
**Industrial automation systems —
Specification of subsets of the protocol for
ISO/IEC 9506**

Systèmes d'automatisation industrielle — Spécification de sous-ensembles du protocole pour l'ISO/CEI 9506



Contents	Page
Foreword	viii
1 Scope	1
2 Normative references	1
3 Protocol subset specification	2
3.1 Notation	2
3.2 Determination of the effective protocol	3
3.3 Protocol for companion standards	4
4 Protocol module	6
4.1 MMSpdu	6
4.2 Confirmed-RequestPDU	7
4.3 Unconfirmed-PDU	7
4.4 Confirmed-ResponsePDU	7
4.5 Confirmed-ErrorPDU	8
4.6 Modifier	8
4.7 ConfirmedServiceRequest	8
4.8 AdditionalService-Request	14
4.9 CS-Request-Detail	16
4.10 UnconfirmedService	16
4.11 CS-Unconfirmed-Detail	17
4.12 ConfirmedServiceResponse	17
4.13 AdditionalService-Response	23
4.14 CS-Response-Detail	25
4.15 ServiceError	25
4.16 AdditionalService-Error	28
4.17 TimeOfDay	29
4.18 Identifier	29
4.19 Integer's	29
4.20 ObjectName	29
4.21 ApplicationReference	30
4.22 FileName	30
4.23 Priority	30
4.24 Severity	30
4.25 Initiate-PDU's	30
4.26 Initiate-Detail's	31
4.27 ServiceSupportOptions	31
4.28 ParameterSupportOptions	34

© ISO/IEC 1994
All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland
Printed in Switzerland

4.29	Conclude-PDU's	34
4.30	Cancel-PDS's	35
4.31	RejectPDU	35
4.32	Status	36
4.33	Robot Status	37
4.34	UnsolicitedStatus	37
4.35	GetNameList	38
4.36	CsAdditionalObjectClasses	39
4.37	Identify	39
4.38	Rename	39
4.39	GetCapabilityList	40
4.40	InitiateDownloadSequence	40
4.41	DownloadSegment	40
4.42	TerminateDownloadSequence	40
4.43	InitiateUploadSequence	41
4.44	UploadSegment	41
4.45	TerminateUploadSequence	41
4.46	RequestDomainDownload	41
4.47	RequestDomainUpload	41
4.48	LoadDomainContent	42
4.49	StoreDomainContent	42
4.50	DeleteDomain	42
4.51	GetDomainAttributes	42
4.52	DomainState	43
4.53	CreateProgramInvocation	43
4.54	DeleteProgramInvocation	44
4.55	Start	44
4.56	ProgramInvocationState	44
4.57	Stop	45
4.58	Resume	45
4.59	Reset	45
4.60	Kill	45
4.61	GetProgramInvocationAttributes	46
4.62	TypeSpecification	46
4.63	AlternateAccess	47
4.64	AccessResult	48
4.65	Data	48
4.66	DataAccessError	49
4.67	VariableAccessSpecification	49
4.68	VariableSpecification	49
4.69	ScatteredAccessDescription	50
4.70	Address	50
4.71	Read	50
4.72	Write	50
4.73	InformationReport	50
4.74	GetVariableAccessAttributes	51

4.75	DefineNamedVariable	51
4.76	DefineScatteredAccess	51
4.77	GetScatteredAccessAttributes	51
4.78	DeleteVariableAccess	52
4.79	DefineNamedVariableList	52
4.80	GetNamedVariableListAttributes	52
4.81	DeleteNamedVariableList	53
4.82	DefineNamedType	53
4.83	GetNamedTypeAttributes	53
4.84	DeleteNamedType	53
4.85	TakeControl	54
4.86	RelinquishControl	54
4.87	DefineSemaphore	54
4.88	DeleteSemaphore	55
4.89	ReportSemaphoreStatus	55
4.90	ReportPoolSemaphoreStatus	55
4.91	ReportSemaphoreEntryStatus	55
4.92	AttachToSemaphore	56
4.93	SemaphoreEntry	56
4.94	Input	56
4.95	Output	56
4.96	DefineEventCondition	57
4.97	DeleteEventCondition	57
4.98	GetEventConditionAttributes	57
4.99	ReportEventConditionStatus	58
4.100	AlterEventConditionMonitoring	58
4.101	TriggerEvent	59
4.102	DefineEventAction	59
4.103	DeleteEventAction	59
4.104	GetEventActionAttributes	59
4.105	ReportEventActionStatus	60
4.106	DefineEventEnrollment	60
4.107	DeleteEventEnrollment	61
4.108	GetEventEnrollmentAttributes	61
4.109	EventEnrollment	61
4.110	ReportEventEnrollmentStatus	62
4.111	AlterEventEnrollment	62
4.112	Eventnotification	62
4.113	AcknowledgeEventNotification	63
4.114	GetAlarmSummary	63
4.115	GetAlarmEnrollmentSummary	64
4.116	AttachToEventCondition	65
4.117	EC-Class	65
4.118	EC-State	65
4.119	EE-State	65
4.120	Transitions	66

