996 released OPC standard based on COM/DCOM technologies .
- OPC specifies the communication of real time plant data between control devices from different manufacturer .
- Specification defined a standard set of objects , interface and methods for use in process control to facilitate interoperability .
  - OPC DA (data Access) is used for read write of real time data .
  - OPC HA (Historical Access) is used for access archived data in Devices .
  - OPC AE(Alarm and Event) is used for exchange of alarms and events between client and server .

OPC Structure

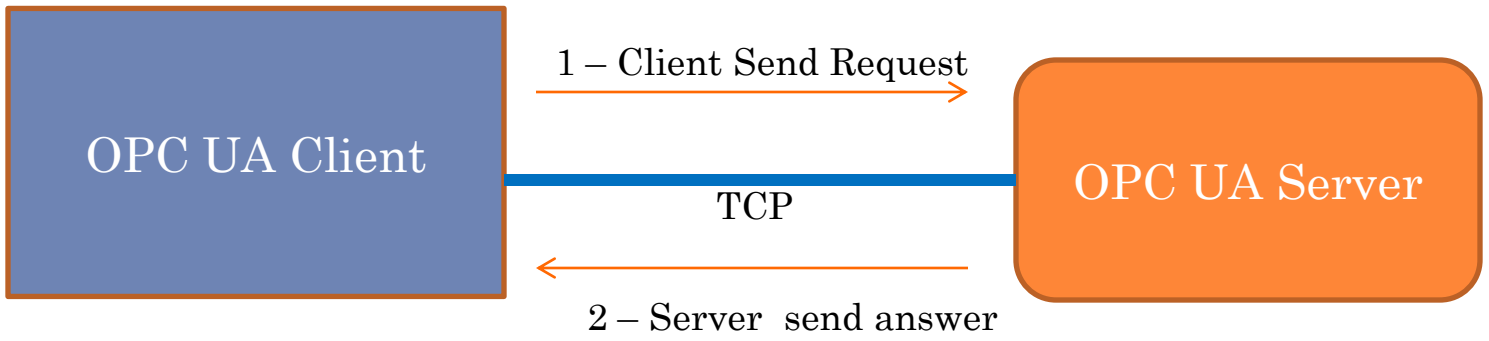


OPC Classic Problems

- Only based on Windows OS
- DCOM is very difficult to configure . It has many different security policy for different windows
- DCOM is not very secure
- OPC is not support event based communications between client and server
- No powerful data modeling
- No redundancy defined in standard
- Not completely implemented for windows CE and Windows Mobile

OPC UA Open Platform Communication Unified Architecture

- OPC foundation solved all OPC classic problems with UA Standard .
- The first version was release after 3 years hard working in 2006 .
- OPC UA is based on Service Oriented Architecture (SOA) and communication layer can be TCP Binary – Http – Web Service , …



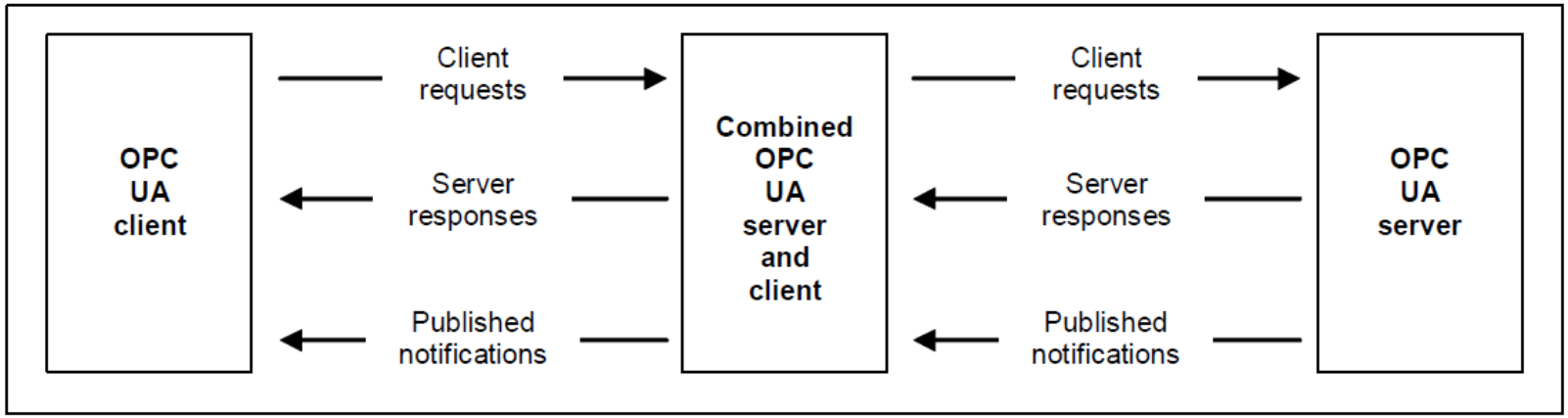
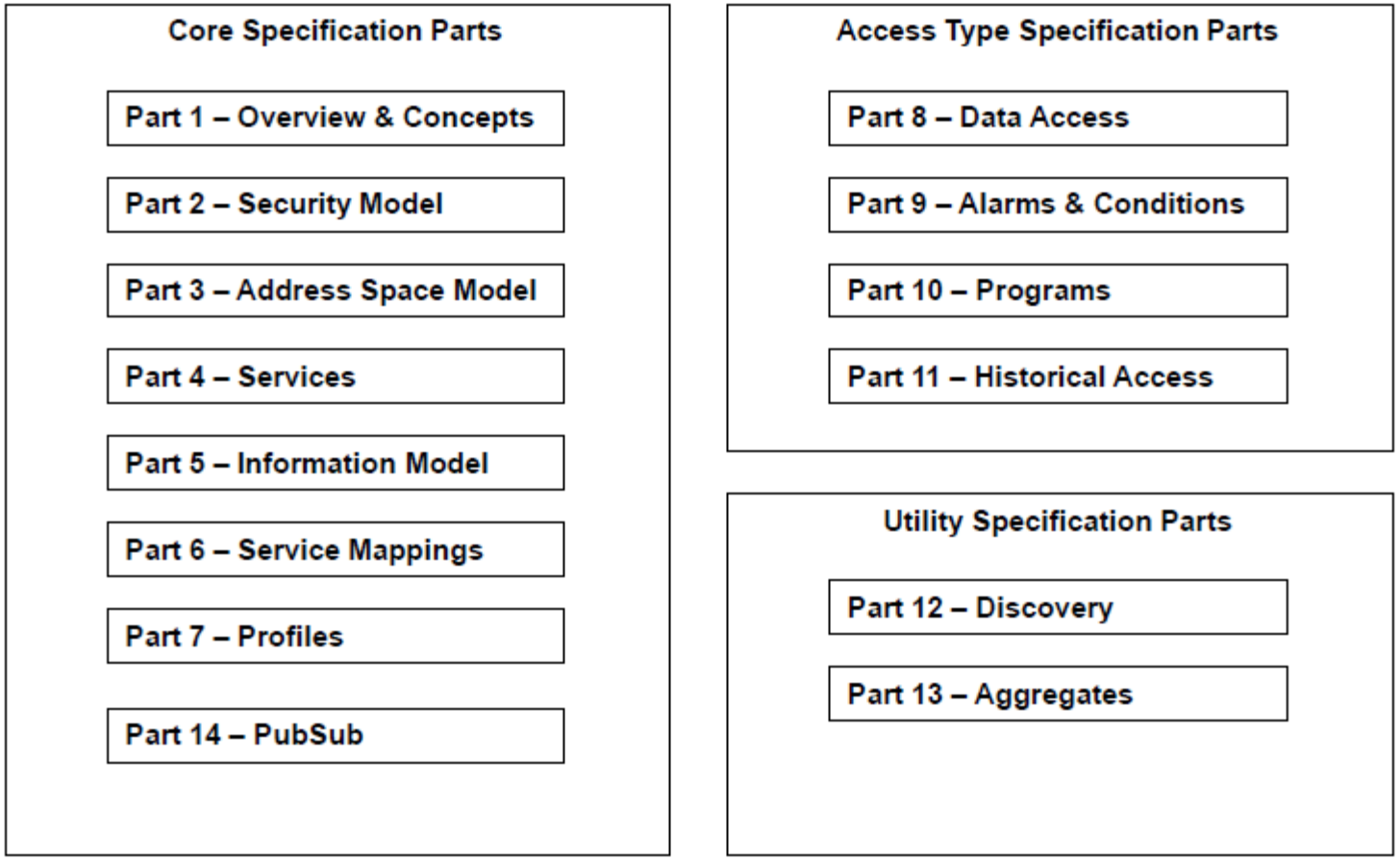


