Stable Implementation Agreements for Open Systems Interconnection Protocols: Part 20 - Manufacturing Message Specification (MMS)

Output from the December 1990 NIST Workshop for Implementors of OSI

SIG Chair: **Herbert Falk**
SIG Editor: **Neal Laurance**

Foreword

This part of the Stable Implementation Agreements was prepared by the Manufacturing Message Specification (MMS) Special Interest Group (MMSSIG) of the National Institute of Standards and Technology (NIST) Workshop for Implementors of Open Systems Interconnection (OSI). See Procedures Manual for Workshop charter.

Text in this part has been approved by the Plenary of the above-mentioned Workshop. Significant technical change has occurred in this part since it was previously presented.

Future changes and additions to this version of these Implementor Agreements will be published as change pages. Deleted and replaced text will be shown as strikeout. New and replacement text will be shown as shaded.

Table of Contents

Part 20 - Manufacturing Message Specification (MMS) 1

0 Introduction 1

1 Scope 1

2 Field of Application 1
2.1 General 1
2.2 Phase 1 agreements 1

3 Normative References 2

4 Definitions 3

5 Corrigenda and addenda 3

6 Status 3

7 General agreements 3
7.1 Max supported PDU size 3
7.2 FileName 4

8 Service-specific agreements 4
8.1 Environment and general management 4
8.1.1 Initiate 4
8.1.1.1 Negotiation of MMS abstract syntaxes 4
8.1.1.2 Max serv outstanding 4
8.1.1.3 Local detail calling 5
8.1.1.4 Local detail called 5
8.2 VMD support 5
8.3 Domain management 6
8.3.1 List of capabilities 6
8.3.2 Initiate Download Sequence service 6
8.3.3 Download Segment service 6
8.3.4 Terminate Download Sequence service 6
8.3.5 Initiate Upload Sequence service 6
8.3.6 Upload Segment service 6
8.3.7 Get Domain Attributes service 7
8.3.8 Get Capability List service 7
8.4 Program Invocation management 7
8.4.1 Start service 7
8.4.2 Stop service 7
8.4.3 Resume service 7
8.4.4 Reset service 7
8.5 Variable access 7

8.5.1 Scattered access . . . 7
8.5.2 Floating point . . . 8
8.6 Semaphore management . . . 8
8.7 Operator communication . . . 8
8.8 Event management . . . 8
8.9 Journal management . . . 8

Annex A (normative)

Backwards compatibility agreements . . . 9
A.1 Introduction . . . 9
A.2 Backwards compatibility agreements for calling MMS implementations . . . 10
A.3 Backwards compatibility agreements for called MMS implementations . . . 10
A.4 General backwards compatibility agreements . . . 11
A.4.1 VMD logical status . . . 11

Annex B (normative)

DIS 9506 Modifications Required for Backwards Compatibility . . . 12
B.1 Introduction . . . 12
B.2 References . . . 12
B.3 General . . . 12
B.3.1 Implementation base . . . 12
B.3.2 Rules of extensibility . . . 12
B.4 Modifications to the protocol definitions . . . 12
B.4.1 Page 39, Section 7.5.2 of DIS 9506–2 . . . 12
B.4.2 Page 49, Section 7.6.4, DIS 9506–2 . . . 13
B.4.3 Page 95, Section 12.2.1 of DIS 9506–2 . . . 13
B.4.4 Page 96, Section 12.3.1 of DIS 9506–2 . . . 13
B.4.5 Page 98, Section 12.4.2 of DIS 9506–2 . . . 14
B.4.6 Page 138, Section 15.14 of DIS 9506–2 . . . 14
B.4.7 Page 166, Section 17.10 of DIS 9506–2 . . . 14
B.5 Behavioral requirements . . . 15
B.5.1 Filenames . . . 15
B.5.2 Identify service . . . 15
B.5.3 Initiate service . . . 15
B.5.3.1 Minimum segment size . . . 15
B.5.3.2 Maximum segment size . . . 15
B.5.4 Abstract syntax name . . . 16
B.5.5 Application context name . . . 16
B.5.6 Minor version number . . . 16
B.6 Parameter CBB subset . . . 16
B.7 Service subset . . . 17

List of Tables

Table 1 - Phase 1 Services 2
Table 2 - MMS Service Subset 17

Part 20 - Manufacturing Message Specification (MMS)

0 Introduction

This section defines Implementors Agreements based on Manufacturing Message Specification (MMS), as defined in ISO/IEC 9506. ISO/IEC 9506 has two parts. Part 1 defines the Virtual Manufacturing Device (VMD), its subordinate abstract objects, and the services on these objects. Part 2 defines the protocol. Future parts may define companion standards.