4.121	AlarmAckRule	66
4.122	EE-Class	66
4.123	EE-Duration	66
4.124	EN-Additional-Detail	66
4.125	EventTime	67
4.126	ReadJournal	67
4.127	WriteJournal	67
4.128	InitializeJournal	67
4.129	ReportJournalStatus	68
4.130	CreateJournal	68
4.131	DeleteJournal	68
4.132	EntryContent	68
4.133	ExchangeData	69
4.134	GetDataExchangeAttributes	69
4.135	AccessCondition	69
4.136	DefineAccessControlList	69
4.137	GetAccesscontrolListAttributes	70
4.138	ReportAccessControlledObjects	71
4.139	DeleteAccessControlList	71
4.140	ChangeAccessControl	72
4.141	ObtainFile	73
4.142	VMDStop	73
4.143	VMDReset	73
4.144	Select	73
4.145	AlterProgramInvocationAttributes	73
4.146	ControlElement	74
4.147	InitiateUnitControlLoad	74
4.148	UnitControlLoadSegment	74
4.149	UnitControlUpload	74
4.150	StartUnitControl	75
4.151	StopUnitControl	75
4.152	CreateUnitControl	75
4.153	AddToUnitControl	75
4.154	RemoveFromUnitControl	76
4.155	GetUnitControlAttributes	76
4.156	LoadUnitControlFromFile	76
4.157	StoreUnitControlToFile	76
4.158	DeleteUnitControl	77
4.159	DefineEventConditionList	77
4.160	DeleteEventConditionList	77
4.161	AddEventConditionListReference	77
4.162	RemoveEventConditionListReference	78
4.163	GetEventConditionListAttributes	78
4.164	ReportEventConditionListStatus	78
4.165	EventConditionStatus	79
4.166	AlterEventConditionListMonitoring	79

Annex A File Management Protocol 80

A.1 FileOpen 80

A.2 FileRead 80

A.3 FileClose 80

A.4 FileRename 81

A.5 FileDelete 81

A.6 FileDirectory 81

Foreword

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

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The main task of technical committees is to prepare International Standards, but in exceptional circumstances a technical committee may propose the publication of a Technical Report of one of the following types:

- type 1, when the required support cannot be obtained for the publication of an International Standard, despite repeated efforts;
- type 2, when the subject is still under technical development or where for any other reason there is the future but not immediate possibility of an agreement on an International Standard;
- type 3, when a technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example).

Technical Reports of types 1 and 2 are subject to review within three years of publication, to decide whether they can be transformed into International Standards. Technical Reports of type 3 do not necessarily have to be reviewed until the data they provide are considered to be no longer valid or useful.

ISO/IEC TR 13345, which is a Technical Report of type 2, was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration*, Subcommittee SC 5, *Architecture and communication*.

This document is being issued in the type 2 Technical Report series of publications (according to subclause G.4.2.2 of part 1 of the ISO/IEC Directives, 1992) as a "prospective standard for provisional application" in the field of industrial automation because there is an urgent need for guidance on how standards in this field should be used to meet an identified need.

This document is not be regarded as an “International Standard”. It is proposed for provisional application so that information and experience of its use in practice may be gathered. Comments on the content of this document should be sent to the ISO Central Secretariat.

A review of this type 2 Technical Report will be carried out not later than two years after its publication with the options of: extension for another two years; conversion into an International Standard; or withdrawal.

Annex A forms an integral part of this Technical Report.

Industrial automation systems — Specification of subsets of the protocol for ISO/IEC 9506

1 Scope

ISO/IEC 9506-1 and ISO/IEC 9506-2 were published in 1990. Since publication, considerable experience has been gained by implementors identifying defects and other shortcomings. One major shortcoming of ISO/IEC 9506-2 is that although it allows subsetting of the protocol in accordance with parameters declared at the time of association establishment, no clear notation for describing the subsets is present. Moreover, for some combinations of parameters, there are possible ambiguities in the specification of the subset.

MMS prescribes a procedure, used at the time of association establishment, in which sets of parameters are exchanged for the purpose of identifying the services that may be performed during the association. The effect of identifying these parameters is such that the protocol available to be used during the association is limited to a proper subset of the entire protocol specified in ISO/IEC 9506-2. The parameters exchanged are of two types:

- a) those that are announced by the two MMS users, the service conformance building block (CBB),
- b) those that are negotiated between the two MMS users, the parameter CBB. These CBBs are proposed by the association initiator and either accepted or rejected by the association responder. Negotiation always works to reduce the set proposed, never to augment it.

Declaration of support of any service CBB requires inclusion of the protocol related to that service in the protocol set to be used on that association. Support of a parameter CBB usually results in the inclusion of some optional fields within the protocol of some service request or response. However, in some cases, support of a parameter CBB implies support of one or more service CBBs, regardless of whether or not support for those service CBBs has been declared. It is in such situations that ambiguity may arise. This Technical Report specifies a uniform technique for resolving such ambiguities.

It introduces a formal notation to describe the subsets in an unambiguous form, and a complete specification of the MMS protocol using this new notation. The notation also supports the protocol of the Companion Standards. The protocol of the Robot Companion Standard (ISO/IEC 9506-3), of the NCC Companion Standard (ISO/IEC 9506-4), and of the Process Industries Companion Standard (the subject of future International Standard ISO/IEC 9506-6) are included.

Annex A specifies the File Management Protocol, the subject of annex C of ISO/IEC 9506-2.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this Technical Report. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this Technical Report are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 9506-1:1990, *Industrial automation systems — Manufacturing Message Specification — Part 1: Service definition.*

ISO/IEC 9506-2:1990, *Industrial automation systems — Manufacturing Message Specification — Part 2: Protocol specification.*

ISO/IEC 9506-1/Amd.1:1993, *Industrial automation systems — Manufacturing Message Specification — Part 1: Service definition AMENDMENT 1 Data exchange.*

ISO/IEC 9506-2/Amd.1:1993, *Industrial automation systems — Manufacturing Message Specification — Part 2: Protocol specification AMENDMENT 1 Data exchange.*

ISO/IEC 9506-1/Amd.2:—¹⁾, *Industrial automation systems — Manufacturing Message Specification — Part 1: Service definition AMENDMENT 2 Conditioned service response.*

ISO/IEC 9506-2/Amd.2:—¹⁾, *Industrial automation systems — Manufacturing Message Specification — Part 2: Protocol specification AMENDMENT 2 Conditioned service response.*

ISO/IEC 9506-3:1991, *Industrial automation systems — Manufacturing Message Specification — Part 3: Companion standard for robotics.*

ISO/IEC 9506-4:1992, *Industrial automation systems — Manufacturing Message Specification — Part 4: Companion standard for numerical control.*

ISO/IEC 8824:1990, *Information technology — Open Systems Interconnection — Specification of Abstract Syntax Notation One (ASN.1).*

3 Protocol subset specification

3.1 Notation

The notation being introduced by this technical report has the form of a preprocessor language in which ASN.1 is embedded. It is very similar in concept to the macro preprocessor for the C language. There are only three commands used in this notation:

- IF (<list of arguments>)
- ELSE
- ENDIF.