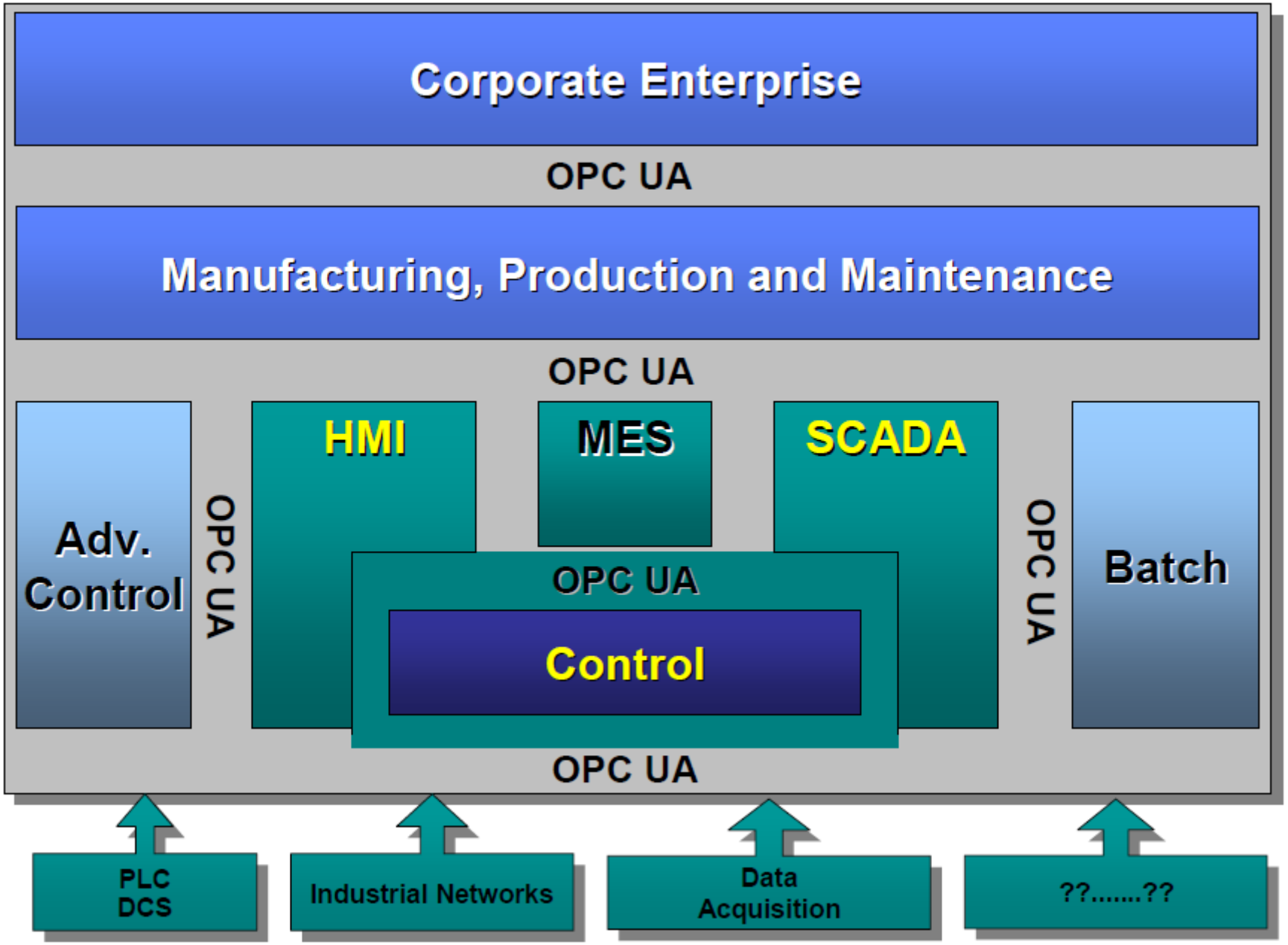
Figure 3 – OPC UA System architecture

OPC UA Specification 14 Part 1250 Page

- 1 - Concepts
- 2 - Security Model
- 3 - Address Space model
- 4 - Services
- 5 – Information Model
- 6 – Mappings
- 7 – profiles
- 8 – Data Access
- 9 – Alarms and Conditions
- 10 – Programs
- 11 – Historical access
- 12 – Discovery and global services
- 13 – Aggregation
- 14 - PubSub

OPC UA Multi-Part Specification




Corporate Enterprise
OPC UA
Manufacturing, Production and Maintenance
OPC UA
Adv. Control
OPC UA
HMI
MES
SCADA
OPC UA
Control
OPC UA
Batch
OPC UA
PLC DCS
Industrial Networks
Data Acquisition
??.....??

Can we use OPC UA for SCADA ?

- Can we use OPC UA for communication between RTU and master SCADA ?
- Can we use OPC UA instead of DNP3 or IEC104 ?
  - We MUST have following functionalities for a SCADA Protocol :
    - Different type of data types
    - Time label for tags
    - Time Synchronization mechanism
    - Event buffering and back filling
    - Freeze of counters
    - Integrity Poll
    - Send changes by RTU without Master Request
  - OPC UA not supported RED Items

