CCSDS

The Consultative Committee for Space Data Systems

Recommendation for Space Data System Standards

MISSION OPERATIONS— MESSAGE ABSTRACTION LAYER BINDING TO HTTP TRANSPORT AND XML ENCODING

RECOMMENDED STANDARD

CCSDS 524.3-B-1

BLUE BOOK
June 2018

The Consultative Committee for Space Data Systems

Recommendation for Space Data System Standards

MISSION OPERATIONS— MESSAGE ABSTRACTION LAYER BINDING TO HTTP TRANSPORT AND XML ENCODING

RECOMMENDED STANDARD

CCSDS 524.3-B-1

BLUE BOOK

June 2018

AUTHORITY

Issue:	Recommended Standard, Issue 1
Date:	June 2018
Location:	Washington, DC, USA

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS documents is detailed in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4), and the record of Agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the e-mail address below.

This document is published and maintained by:

CCSDS Secretariat
National Aeronautics and Space Administration
Washington, DC, USA
E-mail: secretariat@mailman.ccsds.org

STATEMENT OF INTENT

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of its members. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommended Standards** and are not considered binding on any Agency.

This **Recommended Standard** is issued by, and represents the consensus of, the CCSDS members. Endorsement of this **Recommendation** is entirely voluntary. Endorsement, however, indicates the following understandings:

- Whenever a member establishes a CCSDS-related **standard**, this **standard** will be in accord with the relevant **Recommended Standard**. Establishing such a **standard** does not preclude other provisions which a member may develop.

- Whenever a member establishes a CCSDS-related **standard**, that member will provide other CCSDS members with the following information:

  - The **standard** itself.

  - The anticipated date of initial operational capability.

  - The anticipated duration of operational service.

- Specific service arrangements shall be made via memoranda of agreement. Neither this **Recommended Standard** nor any ensuing **standard** is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this **Recommended Standard** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or (3) be retired or canceled.

In those instances when a new version of a **Recommended Standard** is issued, existing CCSDS-related member standards and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each member to determine when such standards or implementations are to be modified. Each member is, however, strongly encouraged to direct planning for its new standards and implementations towards the later version of the Recommended Standard.

FOREWORD

The intended use for this document is to allow the implementation of a protocol layer that binds the Mission Operations (MO) service framework to the Hypertext Transfer Protocol transport using XML Encoding. This document assumes that the reader is familiar with the MO concepts, especially the Message Abstraction Layer (MAL).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CCSDS has processes for identifying patent issues and for securing from the patent holder agreement that all licensing policies are reasonable and non-discriminatory. However, CCSDS does not have a patent law staff, and CCSDS shall not be held responsible for identifying any or all such patent rights.

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommended Standard is therefore subject to CCSDS document management and change control procedures, which are defined in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4). Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

Questions relating to the contents or status of this document should be sent to the CCSDS Secretariat at the e-mail address indicated on page i.

At time of publication, the active Member and Observer Agencies of the CCSDS were:

Member Agencies

- Agenzia Spaziale Italiana (ASI)/Italy.
- Canadian Space Agency (CSA)/Canada.
- Centre National d'Etudes Spatiales (CNES)/France.
- China National Space Administration (CNSA)/People's Republic of China.
- Deutsches Zentrum für Luft- und Raumfahrt (DLR)/Germany.
- European Space Agency (ESA)/Europe.
- Federal Space Agency (FSA)/Russian Federation.
- Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
- Japan Aerospace Exploration Agency (JAXA)/Japan.
- National Aeronautics and Space Administration (NASA)/USA.
- UK Space Agency/United Kingdom.

Observer Agencies

- Austrian Space Agency (ASA)/Austria.
- Belgian Federal Science Policy Office (BFSPO)/Belgium.
- Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
- China Satellite Launch and Tracking Control General, Beijing Institute of Tracking and Telecommunications Technology (CLTC/BITTT)/China.
- Chinese Academy of Sciences (CAS)/China.
- Chinese Academy of Space Technology (CAST)/China.
- Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
- Danish National Space Center (DNSC)/Denmark.
- Departamento de Ciência e Tecnologia Aeroespacial (DCTA)/Brazil.
- Electronics and Telecommunications Research Institute (ETRI)/Korea.
- European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
- European Telecommunications Satellite Organization (EUTELSAT)/Europe.
- Geo-Informatics and Space Technology Development Agency (GISTDA)/Thailand.
- Hellenic National Space Committee (HNSC)/Greece.
- Indian Space Research Organization (ISRO)/India.
- Institute of Space Research (IKI)/Russian Federation.
- Korea Aerospace Research Institute (KARI)/Korea.
- Ministry of Communications (MOC)/Israel.
- Mohammed Bin Rashid Space Centre (MBRSC)/United Arab Emirates.
- National Institute of Information and Communications Technology (NICT)/Japan.
- National Oceanic and Atmospheric Administration (NOAA)/USA.
- National Space Agency of the Republic of Kazakhstan (NSARK)/Kazakhstan.
- National Space Organization (NSPO)/Chinese Taipei.
- Naval Center for Space Technology (NCST)/USA.
- Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
- Scientific and Technological Research Council of Turkey (TUBITAK)/Turkey.
- South African National Space Agency (SANSA)/Republic of South Africa.
- Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
- Swedish Space Corporation (SSC)/Sweden.
- Swiss Space Office (SSO)/Switzerland.
- United States Geological Survey (USGS)/USA.

DOCUMENT CONTROL

Document	**Title**	**Date**	**Status**
CCSDS 524.3-B-1	Mission Operations—Message Abstraction Layer Binding to HTTP Transport and XML Encoding, Recommended Standard, Issue 1	June 2018	Original issue

CONTENTS

CONTENTS (continued)

Section Page

Figure

CONTENTS (continued)

1 INTRODUCTION

1.1 PURPOSE

This Recommended Standard defines two aspects of message exchange between MO service providers and consumers:

a) the binding between the Mission Operations (MO) Message Abstraction Layer (MAL) specified in reference [1] and the Hypertext Transfer Protocol (HTTP) specified in references [2] and [3];

b) an XML encoding for MAL data types.