1) To be published.

The IF command requires an argument list enclosed in parentheses; the arguments are the names of the conformance building blocks, either service or parameter. One or more such arguments must appear. If there is more than one argument, the arguments are separated by one or more spaces. The argument is treated as a boolean variable that has the value true if the corresponding service or parameter building block is supported as a result of the MMS Initiate exchange. If there is one argument, then the lines following the IF statement up to the ELSE statement or to the matching ENDIF statement (if no ELSE statement appears) are to be included in the resulting ASN.1 definition if and only if the conformance building block so named is supported. If there is more than one argument, the lines following the IF statement are to be included if any of the conformance building blocks in the argument list is supported. (This can be thought of as a 'logical OR' function of the conformance building blocks.)

IF statements may be nested to any depth; the effect of

```
IF ( x )  
IF ( y )
```

is to include the lines following these commands if and only if both x and y are true, that is if conformance block x and conformance block y are both included. (This can be thought of as a 'logical AND' function of the conformance building blocks.)

The ELSE statement may appear to allow ASN.1 statements to be included if a conformance building block is not true. Its use is similar to the normal use of ELSE in programming languages.

The ENDIF statement is used to end the scope of an IF statement or ELSE statement. Each IF statement must have a matching ENDIF statement.

3.2 Determination of the effective protocol

The protocol effective for any specific combination of service and parameter CBBs can be determined by the following procedure:

- a) For each service CBB and parameter CBB declared or negotiated by the Initiate exchange, set the corresponding argument equal to true.
- b) Process the entire ASN.1 module specified in clause 4 and in Annex A. For each IF statement, evaluate the argument.
 - i) If any of the elements in the argument is true, retain the statements following the IF statement up to a matching ENDIF or ELSE statement, if present. Discard the statements following the ELSE statement up to the matching ENDIF.
 - ii) If all the elements of the argument are false, discard the statements following up to the matching ELSE or ENDIF. If an ELSE statement is present, retain the statements following it to the matching ENDIF.

- iii) Discard the IF statement, its matching ENDIF, and the ELSE, if present. The result should be an ASN.1 module devoid of IF, ELSE, and ENDIF statements.
- c) In each production replace any occurrence of a comma followed immediately by a right brace with a right brace; i.e., delete such commas.
- d) Form an ASN.1 working module of productions containing only the first production (i.e. the production MMSPDU from 4.1).
- e) Add to the ASN.1 working module any productions referenced in the working module that are not already contained in that module.
- f) Repeat step c) until no new productions are added.

The resulting ASN.1 module is the module that is effective for this combination of CBBs. Receipt of a PDU which is not consistent with this module should result in a reject.

3.3 Protocol for companion standards

In addition to the protocol of ISO 9506-2, the protocol for all the amendments and companion standards has also been included in this report. In order to generate the protocol for a companion standard, the set of parameter CBBs has been augmented with three new symbols:

csr - true if this is an instance of the companion standard for robots,

csnc - true if this is an instance of the companion standard for NC equipment,

cspi - true if this is an instance of the companion standard for the process industries.

Except for the fact that the companion standards define a different abstract syntax that must be negotiated using presentation negotiation, each companion standard acts just like a parameter CBB.

In order to accomodate these companion standards, the list of service and parameter CBBs is augmented by the service and parameter CBBs from those standards.

The complete list of service CBBs is:

status	defineNamedVariableList
getNameList	getNamedVariableListAttributes
identify	deleteNamedVariableList
rename	defineNamedType
read	getNamedTypeAttributes
write	deleteNamedType
getVariableAccessAttributes	input
defineNamedVariable	output
defineScatteredAccess	takeControl
getScatteredAccessAttributes	relinquishControl
deleteVariableAccess	defineSemaphore

deleteSemaphore
 reportSemaphoreStatus
 reportPoolSemaphoreStatus
 reportSemaphoreEntryStatus
 initiateDownloadSequence
 downloadSegment
 terminateDownloadSequence
 initiateUploadSequence
 uploadSegment
 terminateUploadSequence
 requestDomainDownload
 requestDomainUpload
 loadDomainContent
 storeDomainContent
 deleteDomain
 getDomainAttributes
 createProgramInvocation
 deleteProgramInvocation
 start
 stop
 resume
 reset
 kill
 getProgramInvocationAttributes
 obtainFile
 defineEventCondition
 deleteEventCondition
 getEventConditionAttributes
 reportEventConditionStatus
 alterEventConditionMonitoring
 triggerEvent
 defineEventAction
 deleteEventAction
 getEventActionAttributes
 reportEventActionStatus
 defineEventEnrollment
 deleteEventEnrollment
 alterEventEnrollment
 reportEventEnrollmentStatus
 getEventEnrollmentAttributes
 acknowledgeEventNotification
 getAlarmSummary
 getAlarmEnrollmentSummary
 readJournal
 writeJournal
 initializeJournal
 reportJournalStatus

createJournal
 deleteJournal
 getCapabilityList
 fileOpen
 fileRead
 fileClose
 fileRename
 fileDelete
 fileDirectory
 unsolicitedStatus
 informationReport
 eventNotification
 attachToEventCondition
 attachToSemaphore
 conclude
 cancel
 getDataExchangeAttributes
 exchangeData
 vMDStop
 vMDReset
 select
 alterProgramInvocationAttributes
 initiateUnitControlLoad
 unitControlLoadSegment
 unitControlUpload
 startUnitControl
 stopUnitControl
 createUnitControl
 addToUnitControl
 removeFromUnitControl
 getUnitControlAttributes
 loadUnitControlFromFile
 storeUnitControlToFile
 deleteUnitControl
 defineEventConditionList
 deleteEventConditionList
 addEventConditionListReference
 removeEventConditionListReference
 getEventConditionListAttributes
 reportEventConditionListStatus
 alterEventConditionListMonitoring
 defineAccessControlList
 getAccessControlListAttributes
 reportAccessControlledObjects
 deleteAccessControlList
 changeAccessControl