MMS, as described in ISO/IEC 9506, is based on the following ISO documents: ACSE Service and Protocol (ISO 8649, ISO 8650), Presentation Service and Protocol (ISO 8822, ISO 8823), ASN.1 Abstract Syntax Notation and Basic Encoding Rules (ISO 8824, ISO 8825), and Session Service and Protocol (ISO 8326, ISO 8327). These services and protocols are defined architecturally in the OSI Reference Model (ISO 7498). These agreements provide detailed guidance for the implementor, and eliminate ambiguities in interpretations.

An A-Profile based on MMS and these agreements can be used over any T-Profile (see ISO TR 10000) specifying the OSI connection-mode transport service, or the transport profiles used in support of MAP (Manufacturing Automation Protocol), TOP (Technical and Office Protocols), or US GOSIP.

1 Scope

2 Field of Application

2.1 General

Work on implementation agreements will proceed in phases based upon grouping of services and/or contexts. Implementations are not constrained from implementing services or contexts not addressed by the current set of stable agreements. Future phases of work may affect such implementations.

2.2 Phase 1 agreements

These agreements will be based on a subset of MMS services and protocol listed in table 1 and defined in ISO/IEC 9506-1 and ISO/IEC 9506-2.

Table 1 - Phase 1 Services

Initiate
Conclude
Reject
Abort
Status
GetNameList
Identify
UnsolicitedStatus
GetCapabilityList
InitiateDownloadSequence
DownloadSegment
TerminateDownloadSequence
InitiateUploadSequence
UploadSegment
TerminateUploadSequence
DeleteDomain
GetDomainAttributes
Read
Write
InformationReport
GetVariableAccessAttributes
Input
Output
CreateProgramInvocation
DeleteProgramInvocation
Start
Stop
Resume
Reset
Kill
GetProgramInvocationAttributes

3 Normative References

[**1**] ISO/IEC 9506-1: 1990 - *Industrial automation systems - Manufacturing Message Specification Part 1: Service definition*

[**2**] ISO/IEC 9506-2: 1990 - *Industrial automation systems - Manufacturing Message Specification Part 2: Protocol specification*

4 Definitions

The definitions given in ISO/IEC 9506-1 are applicable to this document.

In addition the following definitions are used in this document:

MMS Implementation a term used to describe a system which conforms to ISO/IEC 9506, acting either as client or server, when it is unnecessary to distinguish between the MMS-user and the MMS-provider.

MMSpdu Any valid value of the MMSpdu abstract data type as defined in Clause 7 of ISO/IEC 9506-2, except for the initiate-RequestPDU, initiate-ResponsePDU, and initiate-ErrorPDU choices, encoded in the negotiated transfer syntax.

5 Corrigenda and addenda

(Refer to Working Agreements, Dated December, 1990.)

6 Status

(Refer to Working Agreements, Dated December, 1990.)

7 General agreements

7.1 Max supported PDU size

The max_mms_pdu_size is defined as the maximum number of octets in an MMSpdu encoded using the negotiated transfer syntax. This size shall apply to all MMSpdu's with the exception of the initiate-Request PDU, initiate-Response PDU, and initiate-Error PDU. The max_mms_pdu_size shall be negotiated during connection initiation using the Local Detail Calling and Local Detail Called parameters of the MMS initiate service.

The negotiated max_mms_pdu_size shall be applied as follows:

a) Any received MMSpdu whose length is less than or equal to the negotiated max_mms_pdu_size shall be properly parsed and processed.

b) An MMS implementation should not send an MMSpdu whose size exceeds the negotiated max_mms_pdu_size. If an MMS implementation sends an MMSpdu that exceeds the negotiated max_mms_pdu_size, then it shall be prepared to receive a reject pdu. Should an MMS implementation receive an MMSpdu that exceeds the negotiated max_mms_pdu_size, it shall either reject the MMSpdu or accept the MMSpdu as if no size violation had occurred and perform the expected processing.

c) If an MMS implementation is unable to send a service response because the response would exceed the max_mms_pdu_size, then it shall return a Service response (-) with an error class of SERVICE and an error code of OTHER.

d) When rejecting an MMSpdu because it exceeds the negotiated max_mms_pdu_size, an MMS implementation shall use a Reject PDU Type of PDU-ERROR and a Reject Code of INVALID-PDU in the resulting reject pdu.