The binding from the MAL to HTTP pertains only to the MAL message header and message exchange, and the XML encoding pertains only to the encoding of the MAL message body (see 2.4.1 for more information).

1.2 SCOPE

The scope of this Recommended Standard is the specification of the binding in terms of technology mapping to the HTTP and XML of:

a) MAL message;

b) MAL Transport Interface.

The MAL Blue Book (reference [1]) specifies the MAL protocol in an abstract way, i.e., without defining the concrete protocol data units. The MAL Binding to HTTP Transport and XML Encoding specifies:

a) a complete and unambiguous mapping of the MAL message to the HTTP messages;

b) a complete and unambiguous mapping of the MAL transport interface to the HTTP interface;

c) a complete and unambiguous mapping of the MAL data types to an XML encoding format.

This Recommended Standard does not specify:

a) individual implementations or products;

b) the implementation of entities or interfaces within real systems.

In a concrete deployment, on-the-wire interoperability between Application Layer MO Service consumer and provider will be achieved by encoding the abstract MAL messages in the concrete XML encoding and transmitting them by means of HTTP messages, as defined in this Recommended Standard.

1.3 APPLICABILITY

This Recommended Standard specifies a mapping to a concrete communication protocol that enables different implementations of the MO service framework (see 2.2) to interoperate through HTTP and XML encoding.

1.4 RATIONALE

CCSDS MO services are Application Layer services, which are specified in an abstract implementation- and communication-agnostic manner in terms of the MAL (Message Abstraction Layer).

In a concrete deployment scenario (instantiation of the abstract MO services in a concrete set of technologies) on-the-wire interoperability is achieved by agreeing on a concrete encoding and a concrete communication protocol for the exchange of the messages between the service provider and service consumer.

The goal of this Recommended Standard is to specify how to translate the abstract MAL message model in an unambiguous way into a concrete message-exchange protocol based on HTTP using an XML encoding.

This Recommended Standard also aims at defining a concrete XML encoding format for the MAL data types. The specified XML encoding is generic (i.e., independent of the MAL binding to HTTP protocol), and the resulting encoded MAL messages can be exchanged via any communication protocol for which a binding to the MAL exists. Equally, it is not mandatory to use the XML encoding specified in this book for encoding of the body of the messages when using the MAL to HTTP binding. Any MAL encoding, specified in other books, can be used for encoding the body of the messages when adopting the MAL to HTTP binding specified in this book.

Use of this full specification, as defined, will allow direct interoperability between a service consumer and a service provider complying with the full XML over HTTP specification. Adoption of just parts of this specification will require either out-of-band agreements to support the divergence, or a protocol-translation gateway for interoperability.

1.5 DOCUMENT STRUCTURE

This document is organized as follows:

a) section 1 presents the purpose, scope, applicability, and rationale, and lists the definitions, conventions, and references used throughout this Recommended Standard;

b) section 2 presents an overview of the MAL HTTP Transport and XML Encoding in relation to the MO service framework;

c) section 3 specifies the mapping of the MAL message to the HTTP messages;

d) section 4 specifies the mapping of the MAL transport interface to the HTTP interface;

e) section 5 specifies an XML encoding format for the MAL data types.

1.6 DEFINITIONS

protocol: The set of rules and formats (semantic and syntactic) used to determine the communication behaviour of a protocol layer in the performance of the layer functions. The state machines that operate and the protocol data units that are exchanged specify a protocol.

protocol layer: The implementation of a specific protocol. It provides a protocol service access point to layers above and uses the protocol service access point of the layer below.

protocol service access point: The point at which one layer’s functions are provided to the layer above. A layer may provide protocol services to one or more higher layers and use the protocol services of one or more lower layers.

1.7 NOMENCLATURE

1.7.1 NORMATIVE TEXT

The following conventions apply for the normative specifications in this Recommended Standard:

a) the words ‘shall’ and ‘must’ imply a binding and verifiable specification;

b) the word ‘should’ implies an optional, but desirable, specification;

c) the word ‘may’ implies an optional specification;

d) the words ‘is’, ‘are’, and ‘will’ imply statements of fact.

NOTE – These conventions do not imply constraints on diction in text that is clearly informative in nature.

1.7.2 INFORMATIVE TEXT

In the normative sections of this document, informative text is set off from the normative specifications either in notes or under one of the following subsection headings:

- Overview;
- Background;
- Rationale;
- Discussion.

1.8 BIT NUMBERING CONVENTION

In this document, the following convention is used to identify each bit in an *N*-bit field. The first bit in the field to be transmitted (i.e., the most left justified when drawing a figure) is defined to be ‘Bit 0’; the bit following is defined to be ‘Bit 1’, and so on up to ‘Bit *N*–1’. When the field is used to express a binary value (such as a counter), the Most Significant Bit (MSB) shall be the first transmitted bit of the field, i.e., ‘Bit 0’.

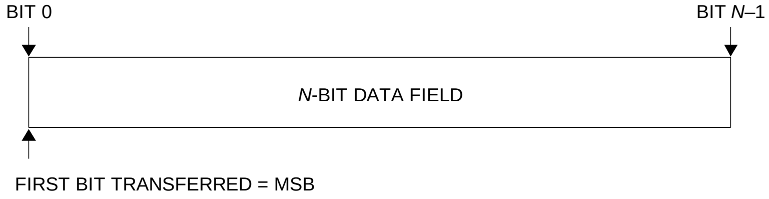

Figure 1-1: Bit Numbering Convention

In accordance with modern data communications practice, spacecraft data fields are often grouped into 8-bit ‘words’ that conform to the above convention. Throughout this Recommended Standard, the following nomenclature is used to describe this grouping:

8-BIT WORD = ‘OCTET’

Figure 1-2: Octet Convention

By CCSDS convention, all ‘spare’ or ‘unused’ bits shall be permanently set to value ‘zero’.