The complete list of parameter (and parameter-like) CBBs is:

str1	real
str2	cei
vnam	des
valt	dei
vadr	recl
vsca	csr
tpy	csnc
vlis	cspi

4 Protocol module

This clause presents the entire protocol module using the macro notation described in Clause 3. It includes all of the protocol for the companion standard for robots, ISO 9506-3, the companion standard for NC equipment, ISO 9506-4, and the companion standard for the process industries, ISO 9506-6. It also includes the protocol for the Data Exchange services and the Conditioned Service Response services.

4.1 MMSpdu

```

IF ( csr csnc cspi )
IF ( csr )
ISO-9506-MMS-ROBOT-1 {iso standard 9506 part(3) mms-robot-module-version1(2) }
ENDIF
IF ( csnc )
ISO-9506-MMS-NCCS-1 {iso standard 9506 part(4) mms-nccs-module-version1(2)}
ENDIF
IF ( cspi )
ISO-9506-MMS-PROCESS-1 {iso standard 9506 part(6) mms-process-module-version1(2)}
ENDIF
ELSE
ISO-9506-MMS-1 {iso standard 9506 part(2) mms-abstract-syntax-version1(1) }
ENDIF

```

DEFINITIONS ::= BEGIN

```

IMPORTS AP-title, AP-invocation-id, AE-qualifier,
        AE-invocation-id, Authentication-value
FROM ISO-8650-ACSE-1
{iso standard 8650 abstract-syntax(2) acse-pdi(1)};

```

```

MMSpdu ::= CHOICE {
    confirmed-RequestPDU           [0] IMPLICIT Confirmed-RequestPDU,
    confirmed-ResponsePDU         [1] IMPLICIT Confirmed-ResponsePDU,
    confirmed-ErrorPDU             [2] IMPLICIT Confirmed-ErrorPDU,
    IF ( unsolicitedStatus informationReport eventNotification )
        unconfirmed-PDU           [3] IMPLICIT Unconfirmed-PDU,
    ENDIF
    rejectPDU                      [4] IMPLICIT RejectPDU,
    IF (cancel)
        cancel-RequestPDU         [5] IMPLICIT Cancel-RequestPDU,
        cancel-ResponsePDU        [6] IMPLICIT Cancel-ResponsePDU,
        cancel-ErrorPDU           [7] IMPLICIT Cancel-ErrorPDU,
    ENDIF
    initiate-RequestPDU            [8] IMPLICIT Initiate-RequestPDU,
    initiate-ResponsePDU           [9] IMPLICIT Initiate-ResponsePDU,

```

initiate-ErrorPDU	[10] IMPLICIT Initiate-ErrorPDU,
conclude-RequestPDU	[11] IMPLICIT Conclude-RequestPDU,
conclude-ResponsePDU	[12] IMPLICIT Conclude-ResponsePDU,
conclude-ErrorPDU	[13] IMPLICIT Conclude-ErrorPDU }

4.2 Confirmed-RequestPDU

```
Confirmed-RequestPDU ::= SEQUENCE {
    invokeID          Unsigned32,
    IF (attachToEventCondition attachToSemaphore )
        listOfModifier    SEQUENCE OF Modifier OPTIONAL,
    ENDIF
    ConfirmedServiceRequest,
    IF ( csr cspi )
        [79] CS-Request-Detail OPTIONAL
        -- shall not be transmitted if value is the value
        -- of a tagged type derived from NULL
    ENDIF
}
```

4.3 Unconfirmed-PDU

```
Unconfirmed-PDU ::= SEQUENCE {
    UnconfirmedService,
    IF ( cspi )
        [79] CS-Unconfirmed-Detail OPTIONAL
        -- shall not be transmitted if value is the value
        -- of a tagged type derived from NULL
    ENDIF
}
```

4.4 Confirmed-ResponsePDU

```
Confirmed-ResponsePDU ::= SEQUENCE {
    invokeID          Unsigned32,
    ConfirmedServiceResponse,
    IF ( csr cspi )
        [79] CS-Response-Detail OPTIONAL
        -- shall not be transmitted if value is the value
        -- of a tagged type derived from NULL
    ENDIF
}
```

4.5 Confirmed-ErrorPDU

```
Confirmed-ErrorPDU ::= SEQUENCE {  
    invokeID                [0] IMPLICIT Unsigned32,  
    IF (attachToEventCondition attachToSemaphore )  
        modifierPosition    [1] IMPLICIT Unsigned32 OPTIONAL,  
    ENDIF  
    serviceError            [2] IMPLICIT ServiceError }
```

4.6 Modifier

```
Modifier ::= CHOICE {  
    IF ( attachToEventCondition )  
        attachToEventCondition [0] IMPLICIT AttachToEventCondition,  
    ENDIF  
    IF ( attachToSemaphore )  
        attachToSemaphore      [1] IMPLICIT AttachToSemaphore  
    ENDIF  
}
```