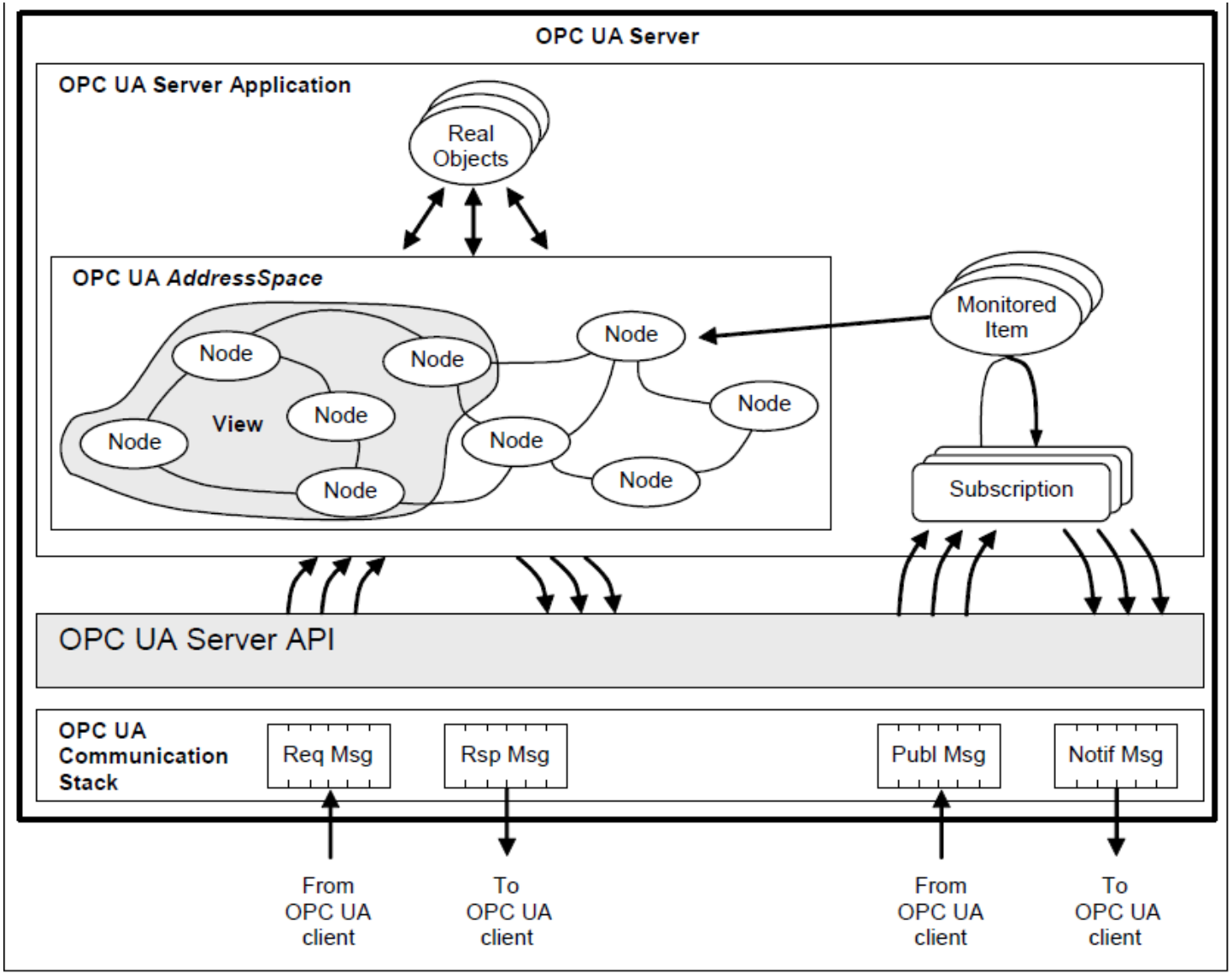


Figure 5 – OPC UA Server architecture

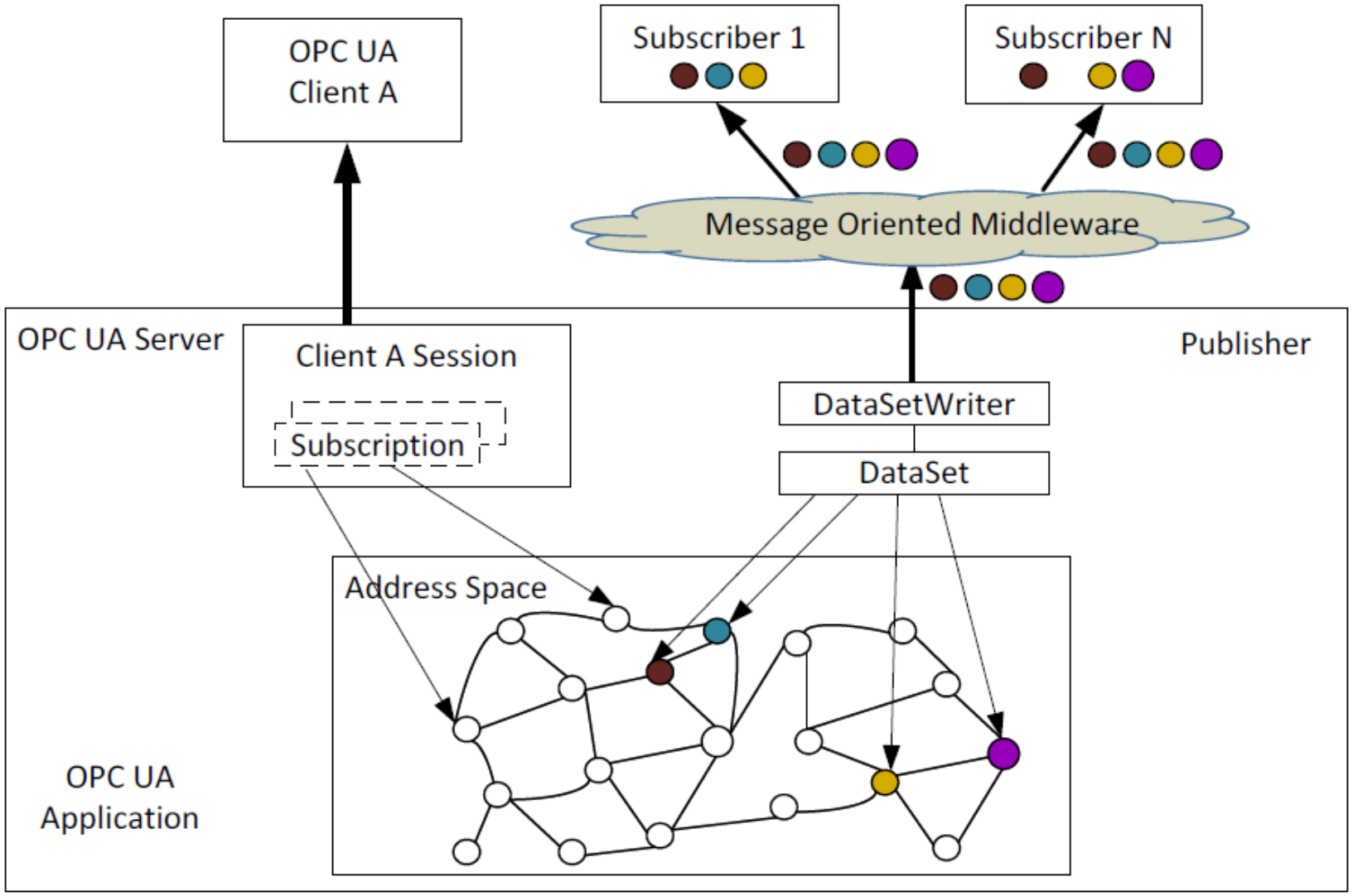


Figure 8 – Integrated Client Server and PubSub models

OPC UA Security

- OPC UA provides countermeasures to resist threats to the security of the information that is communicated
- OPC UA security works within the overall *Cyber Security Management System* (*CSMS*) of a site. Sites often have a *CSMS* that addresses security policy and procedures, personnel, responsibilities, audits, and physical security. A *CSMS* typically addresses threats that include those that were described in 4.3. They also analyze the security risks and determine what security controls the site needs.
- the security requirements of the OPC UA interfaces that are deployed at a site are specified by the site, not by the OPC UA specification