1.9 REFERENCES

The following publications contain provisions which, through reference in this text, constitute provisions of this document. At the time of publication, the editions indicated were valid. All publications are subject to revision, and users of this Recommended Standard are encouraged to investigate the possibility of applying the most recent editions of the publications indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS publications.

NOTE – A list of informative references is provided in annex D.

[1] *Mission Operations Message Abstraction Layer*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 521.0-B-2. Washington, D.C.: CCSDS, March 2013.

[2] R. Fielding and J. Reschke, eds. *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing*. RFC 7230. Reston, Virginia: ISOC, June 2014.

[3] R. Fielding and J. Reschke, eds. *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*. RFC 7231. Reston, Virginia: ISOC, June 2014.

[4] R. Fielding and J. Reschke, eds. *Hypertext Transfer Protocol (HTTP/1.1): Authentication*. RFC 7235. Reston, Virginia: ISOC, June 2014.

[5] *Time Code Formats*. Issue 4. Recommendation for Space Data System Standards (Blue Book), CCSDS 301.0-B-4. Washington, D.C.: CCSDS, November 2010.

[6] R. Hinden and S. Deering. *IP Version 6 Addressing Architecture*. RFC 4291. Reston, Virginia: ISOC, February 2006.

[7] Shudi (Sandy) Gao, C. M. Sperberg-McQueen, and Henry S. Thompson, eds. *W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures*. Version 1.1. W3C Recommendation. N.p.: W3C, 5 April 2012.

[8] David Peterson, et al., eds. *W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes*. Version 1.1. W3C Recommendation. N.p.: W3C, 5 April 2012.

[9] T. Berners-Lee, R. Fielding, and R. Fielding. *Uniform Resource Identifier (URI): Generic Syntax*. STD 66. Reston, Virginia: ISOC, January 2005.

[10] K. Moore. *MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text*. RFC 2047. Reston, Virginia: ISOC, November 1996.

2 OVERVIEW

2.1 GENERAL

This Recommended Standard allows MO services defined in terms of the MAL to interoperate across an end-to-end communication link using a normative binding of the MAL abstractions to the Hypertext Transfer Protocol over TCP/IP for exchanging messages. This is of particular interest for MO services for which the service provider and consumer are both deployed on the ground, for instance, when the MO service provider is located in a Mission Control Centre and the consumer in the Science Control Centre. The messages that the provider and the consumer exchange to implement the MO services are encoded in XML and transferred using HTTP over TCP/IP.

To achieve this goal, this Recommended Standard provides a technology mapping of the MAL transport interface, the MAL abstract message, and the MAL data types specification (reference [1]) to the HTTP protocol stack (references [2], [3], and [4]) and to a concrete XML encoding, which can be used to encode the body of the MAL messages exchanged over the HTTP protocol.

It should be noted that although this specification refers to mapping to HTTP it is recommended that the preferred way of communicating the encoded HTTP messages is using HTTPS (see annex subsection B1 for more information on this).

The MAL Blue Book (reference [1]) defines an abstract transport interface as a set of request and indication primitives. The mapping to a concrete transport protocol specifies how these primitives are realized according to the rules and requirements of that particular messaging protocol.

The mapping of the MAL to a concrete communication protocol translates the MAL message model into one or several protocol-specific Protocol Data Units (PDUs). MAL messages are composed of two conceptual segments, the MAL header and the MAL body. The header of the MAL message contains the metadata and is mapped to the protocol-specific header encodings. The body of the MAL message can be encoded using the encoding that best fits the requirements of a particular deployment.

Full interoperability of services is achieved if the same MAL to transport protocol binding and the same encoding for the body of the MAL messages are used by the service provider and the service consumer. Alternatively, a bridge must be used to translate from one binding/encoding to another (reference [D1]).

The diagram shown in figure 2-1 presents the set of documents in support of the Mission Operations Services Concept. This MO HTTP Transport and XML Encoding book belongs to the technology mappings documentation.

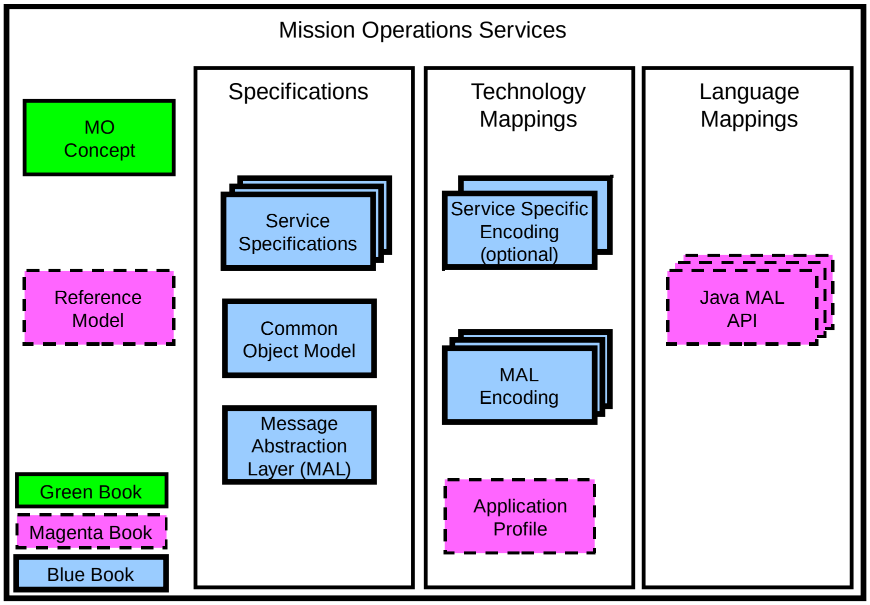

Figure 2-1: Mission Operations Services Concept Document Set

2.2 MO SERVICE FRAMEWORK OVER HTTP AND XML

The CCSDS Spacecraft Monitoring & Control (SM&C) working group has developed a concept for an MO service framework that follows the principles of service-oriented architectures. The framework defines two important aspects, a protocol for interaction between two separate entities, and a set of common services providing functionality shared by most of the MO services. An overview of this framework is presented in figure 2-2.