4.7 ConfirmedServiceRequest

```
ConfirmedServiceRequest ::= CHOICE {  
    IF ( status )  
        status                [0] IMPLICIT Status-Request,  
    ENDIF  
    IF ( getNameList )  
        getNameList           [1] IMPLICIT GetNameList-Request,  
    ENDIF  
    IF ( identify )  
        identify              [2] IMPLICIT Identify-Request,  
    ENDIF  
    IF ( rename )  
        rename                [3] IMPLICIT Rename-Request,  
    ENDIF  
    IF ( read )  
        read                  [4] IMPLICIT Read-Request,  
    ENDIF  
    IF ( write )  
        write                  [5] IMPLICIT Write-Request,  
    ENDIF  
    IF ( vnam vadr )  
        IF ( getVariableAccessAttributes )  
            getVariableAccessAttributes [6] GetVariableAccessAttributes-Request,  
        ENDIF  
    ENDIF  
    IF ( vnam )  
    IF ( vadr )
```

```
IF ( defineNamedVariable )
    defineNamedVariable      [7] IMPLICIT DefineNamedVariable-Request,
ENDIF
ENDIF
ENDIF
IF ( vsca )
IF ( vnam vadr )
IF ( defineScatteredAccess )
    defineScatteredAccess    [8] IMPLICIT DefineScatteredAccess-Request,
ENDIF
IF ( getScatteredAccessAttributes )
    getScatteredAccessAttributes [9] GetScatteredAccessAttributes-Request,
ENDIF
ENDIF
ENDIF
IF ( vsca vnam )
IF ( deleteVariableAccess )
    deleteVariableAccess     [10] IMPLICIT DeleteVariableAccess-Request,
ENDIF
ENDIF
IF ( vlis )
IF ( vnam )
IF ( defineNamedVariableList )
    defineNamedVariableList   [11] IMPLICIT DefineNamedVariableList-Request,
ENDIF
IF ( getNamedVariableListAttributes )
    getNamedVariableListAttributes [12] GetNamedVariableListAttributes-Request,
ENDIF
IF ( deleteNamedVariableList )
    deleteNamedVariableList    [13] IMPLICIT DeleteNamedVariableList-Request,
ENDIF
ENDIF
ENDIF
IF ( vnam )
IF ( defineNamedType )
    defineNamedType           [14] IMPLICIT DefineNamedType-Request,
ENDIF
IF ( getNamedTypeAttributes )
    getNamedTypeAttributes     [15] GetNamedTypeAttributes-Request,
ENDIF
IF ( deleteNamedType )
    deleteNamedType            [16] IMPLICIT DeleteNamedType-Request,
ENDIF
ENDIF
IF ( input )
    input                      [17] IMPLICIT Input-Request,
ENDIF
IF ( output )
    output                     [18] IMPLICIT Output-Request,
```

```
ENDIF
IF ( takeControl )
    takeControl [19] IMPLICIT TakeControl-Request,
ENDIF
IF ( relinquishControl )
    relinquishControl [20] IMPLICIT RelinquishControl-Request,
ENDIF
IF ( defineSemaphore )
    defineSemaphore [21] IMPLICIT DefineSemaphore-Request,
ENDIF
IF ( deleteSemaphore )
    deleteSemaphore [22] DeleteSemaphore-Request,
ENDIF
IF ( reportSemaphoreStatus )
    reportSemaphoreStatus [23] ReportSemaphoreStatus-Request,
ENDIF
IF ( reportPoolSemaphoreStatus )
    reportPoolSemaphoreStatus [24] IMPLICIT ReportPoolSemaphoreStatus-Request,
ENDIF
IF ( reportSemaphoreEntryStatus )
    reportSemaphoreEntryStatus [25] IMPLICIT ReportSemaphoreEntryStatus-Request,
ENDIF
IF ( initiateDownloadSequence )
    initiateDownloadSequence [26] IMPLICIT InitiateDownloadSequence-Request,
    downloadSegment [27] IMPLICIT DownloadSegment-Request,
    terminateDownloadSequence [28] IMPLICIT TerminateDownloadSequence-Request,
ENDIF
IF ( initiateUploadSequence )
    initiateUploadSequence [29] IMPLICIT InitiateUploadSequence-Request,
    uploadSegment [30] IMPLICIT UploadSegment-Request,
    terminateUploadSequence [31] IMPLICIT TerminateUploadSequence-Request,
ENDIF
IF ( requestDomainDownload )
    requestDomainDownload [32] IMPLICIT RequestDomainDownload-Request.
ENDIF
IF ( requestDomainUpload )
    requestDomainUpload [33] IMPLICIT RequestDomainUpload-Request,
ENDIF
IF ( loadDomainContent )
    loadDomainContent [34] IMPLICIT LoadDomainContent-Request,
ENDIF
IF ( storeDomainContent )
    storeDomainContent [35] IMPLICIT StoreDomainContent-Request,
ENDIF
IF ( deleteDomain )
    deleteDomain [36] IMPLICIT DeleteDomain-Request,
ENDIF
IF ( getDomainAttributes )
    getDomainAttributes [37] IMPLICIT GetDomainAttributes-Request,
```

```
ENDIF
IF ( createProgramInvocation )
    createProgramInvocation [38] IMPLICIT CreateProgramInvocation-Request,
ENDIF
IF ( deleteProgramInvocation )
    deleteProgramInvocation [39] IMPLICIT DeleteProgramInvocation-Request,
ENDIF
IF ( start )
    start [40] IMPLICIT Start-Request,
ENDIF
IF ( stop )
    stop [41] IMPLICIT Stop-Request,
ENDIF
IF ( resume )
    resume [42] IMPLICIT Resume-Request,
ENDIF
IF ( reset )
    reset [43] IMPLICIT Reset-Request,
ENDIF
IF ( kill )
    kill [44] IMPLICIT Kill-Request,
ENDIF
IF ( getProgramInvocationAttributes )
    getProgramInvocationAttributes [45] IMPLICIT
        GetProgramInvocationAttributes-Request,
ENDIF
IF ( obtainFile )
    obtainFile [46] IMPLICIT ObtainFile-Request,
ENDIF
IF ( defineEventCondition )
    defineEventCondition [47] IMPLICIT DefineEventCondition-Request,
ENDIF
IF ( deleteEventCondition )
    deleteEventCondition [48] DeleteEventCondition-Request,
ENDIF
IF ( getEventConditionAttributes )
    getEventConditionAttributes [49] GetEventConditionAttributes-Request,
ENDIF
IF ( reportEventConditionStatus )
    reportEventConditionStatus [50] ReportEventConditionStatus-Request,
ENDIF
IF ( alterEventConditionMonitoring )
    alterEventConditionMonitoring [51] IMPLICIT
        AlterEventConditionMonitoring-Request,
ENDIF
IF ( triggerEvent )
    triggerEvent [52] IMPLICIT TriggerEvent-Request,
ENDIF
IF ( defineEventAction )
```

```
    defineEventAction          [53] IMPLICIT DefineEventAction-Request,
ENDIF
IF ( deleteEventAction )
    deleteEventAction          [54] DeleteEventAction-Request,
ENDIF
IF ( getEventActionAttributes )
    getEventActionAttributes    [55] GetEventActionAttributes-Request,
ENDIF
IF ( reportEventActionStatus )
    reportEventActionStatus     [56] ReportEventActionStatus-Request,
ENDIF
IF ( defineEventEnrollment )
    defineEventEnrollment       [57] IMPLICIT DefineEventEnrollment-Request,
ENDIF
IF ( deleteEventEnrollment )
    deleteEventEnrollment       [58] DeleteEventEnrollment-Request,
ENDIF
IF ( alterEventEnrollment )
    alterEventEnrollment        [59] IMPLICIT AlterEventEnrollment-Request,
ENDIF
IF ( reportEventEnrollmentStatus )
    reportEventEnrollmentStatus [60] ReportEventEnrollmentStatus-Request,
ENDIF
IF ( getEventEnrollmentAttributes )
    getEventEnrollmentAttributes [61] IMPLICIT
                                GetEventEnrollmentAttributes-Request,
ENDIF
IF ( acknowledgeEventNotification )
    acknowledgeEventNotification [62] IMPLICIT
                                AcknowledgeEventNotification-Request,
ENDIF
IF ( getAlarmSummary )
    getAlarmSummary             [63] IMPLICIT GetAlarmSummary-Request,
ENDIF
IF ( getAlarmEnrollmentSummary )
    getAlarmEnrollmentSummary   [64] IMPLICIT GetAlarmEnrollmentSummary-Request,
ENDIF
IF ( readJournal )
    readJournal                 [65] IMPLICIT ReadJournal-Request,
ENDIF
IF ( writeJournal )
    writeJournal                [66] IMPLICIT WriteJournal-Request,
ENDIF
IF ( initializeJournal )
    initializeJournal           [67] IMPLICIT InitializeJournal-Request,
ENDIF
IF ( reportJournalStatus )
    reportJournalStatus         [68] ReportJournalStatus-Request,
ENDIF
```

```

IF ( createJournal )
    createJournal          [69] IMPLICIT CreateJournal-Request,
ENDIF
IF ( deleteJournal )
    deleteJournal         [70] IMPLICIT DeleteJournal-Request,
ENDIF
IF ( getCapabilityList )
    getCapabilityList      [71] IMPLICIT GetCapabilityList-Request,
ENDIF
IF ( fileOpen )
    fileOpen               [72] IMPLICIT FileOpen-Request,
ENDIF
IF ( fileRead )
    fileRead               [73] IMPLICIT FileRead-Request,
ENDIF
IF ( fileClose )
    fileClose              [74] IMPLICIT FileClose-Request,
ENDIF
IF ( fileRename )
    fileRename             [75] IMPLICIT FileRename-Request,
ENDIF
IF ( fileDelete )
    fileDelete             [76] IMPLICIT FileDelete-Request,
ENDIF
IF ( fileDirectory )
    fileDirectory          [77] IMPLICIT FileDirectory-Request,
ENDIF
IF ( csr cspi )
    additionalService      [78] AdditionalService-Request,
ENDIF
-- choices [72] through [77] are reserved for use by services
-- defined in annex C
-- choice [79] is reserved
IF ( getDataExchangeAttributes )
    getDataExchangeAttributes [80] GetDataExchangeAttributes-Request,
    -- Shall not appear in minor version 1
ENDIF
IF ( exchangeData )
    exchangeData           [81] IMPLICIT ExchangeData-Request,
    -- Shall not appear in minor version 1
ENDIF
IF ( defineAccessControllist )
    defineAccessControllist [82] IMPLICIT
    DefineAccessControllist-Request,
    -- Shall not appear in minor version 1 or y
ENDIF
IF ( getAccessControllistAttributes )
    getAccessControllistAttributes [83] IMPLICIT
    GetAccessControllistAttributes-Request,