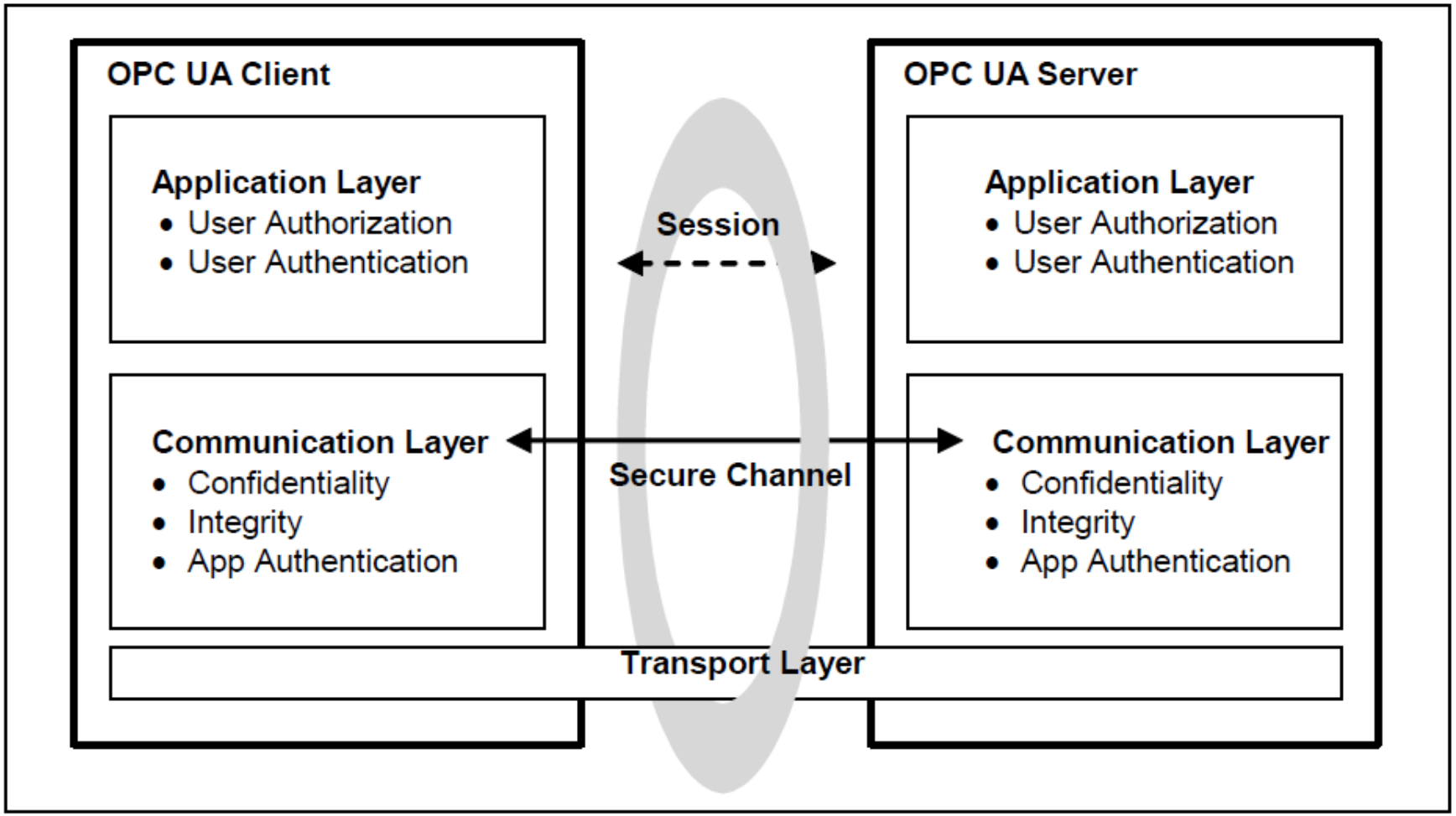


Figure 2 – OPC UA security architecture – Client / Server

Applications Authentication

- *OPC UA Applications* support *Authentication* of the entities with which they are communicating. As specified in the *GetEndpoints* and *OpenSecureChannel* services in Part 4, OPC UA *Client* and *Server* applications identify and authenticate themselves with *X.509 v3 Certificate*s and associated private keys (see [X509]). Some choices of the communication stack require these *Certificate*s to represent the machine or user instead of the application

Secure channel

- The *Secure Channel* provides
  - encryption to maintain *Confidentiality*,
  - *Message Signatures* to maintain *Integrity*
  - *Certificates* to provide application *Authentication*
- OPC UA supports the selection of several security modes:
  - “None”,
  - “Sign”,
  - “SignAndEncrypt”

OPC UA Address Space – Object Model

- The primary objective of the OPC UA *AddressSpace* is to provide a standard way for *Servers* to represent *Objects* to *Clients*.
- It defines *Objects* in terms of *Variables* and *Methods*. It also allows relationships to other *Objects* to be expressed

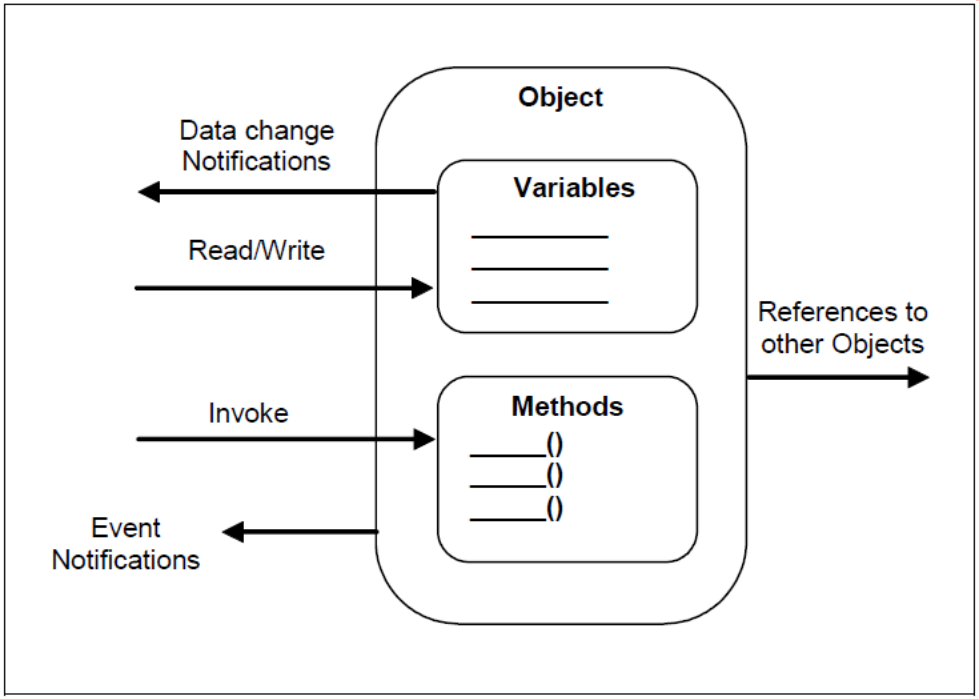


Figure 2 – OPC UA Object Model

Node Model

- The set of *Objects* and related information that the OPC UA *Server* makes available to *Clients* is referred to as its *AddressSpace*.
- Objects and their components are represented in the *AddressSpace* as a set of *Nodes* described by *Attributes* and interconnected by *References*.

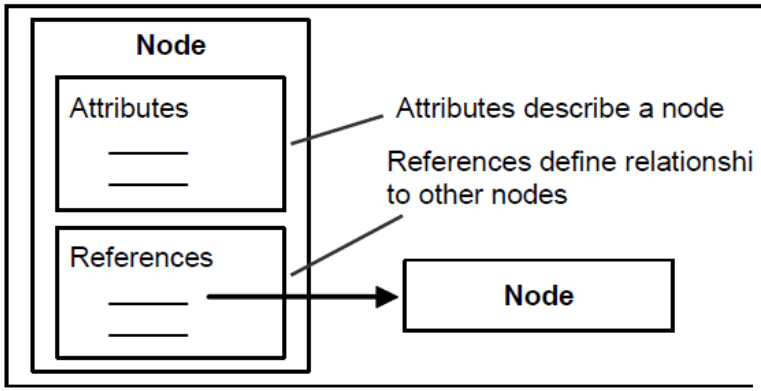
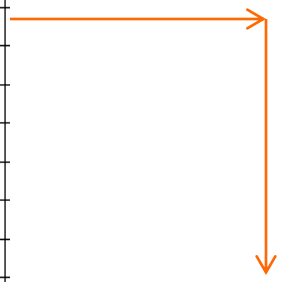


Figure 3 – AddressSpace Node Model

Name	Use	Data Type
Attributes		
NodeId	M	NodeId
NodeClass	M	NodeClass
BrowseName	M	QualifiedName
DisplayName	M	LocalizedText
Description	O	LocalizedText
WriteMask	O	AttributeWriteMask
UserWriteMask	O	AttributeWriteMask
RolePermissions	O	RolePermissionType[]
UserRolePermissions	O	RolePermissionType[]
AccessRestrictions	O	AccessRestrictionsType
References		