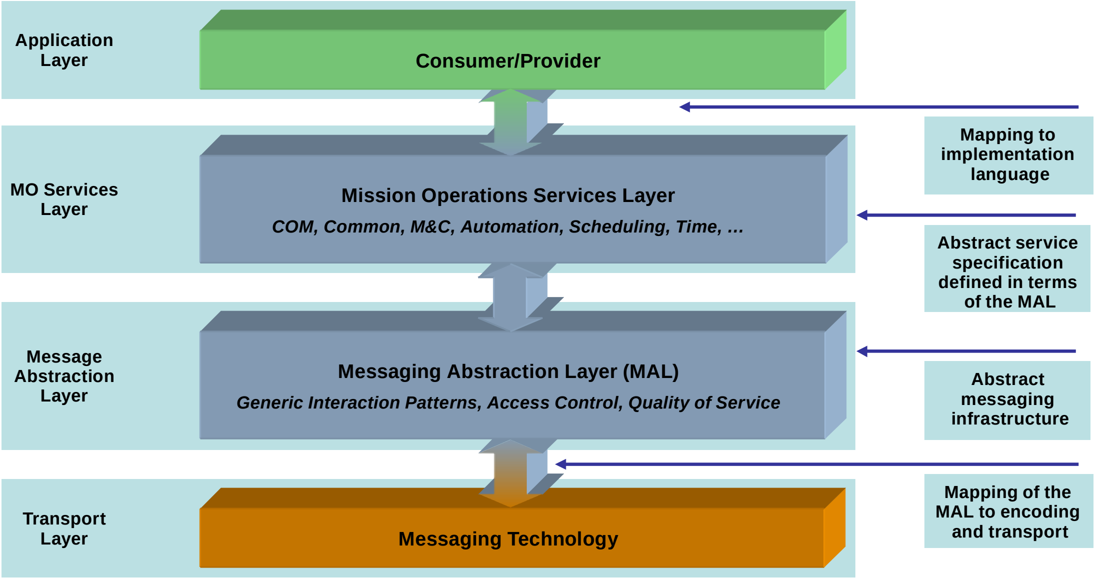
Application Layer
Consumer/Provider
MO Services Layer
Mission Operations Services Layer
COM, Common, M&C, Automation, Scheduling, Time, ...
Message Abstraction Layer
Messaging Abstraction Layer (MAL)
Generic Interaction Patterns, Access Control, Quality of Service
Transport Layer
Messaging Technology
Mapping to implementation language
Abstract service specification defined in terms of the MAL
Abstract messaging infrastructure
Mapping of the MAL to encoding and transport

Figure 2-2: Overview of the MO Service Framework

This Recommended Standard defines how the MAL layer is mapped to the specific Transport Layer technology called Hypertext Transfer Protocol over Transmission Control Protocol. More specifically, it specifies:

a) how the specific technology is to be used;

b) how any transmission errors or issues are to be communicated to higher layers;

c) how all underlying Data Link or Network Layer issues are to be handled;

d) the physical representation of the MAL messages necessary to constitute the operation templates;

e) the mapping of the message structure rules for that technology;

f) the encoding of the MAL data types.

It does not specify:

a) individual application services, implementations, or products;

b) the implementation of entities or interfaces within real systems;

c) the methods or technologies required to acquire data;

d) the management activities required to schedule a service;

e) the representation of any service-specific PDUs (this is derived from the encoding format defined in this document in section 5).

The MAL Blue Book (reference [1]) groups all the interfaces to the Transport Layer in a single place called the MAL transport interface (subsection 3.7 of reference [1]). Because of this, only the MAL transport interface needs to be mapped to the HTTP protocol and underlying transport protocol, without the need to map the entire MAL Blue Book.

Figure 2-3 expands the previous figure (figure 2-2) by presenting the MAL HTTP Transport Layer in the MO service framework stack, and by highlighting the various interfaces and their main primitives.

Figure 2-3 shows that the mapping of the MAL transport interface to the Transport Layer requires the insertion of a layer in between. This layer is called the MAL HTTP Transport. It is responsible for the mapping to XML and HTTP of the abstract MAL message, which is then transferred through PDUs of the Transport Layer.

The protocol stack represented in figure 2-3 is conceptual. It can be implemented in various ways. For example, an implementation of the stack may, for performance reasons, merge the MAL layer, the MAL Transport, and HTTP into a single layer called ‘MAL over HTTP’.

The names of the main interfaces used and implemented by each layer are given by figure 2-3. The main primitives are shown for each interface:

a) the primitives for every operation provided by an MO service;

b) the primitives for every interaction pattern provided by MAL;

c) the primitives for transmitting and receiving a single MAL message or multiple MAL messages;

d) the primitives for transmitting and receiving data on the Transport Layer.