7.2 FileName

Restrictions for the use of the type FileName in the MMS Abstract Syntax are specified in section 9.1 of part 9 of these agreements.

8 Service-specific agreements

8.1 Environment and general management

8.1.1 Initiate

8.1.1.1 Negotiation of MMS abstract syntaxes

On the A-ASSOCIATE response, the MMS responder shall not accept more than one presentation context derived from an MMS abstract syntax. For this agreement, the term 'MMS abstract syntax' shall represent an abstract syntax from the set containing the abstract syntax defined in clause 19 of ISO/IEC 9506-2 and abstract syntaxes defined by MMS companion standards.

NOTE - There are technical problems with describing operation in multiple MMS abstract syntaxes over a single association. These problems have been identified as of 9/90, and form the basis of the prior agreement.

8.1.1.2 Max serv outstanding

An MMS Implementation which intends to conform only with the Client Conformance Requirements for Requester CBBs shall:

a) propose one or greater for the value of the Proposed Max Serv Outstanding Called parameter in the Initiate service when initiating the application association (calling);

b) offer one or greater for the value of the Negotiated Max Serv Outstanding Calling parameter in the Initiate service when receiving the application association initiation (called).

An MMS Implementation which intends to conform to one or more Server Conformance Requirements for Responder CBBs shall:

a) propose one or greater for the value of the Proposed Max Serv Outstanding Calling parameter in the Initiate service when initiating the application association (calling);

b) offer one or greater for the value of the Negotiated Max Serv Outstanding Called parameter in the Initiate service when receiving the application association initiation (called).

8.1.1.3 Local detail calling

The local detail calling parameter in the initiate request primitive shall specify the max_mms_pdu_size guaranteed to be supported by the calling MMS implementation. If the local detail calling parameter is absent from the request primitive, then the calling MMS implementation guarantees support for an unlimited max_mms_pdu_size.

If present in the request or indication primitives, the local detail calling parameter shall not be less than 64; however, it is recommended that at least 512 octets be supported.

8.1.1.4 Local detail called

The local detail called parameter in the initiate response shall specify the negotiated max_mms_pdu_size for the application association.

If the local detail calling parameter was omitted in the indication primitive, then the local detail called parameter:

a) may be omitted from the response, indicating that the calling MMS implementation and the called MMS implementation are prepared to support an unbounded max_mms_pdu_size;

b) may be specified in the response, indicating a requirement to support the specified value for max_mms_pdu_size.

If the local detail calling parameter was included in the request, then this parameter shall be present in the response and its value shall be less than or equal to the value of the local detail calling parameter of the request.

If present in the response, the local detail called parameter shall not be less than 64; however, it is recommended that at least 512 octets be supported.

8.2 VMD support

8.3 Domain management

8.3.1 List of capabilities

Only one capability shall be described in each Visible String of the SEQUENCE OF.

8.3.2 Initiate Download Sequence service

The List of Capability parameter shall follow the limitations of 8.3.1.

The syntax and semantics of the capabilities shall be defined by the Server in the PICS. Any deviation from the defined syntax and semantics shall be grounds for the Server to return a service error with Error Class = RESOURCE and Error Code = CAPABILITY-UNKNOWN.

8.3.3 Download Segment service

A client that receives a Download Segment indication after issuing a Download Segment Result(+) with the MoreFollows parameter equal to FALSE or after issuing a Download Segment Result(-) shall issue either a service error, specifying an Error Class = SERVICE and an Error Code = PRIMITIVES-OUT-OF-SEQUENCE, or an Abort request.

8.3.4 Terminate Download Sequence service

If a client receives a Terminate Download Sequence indication in which the Discard parameter is absent and the client has not issued a Download Segment response with the More Follows parameter = FALSE for that Domain, it shall behave as if it had received a Terminate Download Sequence indication with the Discard parameter present with error class = VMD-STATE and error code = DOMAIN-TRANSFER-PROBLEM. It is then up to the client application to determine the true state of the Domain and take any recovery action.

8.3.5 Initiate Upload Sequence service

The List of Capability parameter shall follow the limitations of 8.3.1.