Node Class

Name	Use	Data Type
Attributes		
NodeId	M	NodeId
NodeClass	M	NodeClass
BrowseName	M	QualifiedName
DisplayName	M	LocalizedText
Description	O	LocalizedText
WriteMask	O	AttributeWriteMask
UserWriteMask	O	AttributeWriteMask
RolePermissions	O	RolePermissionType[]
UserRolePermissions	O	RolePermissionType[]
AccessRestrictions	O	AccessRestrictionsType
References		



Name
OBJECT_1
VARIABLE_2
METHOD_4
OBJECT_TYPE_8
VARIABLE_TYPE_16
REFERENCE_TYPE_32
DATA_TYPE_64
VIEW_128

Modeling Sample , OPC Classic

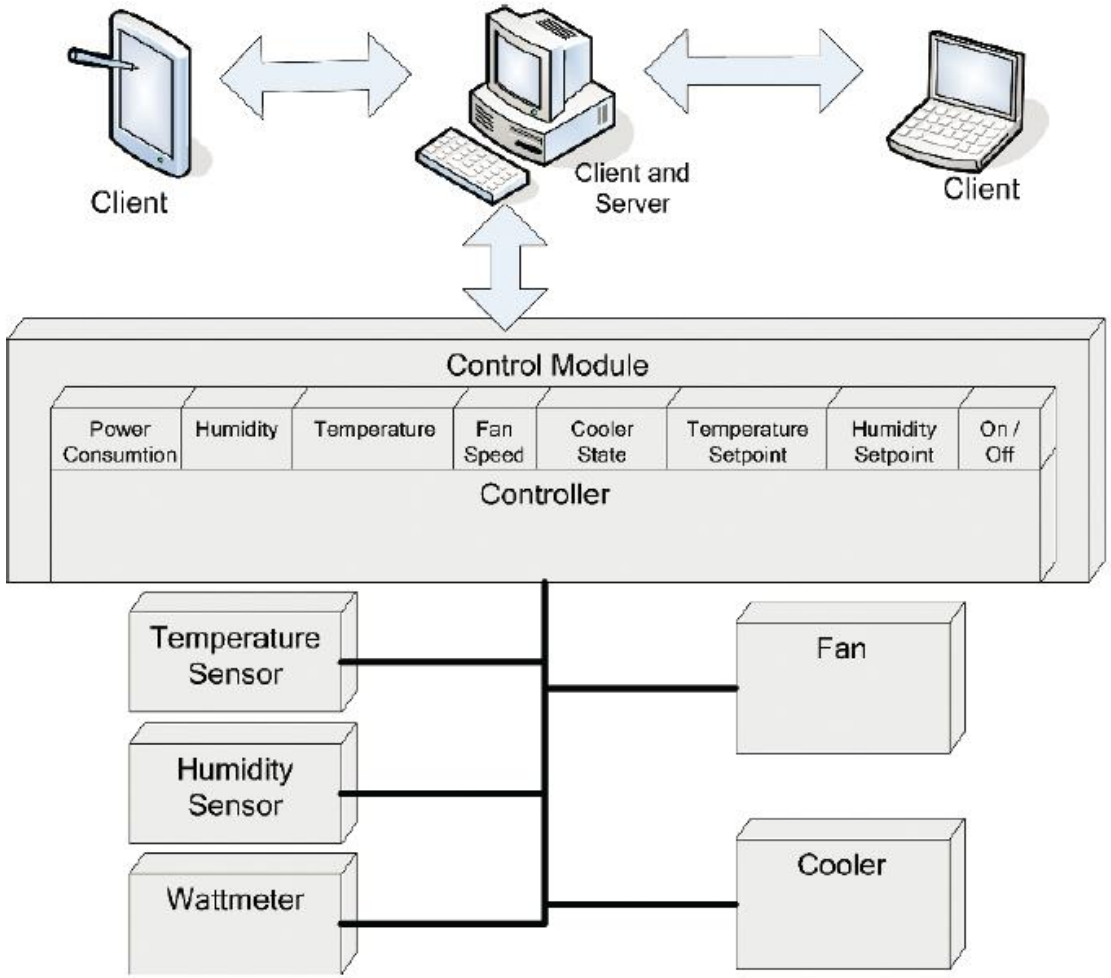


Fig. 3.1 Air conditioner application scenario

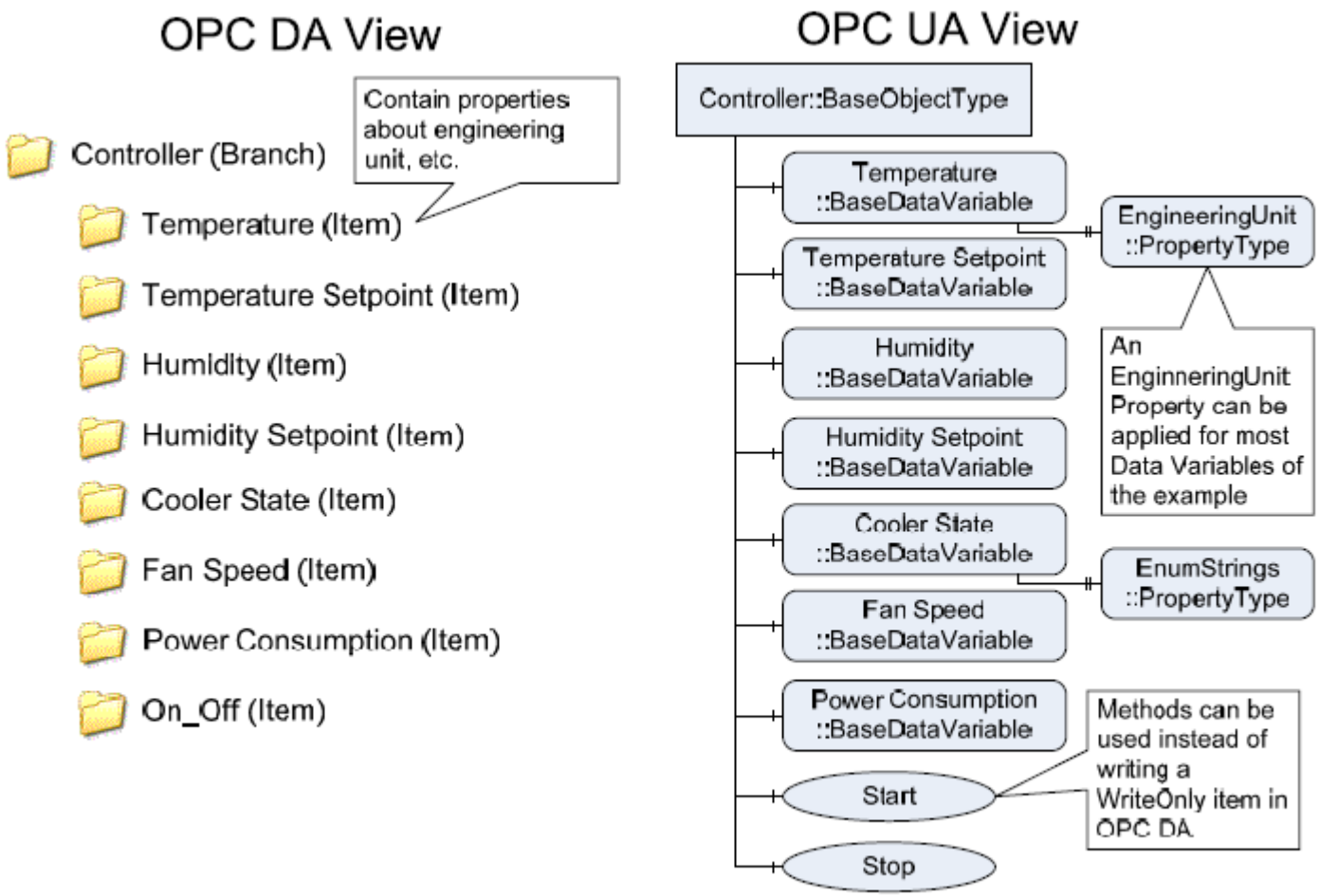


Fig. 3.2 Simple mapping to OPC DA and OPC UA

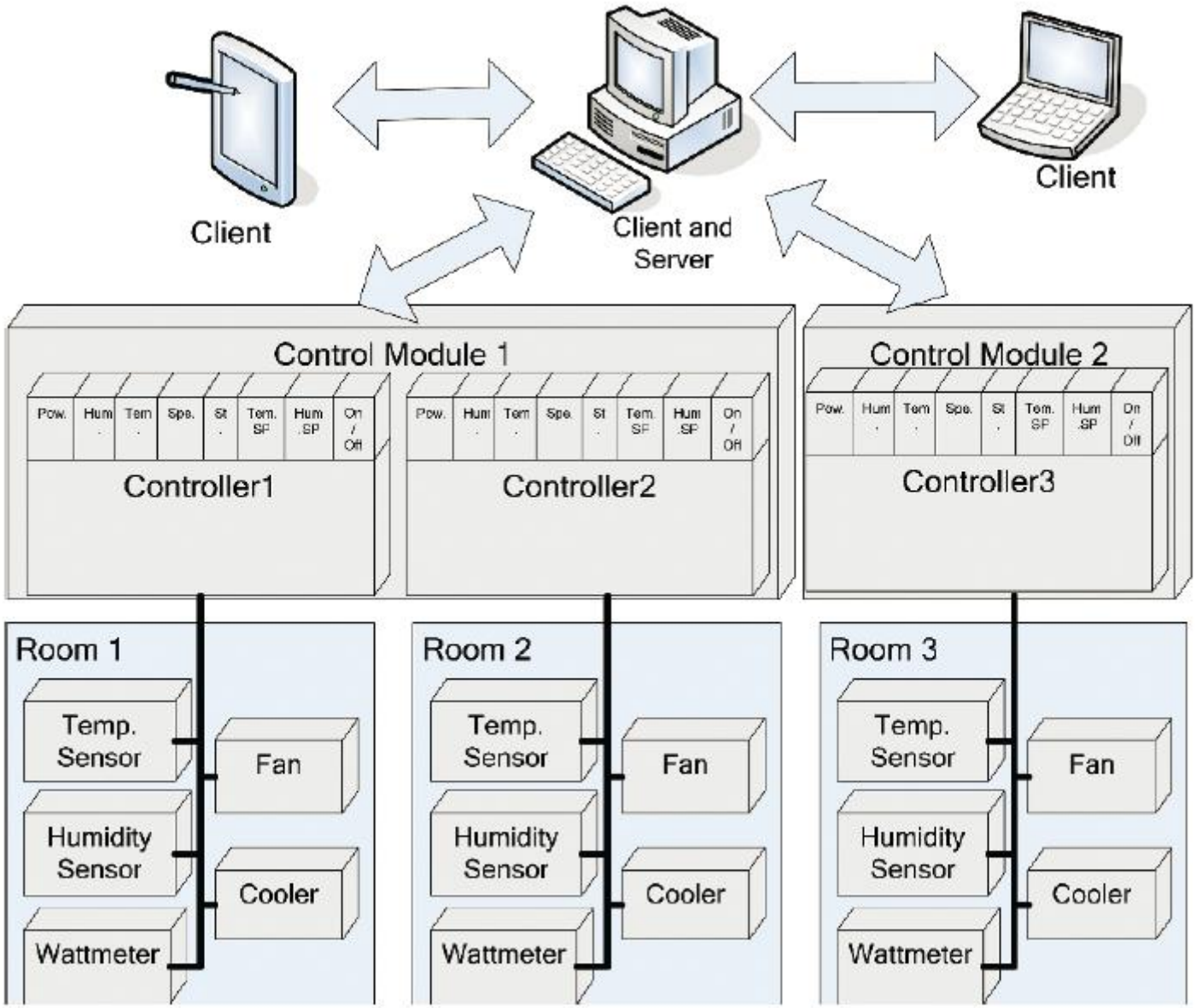


Fig. 3.3 Multiple air conditioner