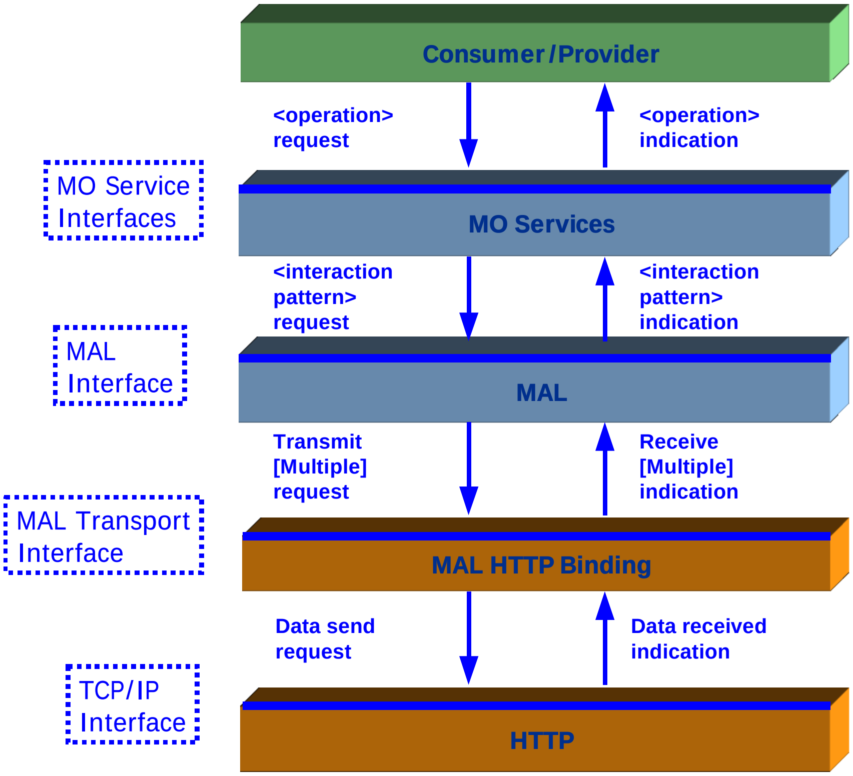

Figure 2-3: MO Test Service Framework above HTTP

2.3 TYPICAL USE

Possible uses of the MAL HTTP Transport binding may be between MO entities (service consumer and provider) operating on ground; for example:

a) ground applications deployed on the same machine or interacting over a local area network using Transmission Control Protocol (TCP)/Internet Protocol (IP);

b) ground components interacting over a wide area network or over the Internet;

c) mobile applications consuming MAL services over wireless networks supporting TCP/IP.

The mapping of the MAL operations over the HTTP protocol, as specified by this Recommended Standard, requires the presence of an HTTP server located also on the consumer side, when used with services that define operations with interaction patterns INVOKE, PROGRESS, or PUBSUB (figure 2-4), as these patterns define multiple return messages to the consumer.

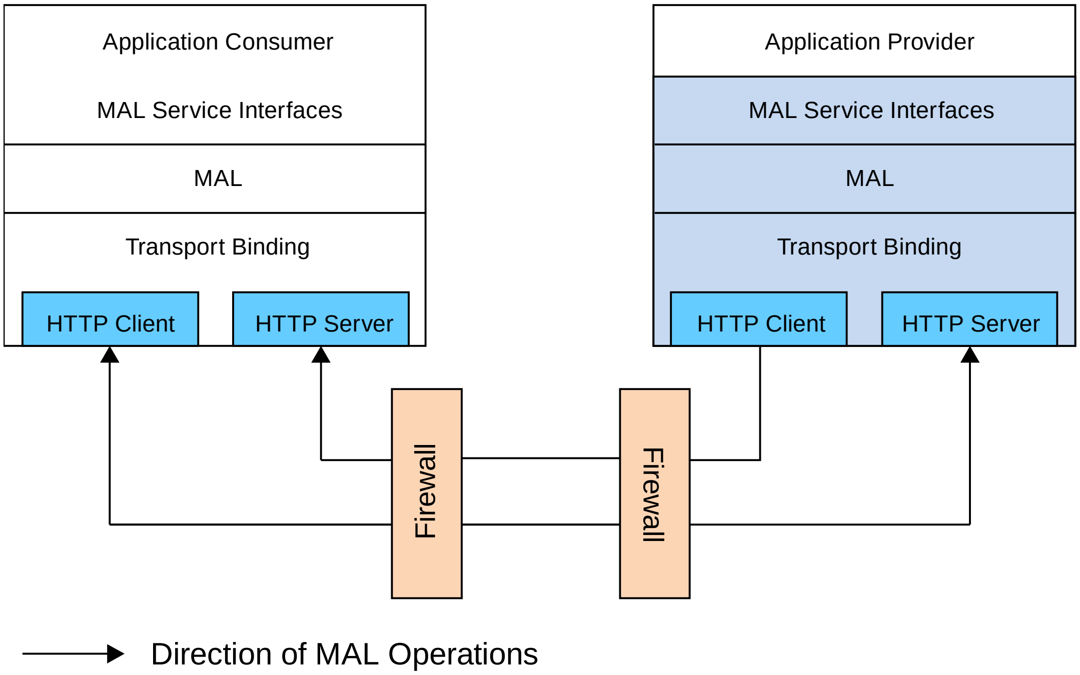

Figure 2-4: Deployment Using the MAL HTTP Transport Binding

The binding specified in this Recommended Standard foresees, for a provider or a broker, the knowledge of the HTTP consumer endpoint to be used when sending messages related to the following Interaction Patterns and Interaction Stages:

a) INVOKE – RESPONSE (provider to consumer);

b) INVOKE – RESPONSE ERROR (provider to consumer);

c) PROGRESS – UPDATE (provider to consumer);

d) PROGRESS – UPDATE ERROR (provider to consumer);

e) PROGRESS – RESPONSE (provider to consumer);

f) PROGRESS – RESPONSE ERROR (provider to consumer);

g) PUBSUB – NOTIFY (broker to consumer);

h) PUBSUB – NOTIFY (broker to consumer);

i) PUBSUB – NOTIFY ERROR (broker to consumer).