```

```

        -- Shall not appear in minor version 1 or y
ENDIF
IF ( reportAccessControlledObjects )
    reportAccessControlledObjects    [84] IMPLICIT
        ReportAccessControlledObjects-Request,
        -- Shall not appear in minor version 1 or y
ENDIF
IF ( deleteAccessControlList )
    deleteAccessControlList    [85] IMPLICIT
        DeleteAccessControlList-Request,
        -- Shall not appear in minor version 1 or y
ENDIF
IF ( changeAccessControl )
    changeAccessControl          [86] IMPLICIT
        ChangeAccessControl-Request
        -- Shall not appear in minor version 1 or y
ENDIF
}

```

4.8 AdditionalService-Request

```

AdditionalService-Request ::= CHOICE {
    IF ( csr )
    IF ( vMDStop )
        vMDStop                [0] IMPLICIT VMDStop-Request,
    ENDIF
    IF ( vMDReset )
        vMDReset                [1] IMPLICIT VMDReset-Request,
    ENDIF
    IF ( select )
        select                  [2] IMPLICIT Select-Request,
    ENDIF
    IF ( alterProgramInvocationAttributes )
        alterPI                 [3] IMPLICIT AlterProgramInvocationAttributes-Request,
    ENDIF
    IF ( cspi )
    IF ( initiateUnitControlLoad )
        initiateUCLoad          [4] IMPLICIT InitiateUnitControlLoad-Request,
    ENDIF
    IF ( unitControlLoadSegment )
        uCLoad                  [5] IMPLICIT UnitControlLoadSegment-Request,
    ENDIF
    IF ( unitControlUpload )
        uCUpload                [6] IMPLICIT UnitControlUpload-Request,
    ENDIF
    IF ( startUnitControl )
        startUC                  [7] IMPLICIT StartUnitControl-Request,
    ENDIF
}