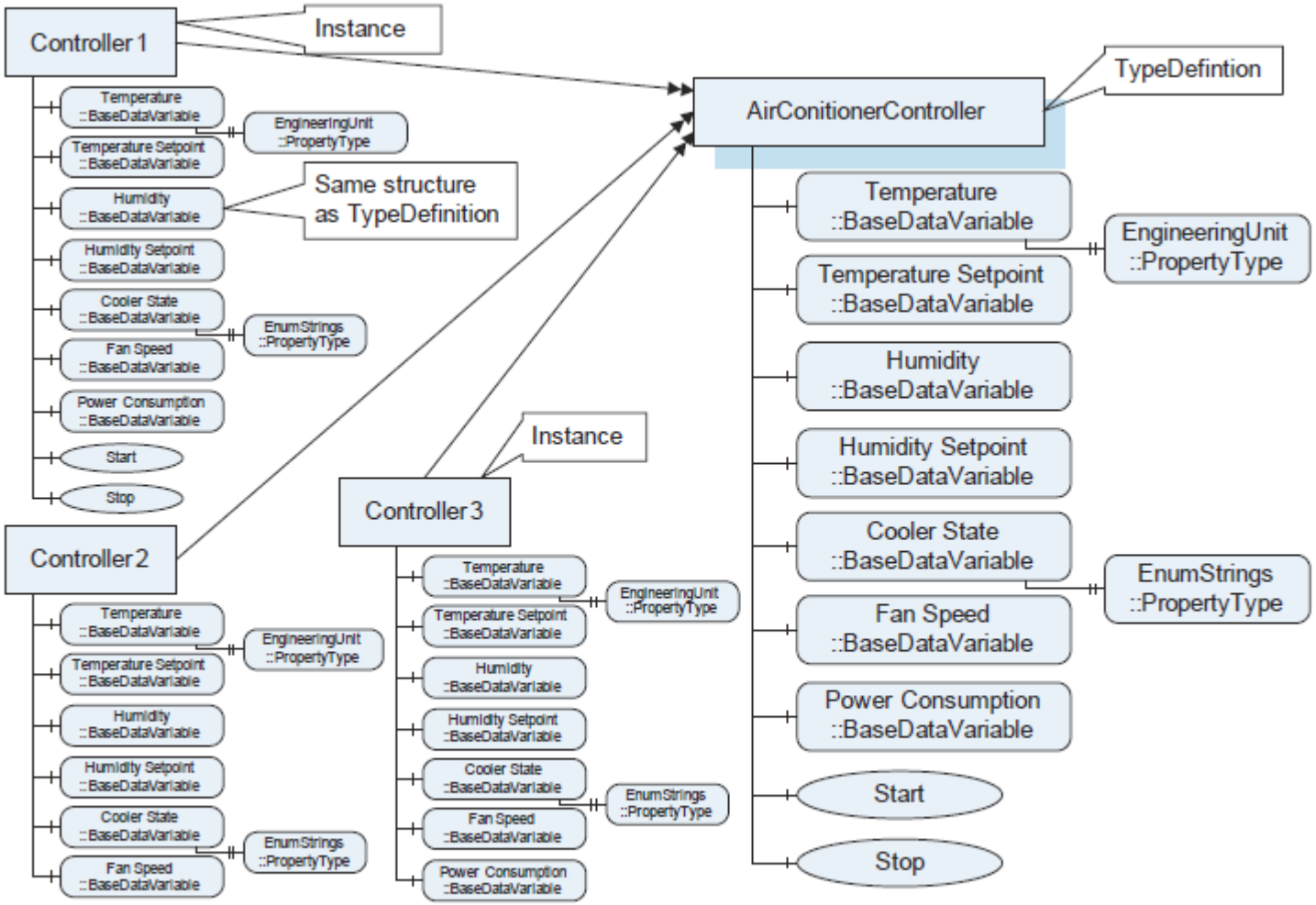


Fig. 3.4 TypeDefinition for the controller

OPC UA Services

Table 5.1 Services grouped by use cases

Use case	Service sets or services
Find servers	Discovery Services Set
Connection management between clients and servers	Secure Channel Service Set Session Service Set
Find information in the Address Space	View Service Set
Read and write data and metadata	Read and Write Service
Subscribe for data changes and Events	Subscription Service Set Monitored Item Service Set
Calling Methods defined by the server	Call Service
Access history of data and Events	HistoryRead and HistoryUpdate Service
Find information in a complex Address Space	Query Service Set
Modify the structure of the server Address Space	Node Management Service Set