8.3.6 Upload Segment service

A server that receives an Upload Segment indication for an Upload State Machine for which it has issued an Upload Segment Result(-) or an Upload Segment Result(+) with the MoreFollows parameter equal to FALSE, shall issue either a service error, specifying an Error Class = SERVICE and an Error Code = PRIMITIVES-OUT-OF-SEQUENCE, or an Abort request.

8.3.7 Get Domain Attributes service

The List of Capability parameter shall follow the limitations of 8.3.1.

8.3.8 Get Capability List service

The List of Capability parameter shall follow the limitations of 8.3.1.

8.4 Program Invocation management

8.4.1 Start service

A ProgramInvocationState of non-existent shall be returned in a Result(-) when a request to Start a non-existent Program Invocation is received.

8.4.2 Stop service

A ProgramInvocationState of non-existent shall be returned in a Result(-) when a request to Stop a non-existent Program Invocation is received.

8.4.3 Resume service

A ProgramInvocationState of non-existent shall be returned in a Result(-) when a request to Resume a non-existent Program Invocation is received.

8.4.4 Reset service

A ProgramInvocationState of non-existent shall be returned in a Result(-) when a request to Reset a non-existent Program Invocation is received.

8.5 Variable access

8.5.1 Scattered access

It is strongly recommended that for services which use variable access, a Variable List Name or List of Variable be used instead of Scattered Access.

No implementations shall be required to propose or accept the VSCA Parameter CBB.

8.5.2 Floating point

It is strongly recommended for services which use floating point types or values, that the choice of floating-point in the Data and TypeSpecification productions be used instead of the real choice.

No implementations shall be required to propose or accept the REAL parameter CBB.

8.6 Semaphore management

Semaphore services are not considered in Phase 1.

8.7 Operator communication

No Operator Communication agreements have been identified to date.

8.8 Event management

Event Management services are not considered in Phase 1.

8.9 Journal management

Journal Management services are not considered in Phase 1.

Annex A (normative)

Backwards compatibility agreements

A.1 Introduction

There is an installed base of real DIS 9506 based implementations. Providing support for application connectivity to both DIS and IS is desired as a migration strategy. These implementation agreements will allow IS based implementations to interoperate with DIS based implementations as described in ANNEX B. To achieve this backwards compatibility, the IS implementation shall support all of the agreements in this section.

It was found that the Abstract Syntax name object identifiers of both DIS and IS were identical. Therefore, the use of zero as the version number allows differentiation between an IS and a DIS based implementation. Since the abstract syntax name object identifier of any companion standard is different from that used by the DIS implementations, DIS implementations cannot interoperate with IS based implementations in a companion standard context.

NOTES

1 The value of zero is a valid value for this parameter in the DIS and not in the IS.

2 There are three types of implementations when considering MMS backwards compatibility.

IMP-1: An implementation based on DIS 9506 as described in Annex B;

IMP-2: An implementation based on IS 9506 with no backwards compatibility agreements applied;

IMP-3: An implementation based on IS 9506 which includes the backwards compatibility agreements of this annex. Such an implementation can dynamically negotiate to interoperate with an IMP-1, an IMP-2, or an IMP-3 implementation.

Since the value of the minor version number is zero for DIS-based implementations, and is one or greater for implementations of ISO/IEC 9506, this value can be used to differentiate between IMP-1 and IMP-2. An IMP-3 system can interoperate with either of these systems. If an IMP-3 is the Calling system, it will offer a value of one (or greater) for the proposed version number. An IMP-1 system will respond with a value of the negotiated version number of zero, using the negotiation procedure defined in ISO/IEC 9506. The IMP-3 system will accept this response. If the IMP-3 system is the Called system and has received an Initiate request with a value of zero for the proposed version number (from an IMP-1 system), it will respond with a value of zero for the negotiated version number. By following this procedure, an IMP-3 can interoperate with an IMP-2 or with another IMP-3 viewed as an IMP-2. After association context establishment, an IMP-3 system shall behave as either an IMP-1 or an IMP-2 system as appropriate on that particular association. The remainder of this section describes additional agreements which change an IMP-2 implementation into an IMP-3 implementation.

A.2 Backwards compatibility agreements for calling MMS implementations

A calling MMS implementation shall be capable of receiving and supporting a negotiatedVersionNumber parameter in the Initiate Service confirm of zero.

A calling MMS implementation which has received a negotiatedVersionNumber parameter in the Initiate Service confirm of zero shall support the modifications described in A.4.