```

```
IF ( stopUnitControl )
    stopUC [8] IMPLICIT StopUnitControl-Request,
ENDIF
IF ( createUnitControl )
    createUC [9] IMPLICIT CreateUnitControl-Request,
ENDIF
IF ( addToUnitControl )
    addToUC [10] IMPLICIT AddToUnitControl-Request,
ENDIF
IF ( removeFromUnitControl )
    removeFromUC [11] IMPLICIT RemoveFromUnitControl-Request,
ENDIF
IF ( getUnitControlAttributes )
    getUCAAttributes [12] IMPLICIT GetUnitControlAttributes-Request,
ENDIF
IF ( loadUnitControlFromFile )
    loadUCFromFile [13] IMPLICIT LoadUnitControlFromFile-Request,
ENDIF
IF ( storeUnitControlToFile )
    storeUCToFile [14] IMPLICIT StoreUnitControlToFile-Request,
ENDIF
IF ( deleteUnitControl )
    deleteUC [15] IMPLICIT DeleteUnitControl-Request,
ENDIF
IF ( defineEventConditionList )
    defineECL [16] IMPLICIT DefineEventConditionList-Request,
ENDIF
IF ( deleteEventConditionList )
    deleteECL [17] IMPLICIT DeleteEventConditionList-Request,
ENDIF
IF ( addEventConditionListReference )
    addECLReference [18] IMPLICIT AddEventConditionListReference-Request,
ENDIF
IF ( removeEventConditionListReference )
    removeECLReference [19] IMPLICIT RemoveEventConditionListReference-Request,
ENDIF
IF ( getEventConditionListAttributes )
    getECLAttributes [20] IMPLICIT GetEventConditionListAttributes-Request,
ENDIF
IF ( reportEventConditionListStatus )
    reportECLStatus [21] IMPLICIT ReportEventConditionListStatus-Request,
ENDIF
IF ( alterEventConditionListMonitoring )
    alterECLMonitoring [22] IMPLICIT AlterEventConditionListMonitoring-Request
ENDIF
ENDIF
}
```

4.9 CS-Request-Detail

```

CS-Request-Detail ::= CHOICE {
otherRequests          NULL,
    -- this choice shall be selected if the tag value of the
    -- ConfirmedServiceRequest does not match any of the tags below
IF ( createProgramInvocation )
    createProgramInvocation [38] IMPLICIT
        CS-CreateProgramInvocation-Request,
ENDIF
IF ( start )
    start [40] IMPLICIT CS-Start-Request,
ENDIF
IF ( resume )
    resume [42] IMPLICIT CS-Resume-Request,
ENDIF
IF ( defineEventCondition )
    defineEventCondition [47] IMPLICIT CS-DefineEventCondition-Request,
ENDIF
IF ( alterEventConditionMonitoring )
    alterEventConditionMonitoring [51] IMPLICIT
        CS-AlterEventConditionMonitoring-Request,
ENDIF
IF ( defineEventEnrollment )
    defineEventEnrollment [57] IMPLICIT CS-DefineEventEnrollment-Request,
ENDIF
IF ( alterEventEnrollment )
    alterEventEnrollment [59] IMPLICIT CS-AlterEventEnrollment-Request
ENDIF
}

```

4.10 UnconfirmedService

```

UnconfirmedService ::= CHOICE {
IF (informationReport )
    informationReport [0] IMPLICIT InformationReport,
ENDIF
IF ( unsolicitedStatus )
    unsolicitedStatus [1] IMPLICIT UnsolicitedStatus,
ENDIF
IF ( eventNotification )
    eventNotification [2] IMPLICIT EventNotification,
ENDIF
IF ( )
    additionalService [3] AdditionalUnconfirmedService
ENDIF
}

```

```

CS-Unconfirmed-Detail ::= CHOICE {
  IF ( cspi )
    eventNotification          [2] IMPLICIT CS-EventNotification,
  ENDIF
  otherRequests               NULL
    -- this choice shall be selected if the tag value of the
    -- UnconfirmedService does not match any of the tags above
}

```

```

ConfirmedServiceResponse ::= CHOICE {
  IF ( status )
    status [0] IMPLICIT Status-Response,
  ENDIF
  IF ( getNameList )
    getNameList [1] IMPLICIT GetNameList-Response,
  ENDIF
  IF ( identify )
    identify [2] IMPLICIT Identify-Response,
  ENDIF
  IF ( rename )
    rename [3] IMPLICIT Rename-Response,
  ENDIF
  IF ( read )
    read [4] IMPLICIT Read-Response,
  ENDIF
  IF ( write )
    write [5] IMPLICIT Write-Response,
  ENDIF
  IF ( vnam vadr )
    IF ( getVariableAccessAttributes )
      getVariableAccessAttributes [6] IMPLICIT
        GetVariableAccessAttributes-Response,
    ENDIF
  ENDIF
  IF ( vnam )
  IF ( vadr )
  IF ( defineNamedVariable )
    defineNamedVariable [7] IMPLICIT DefineNamedVariable-Response,
  ENDIF
  ENDIF
  ENDIF
  IF ( vsca )
  IF ( vnam vadr )
  IF ( defineScatteredAccess )
    defineScatteredAccess [8] IMPLICIT DefineScatteredAccess-Response,