Request and Response Header

Table 5.2 Request header parameters

Parameter	Description
AuthenticationToken	The secret Session identifier used to assign the Service call to the Session context created between client and server application
RequestHandle	A client defined handle assigned to the Service call
Timestamp	The time the client sent the request
TimeoutHint	The timeout set in the client side UA Stack for the call. This hint can be used by the server to cancel long running calls if the timeout expires
ReturnDiagnostic	Indicates if the client requests the server to return additional detailed diagnostic information in the case of an error instead of returning only a status code
AuditEntryId	A string that identifies the client or user that initiated the action. The string is empty if this parameter is not used. It is used for auditing (see Sect. 9.5)

Table 5.3 Response header parameters

Parameter	Description
ServiceResult	UA defined result code for the Service call
RequestHandle	The client defined handle assigned to the Service call
Timestamp	The time the server sent the response
ServiceDiagnostics	Client requested detailed diagnostic information

OPC UA Mappings

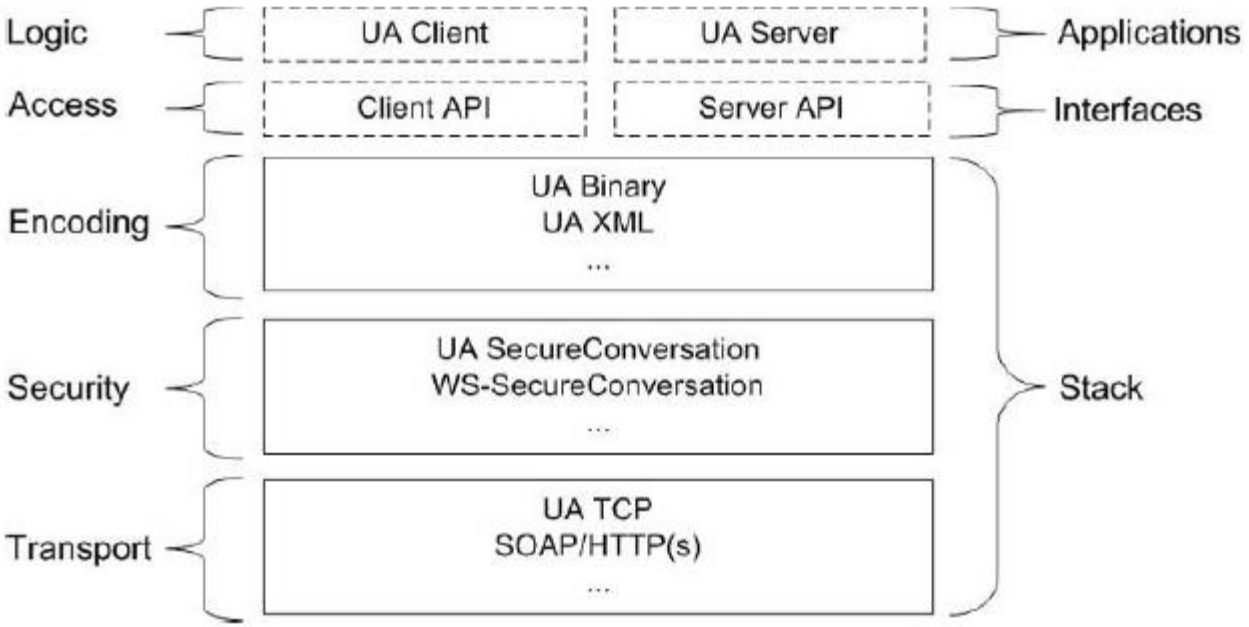


Fig. 6.1 Mappings

OPC UA Security Architecture

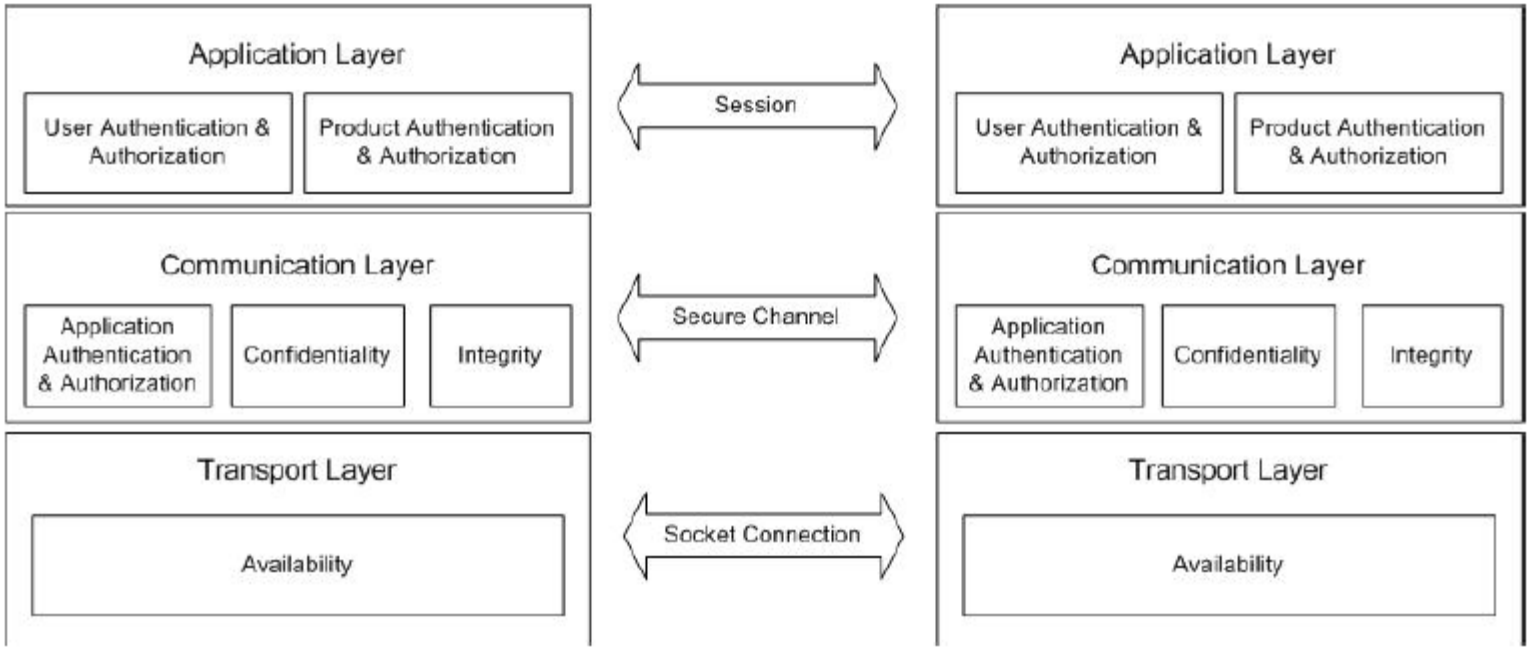


Fig. 7.2 OPC UA security architecture

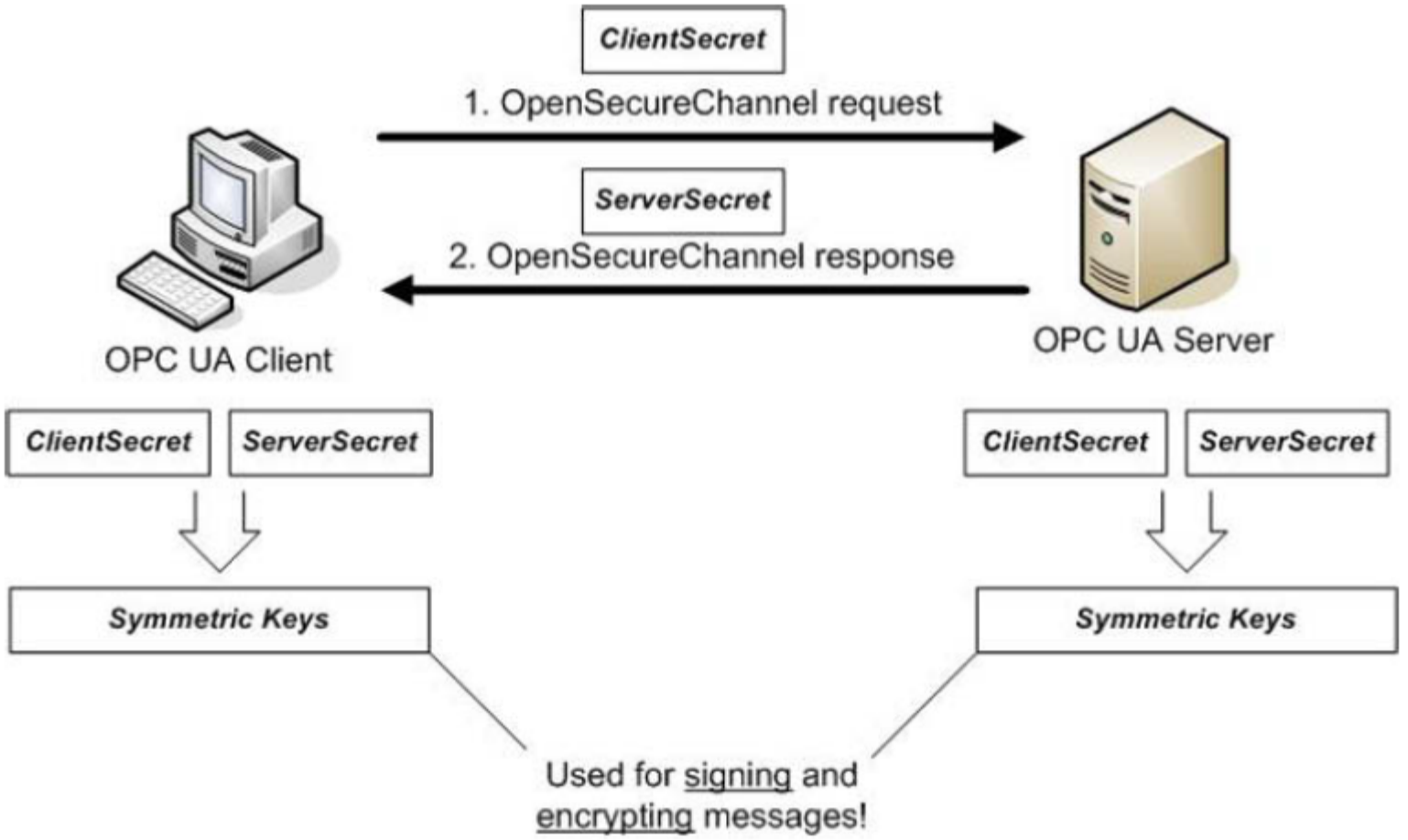


Fig. 7.3 Creating Symmetric Keys

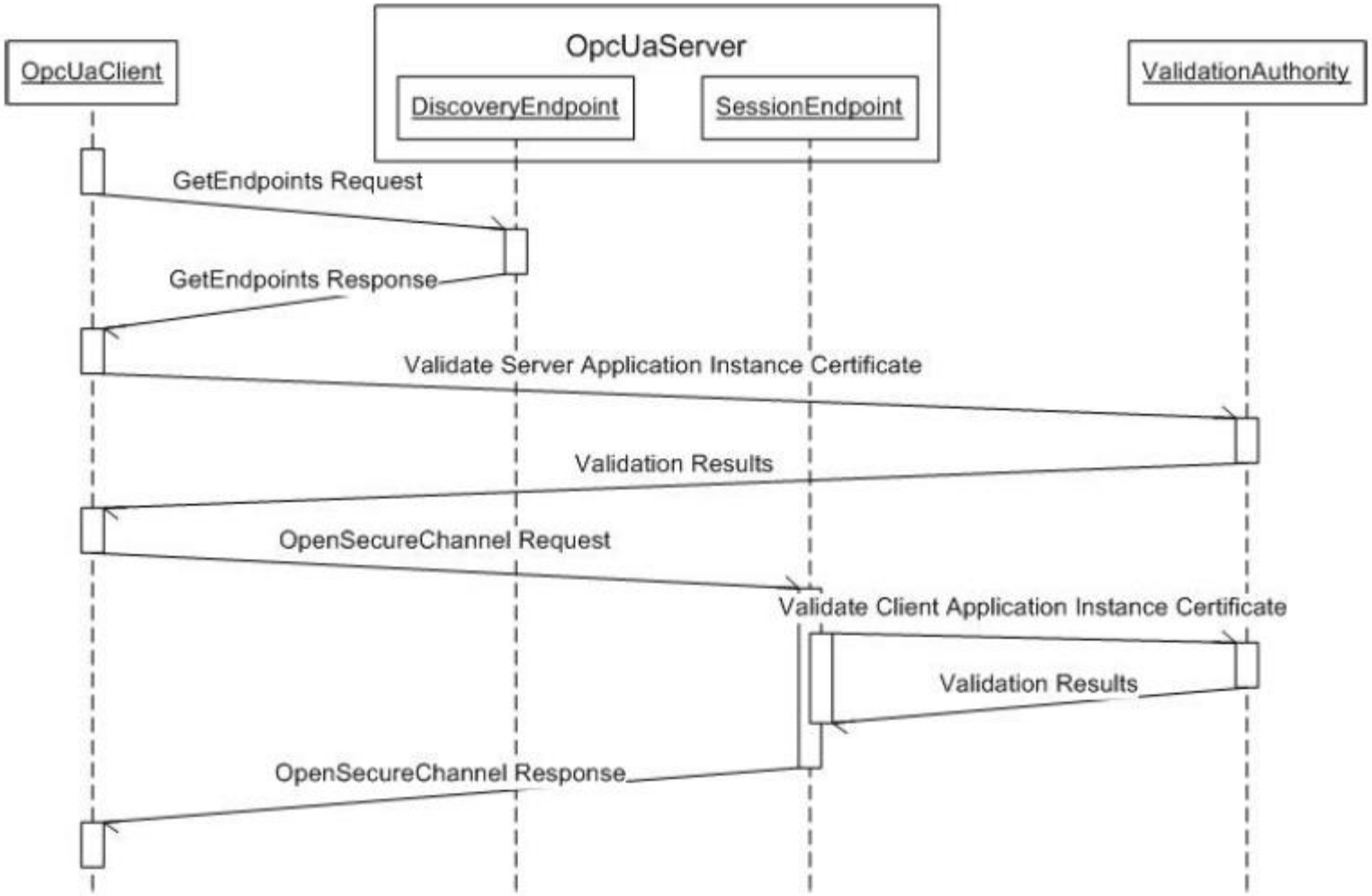


Fig. 7.4 Creating an OPC UA Secure Channel