The mechanism of how the provider/broker determines the information about the location of the consumer endpoint is implementation-specific (e.g., by configuration, using the 'URI From' header field directly or via a mapping table) and therefore not defined by this Recommended Standard.

Nevertheless, the mapping of the MAL operations over the HTTP protocol as specified by this Recommended Standard prescribes the presence of an HTTP server located also on the consumer side, in the case of services defining operations with interaction pattern INVOKE, PROGRESS, or PUBSUB (figure 2-5).

The envisaged scenario is one in which both parties are known to each other and, if one or both parties are behind a firewall, each has control over its respective firewall, as depicted in figure 2-5).

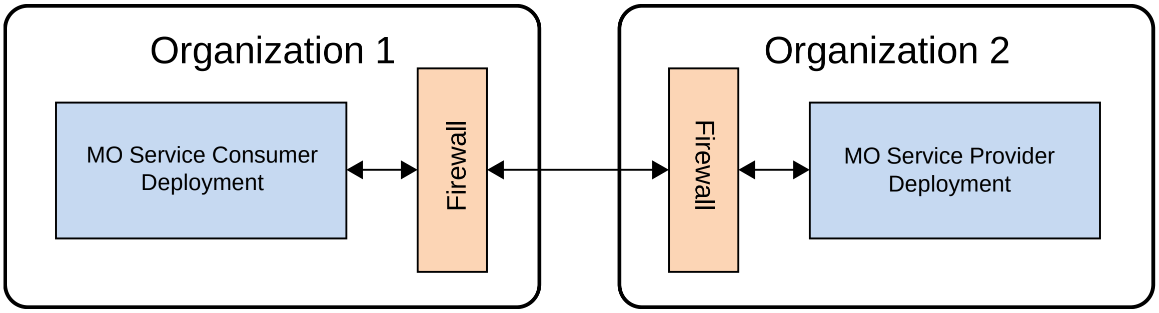
Organization 1
MO Service Consumer Deployment
Firewall
Organization 2
Firewall
MO Service Provider Deployment

Figure 2-5: Point to Point between Two Well-Known Applications

In such situations, the usage of the HTTP transport mapping, compared to other low-level MAL transport mappings, brings the following advantages:

a) HTTP is located above the Transport Layer in the communication protocol stack; hence an HTTP implementation does not have to deal with the low-level implementation details, compared to a transport protocol implementation;

b) HTTP is firewall-friendly as compared to binary data sent over random TCP/IP ports. A stateful HTTP firewall/proxy might easily enforce access control and routing rules by inspecting the contents of the HTTP header fields;

c) support for development and debugging is provided by an extensive set of existing tools.

Further discussion of the issues related to MAL consumers requiring an HTTP server and firewall considerations is contained in annex subsection B1.6.

2.4 MAL MESSAGE MAPPING

2.4.1 MAPPING TO HTTP

The Hypertext Transfer Protocol (HTTP) is a stateless application-level request/response protocol for distributed, collaborative, hypertext information systems. It uses extensible semantics and self-descriptive message payloads for flexible interactions.

This Recommended Standard introduces a MAL HTTP Message and an XML encoded body, which is delivered using the HTTP underlying transport protocol. Therefore each field of the MAL message needs to map to a field of this MAL HTTP Message.

Figure 2-6 illustrates the mapping of the MAL message to the MAL HTTP Message, transmitted over the transport protocol. Most of the MAL message fields are mapped according to a one-to-one equivalence. The following colour convention is used:

Table 2-1: MAL Field Colour Conventions

Background Colour	Meaning
Blue	MAL header field mapped one-to-one to HTTP header fields
Red	Standard/Custom HTTP fields not part of the MAL header field set, but introduced by this Recommended Standard
Yellow	MAL header fields mapped to several HTTP fields

In this case the original MAL header field name is kept and the background colour is blue. However, the following fields require a more complex mapping:

a) If the destination specified in the ‘URI To’ field coincides with the HTTP destination, the MAL header field ‘URI To’ is mapped to the HTTP header field ‘Host’ and to the ‘request-target’ field of the HTTP request-line; in this case the HTTP header field ‘X-URI-To’ is not provided.

b) Otherwise, if the destination specified in the ‘URI-To’ field does not coincide with the HTTP destination, the MAL header field ‘URI To’ is mapped to the HTTP header field ‘X-URI-To’. The background colour is yellow. This is the case in multi-hop scenarios.

A custom header field named ‘Version Number’ is introduced. The purpose of this field is to allow future evolution of the message header and body content as defined by this version of the Recommended Standard. The background colour is red.

This Recommended Standard prescribes an XML encoding for the message body. However, in order to allow flexibility in the selection of the encoding formats to be used for the MAL message body, this Recommended Standard uses two additional header fields:

a) the standard header field ‘Content-Type’ identifies which encoding is used to encode the MAL message body; the background colour is red;

b) the standard header field ‘Content-Length’ reports the length in octets of the MAL message body; the background colour is red.

Finally, the MAL message body field and its equivalent HTTP message body have a grey background.

NOTE – Any full implementation of the MAL HTTP binding requires explicit support for multi-hop scenarios. A simple implementation could retrieve the connection information for a response message from the MAL message headers directly, in which case a multi-hop scenario is not possible. A suggested alternative approach could use a routing table to support multi-hop scenarios. Only in this case, the X-MAL-URI-To header is relevant.