```

```
ENDIF
IF ( getScatteredAccessAttributes )
    getScatteredAccessAttributes [9] IMPLICIT
        GetScatteredAccessAttributes-Response,
ENDIF
ENDIF
ENDIF
IF ( vsca vnam )
IF ( deleteVariableAccess )
    deleteVariableAccess [10] IMPLICIT DeleteVariableAccess-Response,
ENDIF
ENDIF
IF ( vlis )
IF ( vnam )
IF ( defineNamedVariableList )
    defineNamedVariableList [11] IMPLICIT DefineNamedVariableList-Response,
ENDIF
IF ( getNamedVariableListAttributes )
    getNamedVariableListAttributes [12] IMPLICIT
        GetNamedVariableListAttributes-Response,
ENDIF
IF ( deleteNamedVariableList )
    deleteNamedVariableList [13] IMPLICIT DeleteNamedVariableList-Response,
ENDIF
ENDIF
ENDIF
IF ( vnam )
IF ( defineNamedType )
    defineNamedType [14] IMPLICIT DefineNamedType-Response,
ENDIF
IF ( getNamedTypeAttributes )
    getNamedTypeAttributes [15] IMPLICIT GetNamedTypeAttributes-Response,
ENDIF
IF ( deleteNamedType )
    deleteNamedType [16] IMPLICIT DeleteNamedType-Response,
ENDIF
ENDIF
IF ( input )
    input [17] IMPLICIT Input-Response,
ENDIF
IF ( output )
    output [18] IMPLICIT Output-Response,
ENDIF
IF ( takeControl )
    takeControl [19] TakeControl-Response,
ENDIF
IF ( relinquishControl )
    relinquishControl [20] IMPLICIT RelinquishControl-Response,
ENDIF
```

```
IF ( defineSemaphore )
    defineSemaphore          [21] IMPLICIT DefineSemaphore-Response,
ENDIF
IF ( deleteSemaphore )
    deleteSemaphore         [22] IMPLICIT DeleteSemaphore-Response,
ENDIF
IF ( reportSemaphoreStatus )
    reportSemaphoreStatus    [23] IMPLICIT ReportSemaphoreStatus-Response,
ENDIF
IF ( reportPoolSemaphoreStatus )
    reportPoolSemaphoreStatus [24] IMPLICIT ReportPoolSemaphoreStatus-Response,
ENDIF
IF ( reportSemaphoreEntryStatus )
    reportSemaphoreEntryStatus [25] IMPLICIT ReportSemaphoreEntryStatus-Response,
ENDIF
IF ( initiateDownloadSequence )
    initiateDownloadSequence [26] IMPLICIT InitiateDownloadSequence-Response,
    downloadSegment          [27] IMPLICIT DownloadSegment-Response,
    terminateDownloadSequence [28] IMPLICIT
        TerminateDownloadSequence-Response,
ENDIF
IF ( initiateUploadSequence )
    initiateUploadSequence    [29] IMPLICIT InitiateUploadSequence-Response,
    uploadSegment             [30] IMPLICIT UploadSegment-Response,
    terminateUploadSequence    [31] IMPLICIT TerminateUploadSequence-Response,
ENDIF
IF ( requestDomainDownload )
    requestDomainDownload     [32] IMPLICIT RequestDomainDownload-Response,
ENDIF
IF ( requestDomainUpload )
    requestDomainUpload       [33] IMPLICIT RequestDomainUpload-Response,
ENDIF
IF ( loadDomainContent )
    loadDomainContent          [34] IMPLICIT LoadDomainContent-Response,
ENDIF
IF ( storeDomainContent )
    storeDomainContent         [35] IMPLICIT StoreDomainContent-Response,
ENDIF
IF ( deleteDomain )
    deleteDomain              [36] IMPLICIT DeleteDomain-Response,
ENDIF
IF ( getDomainAttributes )
    getDomainAttributes        [37] IMPLICIT GetDomainAttributes-Response,
ENDIF
IF ( createProgramInvocation )
    createProgramInvocation    [38] IMPLICIT CreateProgramInvocation-Response,
ENDIF
IF ( deleteProgramInvocation )
    deleteProgramInvocation    [39] IMPLICIT DeleteProgramInvocation-Response,
```

```

ENDIF
IF ( start )
    start [40] IMPLICIT Start-Response,
ENDIF
IF ( stop )
    stop [41] IMPLICIT Stop-Response,
ENDIF
IF ( resume )
    resume [42] IMPLICIT Resume-Response,
ENDIF
IF ( reset )
    reset [43] IMPLICIT Reset-Response,
ENDIF
IF ( kill )
    kill [44] IMPLICIT Kill-Response,
ENDIF
IF ( getProgramInvocationAttributes )
    getProgramInvocationAttributes [45] IMPLICIT
        GetProgramInvocationAttributes-Response,
ENDIF
IF ( obtainFile )
    obtainFile [46] IMPLICIT ObtainFile-Response,
ENDIF
IF ( defineEventCondition )
    defineEventCondition [47] IMPLICIT DefineEventCondition-Response,
ENDIF
IF ( deleteEventCondition )
    deleteEventCondition [48] IMPLICIT DeleteEventCondition-Response,
ENDIF
IF ( getEventConditionAttributes )
    getEventConditionAttributes [49] IMPLICIT
        GetEventConditionAttributes-Response,
ENDIF
IF ( reportEventConditionStatus )
    reportEventConditionStatus [50] IMPLICIT
        ReportEventConditionStatus-Response,
ENDIF
IF ( alterEventConditionMonitoring )
    alterEventConditionMonitoring [51] IMPLICIT
        AlterEventConditionMonitoring-Response,
ENDIF
IF ( triggerEvent )
    triggerEvent [52] IMPLICIT TriggerEvent-Response,
ENDIF
IF ( defineEventAction )
    defineEventAction [53] IMPLICIT DefineEventAction-Response,
ENDIF
IF ( deleteEventAction )
    deleteEventAction [54] IMPLICIT DeleteEventAction-Response,

```