A calling MMS implementation shall be capable of receiving an Application Context Name parameter value appropriate to an IMP-1 or IMP-2 in the A-Associate confirm.

A calling MMS implementation which has received a negotiatedVersionNumber of zero shall be capable of receiving and supporting an Initiate Response which has been encoded according to the modifications described in Appendix B, specifically the capability of receiving and supporting a negotiatedParameterCBB containing exactly 7 bits.

If a calling MMS implementation receives an Initiate confirm primitive with a negotiatedVersionNumber parameter equal to zero, the calling MMS implementation shall support the VLIS conformance building block if the implementation claims support for any service which contains one or more parameters which indicate the VLIS CBB in its service definition.

A.3 Backwards compatibility agreements for called MMS implementations

A called MMS implementation shall be capable of receiving and supporting a proposedVersionNumber parameter in the Initiate Service indication of zero.

A called MMS implementation which has received a proposedVersionNumber parameter in the Initiate Service indication of zero shall support the modifications in A.4.

A called MMS implementation shall be capable of receiving an Application Context Name parameter appropriate to an IMP-1 or IMP-2 in the A-Associate indication.

A called MMS implementation shall be capable of receiving and supporting an Initiate Request which has been encoded according to the modifications described in Appendix B, specifically the capability of receiving and supporting a proposedParameterCBB containing exactly 7 bits.

If a called MMS implementation receives an Initiate indication primitive with a proposedVersionNumber parameter equal to zero, the called MMS implementation shall support the VLIS conformance building block if the implementation claims support for any service which contains one or more parameters which indicate the VLIS CBB in its service definition.

A.4 General backwards compatibility agreements

A.4.1 VMD logical status

If the current VMD State is SUPPORT-SERVICES-ALLOWED and the association minor version number is zero, then the vmdLogicalStatus parameter shall have a value of STATE-CHANGES-ALLOWED in a Status response or in an unsolicitedStatus request.

Annex B (normative)

DIS 9506 Modifications Required for Backwards Compatibility

B.1 Introduction

This annex is an integral part of this part. It documents the modifications to DIS 9506 required to describe implementations for which the agreements of this part provide backwards compatibility. This annex as applied to DIS 9506 is referred to as Version 0.

B.2 References

[**1**] *MMS/1 Manufacturing Message Specification - ISO DIS 9506 - Service Definition*, December 1987

[**2**] *MMS/2 Manufacturing Message Specification - ISO DIS 9506 - Protocol Specification*, December 1987

[**3**] *NBS OSI Implementors Workshop Agreements* - December 1987

B.3 General

B.3.1 Implementation base

Version 0 is based upon Reference [3] in B.2 as it applies to MMS.

B.3.2 Rules of extensibility

The following sentence is appended to the last paragraph in section 8.2.1.1.5.2 Proposed Parameter CBB and the last paragraph in section 8.2.1.2.5.2 Negotiated Parameter CBB of DIS 9506-1.

"Any additional bits shall be ignored."

B.4 Modifications to the protocol definitions

B.4.1 Page 39, Section 7.5.2 of DIS 9506-2

```
CHANGE

 reportEventEnrollmentStatus   [60] IMPLICIT
                                    ReportEventEnrollmentStatus-Request,

TO

 reportEventEnrollmentStatus   [60] ReportEventEnrollmentStatus-Request,
```

B.4.2 Page 49, Section 7.6.4, DIS 9506-2

```
CHANGE

 ApplicationReference  ::= SEQUENCE {
   ap-title                ISO-8650-ACSE-1.AP-title OPTIONAL,
   ap-invocation-id        ISO-86 50-ACSE-1.AP-invocation-id OPTIONAL,
   ae-qualifier            ISO-8650-ACSE-1.AE-qualifier OPTIONAL,
   ae-invocation-id        ISO-8650-ACSE-1.AE-invocation-id OPTIONAL }

TO

 ApplicationReference  ::= SEQUENCE {
   ap-title                [0] OBJECT IDENTIFIER OPTIONAL,
   ap-invocation-id        [1] INTEGER OPTIONAL,
   ae-qualifier            [2] INTEGER OPTIONAL,
   ae-invocation-id        [3] INTEGER OPTIONAL }
```

B.4.3 Page 95, Section 12.2.1 of DIS 9506-2

```
CHANGE

 structure   [2] IMPLICIT SEQUENCE OF SEQUENCE {

TO

 structure   [2] IMPLICIT SEQUENCE {
```