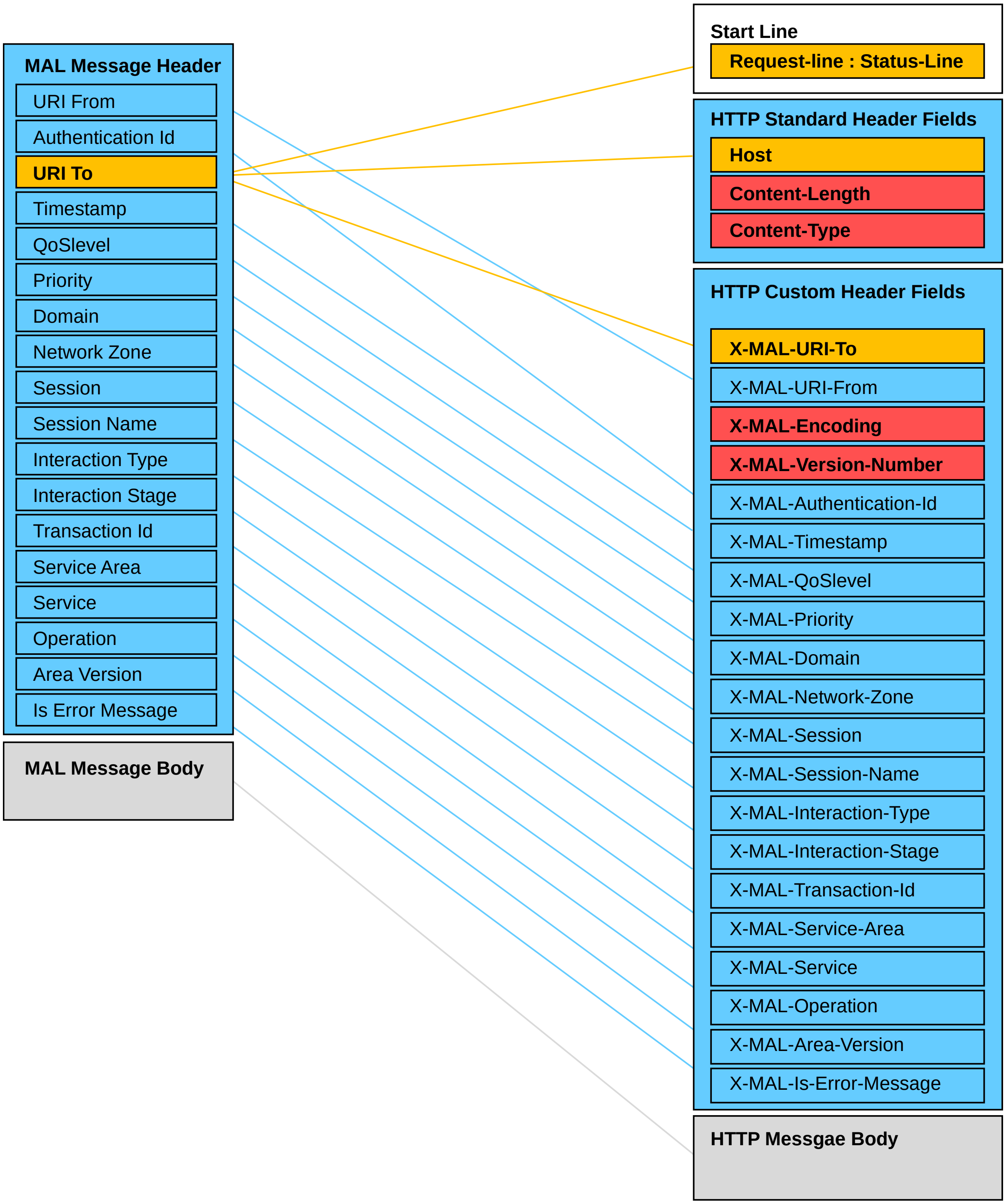

Figure 2-6: MAL Message Mapping to HTTP

2.4.2 MAPPING SPECIFICATION

The MAL Binding to HTTP Transport and XML Encoding defines an XML encoding format for every MAL data type (see section 5).

An XML XSD schema notation is used to specify the format of the mapping result.

2.4.3 COMPLETE MAPPING

The MAL message mapping completeness is ensured by the following conditions:

a) every MAL data type is mapped;

b) every MAL message field is mapped to XML or to an HTTP header field;

c) every mandatory HTTP message field is assigned.

Moreover, the translation from a MAL message to its HTTP and XML message forms is reversible. No information is lost in the translation from a MAL message to its HTTP and XML message forms.

2.5 MAL TRANSPORT INTERFACE MAPPING

The mapping of the MAL transport interface requires specifying the expected behaviour for each of the MAL transport primitives. Three types of behaviour are defined:

a) a MAL transport request initiating an HTTP request by sending a message request and returning a reply;

b) an HTTP indication initiating a MAL transport indication when receiving a request or response message;

c) a MAL transport request returning a reply without calling the HTTP layer (for returning local transport errors such as unresolvable TCP/IP addresses).

The MAL transport mapping is complete as all the primitives are mapped.

Moreover, the behaviour of each primitive is fully specified.

3 MAL MESSAGE MAPPING

3.1 OVERVIEW

This section specifies how the MAL message header, body, and QoS properties are mapped to the HTTP messages. Table 3-1 is taken from reference [1] and provides the full list of fields in the MAL message header.

Table 3-1: MAL Message Header Fields

Field	Type	Value
URI From	URI	Message Source URI
Authentication Id	Blob	Source Authentication Identifier
URI To	URI	Message Destination URI
Timestamp	Time	Message generation timestamp
QoSlevel	QoSLevel	The QoS level of the message
Priority	UInteger	The QoS priority of the message
Domain	List<Identifier>	Domain of the message
Network Zone	Identifier	Network zone of the message
Session	SessionType	Type of session of the message
Session Name	Identifier	Name of the session of the message
Interaction Type	InteractionType	Interaction Pattern Type
Interaction Stage	UOctet	Interaction Pattern Stage
Transaction Id	Long	Unique to consumer
Service Area	UShort	Service Area Identifier
Service	UShort	Service Identifier
Operation	UShort	Service Operation Identifier
Area version	UOctet	Area version
Is Error Message	Boolean	‘True’ if this is an error message; otherwise ‘False’

The MAL message header is mapped to the HTTP start-line, and to HTTP standard and custom header fields.