B.4.4 Page 96, Section 12.3.1 of DIS 9506-2

CHANGE
named [4] IMPLICIT SEQUENCE {
TO
named [5] IMPLICIT SEQUENCE {

B.4.5 Page 98, Section 12.4.2 of DIS 9506-2

CHANGE
generalized-time [10] IMPLICIT GeneralizedTime,
TO
generalized-time [11] IMPLICIT GeneralizedTime,

B.4.6 Page 138, Section 15.14 of DIS 9506-2

CHANGE
additionalDetail [9] IMPLICIT EE-Additional-Detail OPTIONAL
TO
additionalDetail [9] EE-Additional-Detail OPTIONAL

B.4.7 Page 166, Section 17.10 of DIS 9506-2

CHANGE the transfer syntax object identifier value from
{ iso asn1(1) basic-encoding(1) }
TO
{ joint-iso-ccitt asn1(1) basic-encoding(1) }

B.5 Behavioral requirements

B.5.1 Filenames

File Names are specified in accordance with the NBS Implementors' agreements for FTAM Reference [3] in B.2.

B.5.2 Identify service

In the Identify service, the vendor, model and revision fields may be of any length, but only the first 64, 16, and 16 octets respectively are treated as significant.

B.5.3 Initiate service

An MMS Client will:

a) propose 1 or greater for the value of the Proposed Max Serv Outstanding Called parameter in the Initiate service when initiating the application association (calling);

b) offer 1 or greater for the value of the Negotiated Max Serv Outstanding Calling parameter in the Initiate service when receiving the application association initiation (called).

An MMS Server will:

a) propose 1 or greater for the value of the Proposed Max Serv Outstanding Calling parameter in the Initiate service when initiating the application association (calling);

b) offer 1 or greater for the value of the Negotiated Max Serv Outstanding Called parameter in the Initiate service when receiving the application association initiation (called).

B.5.3.1 Minimum segment size

MMS implementations are able to parse and process 512 octets of MMSpdu as they are encoded in ASN.1 basic encoding rules.

B.5.3.2 Maximum segment size

The Max Segment Size is defined as the maximum number of octets in an MMSpdu encoded using the negotiated transfer syntax. This size will apply to all MMSpdu's with the exception of the initiate-Request PDU, initiate-Response PDU, and the initiate-Error PDU. The max segment size will be negotiated during connection initiation using the Proposed Max Segment Size and Negotiated Max Segment Size parameters of the MMS initiate service.

The Max Segment Size will be applied as follows:

a) Any received MMSpdu which is less than or equal to the Max Segment Size will be properly parsed and processed;

b) An MMS implementation will not send an MMSpdu whose size exceeds the Max Segment Size.

B.5.4 Abstract syntax name

The ASN.1 object identifier value for the abstract syntax name will be the same as specified on page 166, section 17.10 of DIS 9506-2.

B.5.5 Application context name

The ASN.1 object identifier value for the application context name will be the same as specified on page 166, section 17.11 of DIS 9506-2.

An MMS implementation ignores the Application Context Name in the A-Associate indication and the A-Associate confirm.

B.5.6 Minor version number

The Minor Version Number is zero.

B.6 Parameter CBB subset

The following subset of MMS Parameter CBBs were considered during preparation of this annex:

a) STR1;

b) NEST;

c) VADR;

d) VNAM.

B.7 Service subset

The following subset of MMS services were considered during preparation of this annex.

Table 2 - MMS Service Subset

Initiate	Output
Conclude	TakeControl
Cancel	RelinquishControl
Status	ReportSemaphoreStatus
GetNameList	ReportPoolSemaphoreStatus
Identify	ReportSemaphoreEntryStatus
UnsolicitedStatus	CreateProgramInvocation
GetCapabilityList	DeleteProgramInvocation
InitiateDownloadSequence	Start
DownloadSegment	Stop
TerminateDownloadSequence	Resume
InitiateUploadSequence	Reset
UploadSegment	Kill
TerminateUploadSequence	GetProgramInvocationAttributes
RequestDomainDownload	ObtainFile
RequestDomainUpload	GetEventConditionAttributes
LoadDomainContent	ReportEventConditionStatus
StoreDomainContent	GetAlarmSummary
DeleteDomain	ReadJournal
GetDomainAttributes	WriteJournal
Read	InitializeJournal
Write	CreateJournal
InformationReport	DeleteJournal
GetVariableAccessAttributes	ReportJournalStatus
Input	