The HTTP message structure, including precise format definition, is specified by the HTTP Message Syntax and Routing reference [2] and HTTP Semantics and Content reference [3]. These define in particular the header fields that may be used in all messages, or specifically for request messages, or specifically for response messages. The message header fields are used to provide more information about the context, make a request conditional based on the target resource state, suggest preferred formats for the response, supply authentication credentials, or modify the expected request processing. These fields act as request modifiers and are not mandatory.

Each HTTP message is composed of a start-line, a header, and a body. The HTTP start-line can be either a request-line, in the case of a request message, or a status-line, in the case of a response message.

3.2 GENERAL

3.2.1 START-LINE

For MAL HTTP binding, fields composing a start-line shall be set as specified in table 3-2.

Table 3-2: HTTP Start-Line Format

Line field	Description
request-line	
Method	The request method to be performed on the target resource. The request method is case-sensitive. Shall be set to POST.
request-target	The target resource upon which to apply the request. In the case where the HTTP destination coincides with the Destination Id specified in the 'URI To' MAL header field, this field shall be set to the Destination ID of the destination service or application. In the case where the HTTP destination is being used as a relay to another final destination (where the HTTP destination does not coincide with the Destination Id specified in the 'URI To' MAL header field) this field shall be set to the Destination Id of the relay application which is responsible for instigating the next part of the relay.
HTTP-version	The version of the HTTP protocol in use. Shall be set and used as defined by HTTP (see references [2] and [3]).
status-line	
HTTP-version	The version of the HTTP protocol in use. Shall be set and used as defined by HTTP (see references [2] and [3]).
status-code	3-digit integer code describing the result of the server's attempt to understand and satisfy the client's corresponding request. Shall be set as defined in table 3-4, or for specific error code as defined by HTTP (see references [2] and [3]).
reason-phrase	Textual description associated with the numeric status code. Should be ignored by the receiver of the receive message and is only provided for information.

3.2.2 MAL TO HTTP INTERACTION MAPPING

3.2.2.1 Discussion

HTTP mandates that for each request message, a response message is required to be sent in the other direction. Therefore, additional HTTP response messages need to be provided in some cases, even if these are not foreseen or used by the MAL. In this case the MAL interaction stage in table 3-3 is marked 'n/a' and is to be ignored.

3.2.2.2 Requirements

The mapping of the MAL interaction messages to the corresponding HTTP request and response messages, and to the HTTP message initiator, shall be as specified in table 3-3.

NOTE – Reference [1] provides full description of the relevant interaction types, the MAL interaction stages, and the initiators given table 3-3.

Table 3-3: MAL Interaction to HTTP Message Mapping

Interaction Type	MAL Interaction Stage	HTTP Message	
		Request / Response	Initiator
SEND	SEND	Request	Consumer
	n/a	Response	Provider
SUBMIT	SUBMIT	Request	Consumer
	ACK	Response	Provider
	ERROR	Response	Provider
REQUEST	REQUEST	Request	Consumer
	RESPONSE	Response	Provider
	ERROR	Response	Provider
INVOKE	INVOKE	Request	Consumer
	ACK	Response	Provider
	ACK_ERROR	Response	Provider
	RESPONSE	Request	Provider
	n/a	Response	Consumer
	RESPONSE_ERROR	Request	Provider
	n/a	Response	Consumer
PROGRESS	PROGRESS	Request	Consumer
	ACK	Response	Provider
	ACK_ERROR	Response	Provider
	UPDATE	Request	Provider
	n/a	Response	Consumer
	UPDATE_ERROR	Request	Provider
	n/a	Response	Consumer
	RESPONSE	Request	Provider
	n/a	Response	Consumer
	RESPONSE ERROR	Request	Provider
	n/a	Response	Consumer
PUBLISH-SUBSCRIBE	REGISTER	Request	Consumer
	REGISTER_ACK	Response	Broker
	REGISTER_ERROR	Response	Broker
	PUBLISH_REGISTER	Request	Provider
	PUBLISH_REGISTER_ACK	Response	Broker
	PUBLISH_REGISTER_ERROR	Response	Broker
	PUBLISH	Request	Provider

Interaction Type	MAL Interaction Stage	HTTP Message	
		Request / Response	Initiator
	n/a	Response	Broker
	PUBLISH_ERROR	Request	Broker
	n/a	Response	Provider
	NOTIFY	Request	Broker
	n/a	Response	Consumer
	NOTIFY ERROR	Request	Broker
	n/a	Response	Consumer
	DEREGISTER	Request	Consumer
	DEREGISTER _ACK	Response	Broker
	PUBLISH_DEREGISTER	Request	Provider
	PUBLISH_DEREGISTER_ACK	Response	Broker

3.2.3 STATUS CODES

3.2.3.1 Discussion

The status line includes a three-digit integer code describing the result of the server's attempt to understand and satisfy the client's corresponding request. The first digit of the status-code defines the class of response. The last two digits do not have any categorization role. There are five values for the first digit:

a) 1xx (Informational): the request was received, continuing process;

b) 2xx (Successful): the request was successfully received, understood, and accepted;

c) 3xx (Redirection): further action needs to be taken in order to complete the request;

d) 4xx (Client Error): the request contains bad syntax or cannot be fulfilled;

e) 5xx (Server Error): the server failed to fulfil an apparently valid request.

3.2.3.2 Requirements

3.2.3.2.1 Errors Detected at the MAL Protocol Level

3.2.3.2.1.1 HTTP status codes shall be set as defined in table 3-4, according to the Interaction Type and Stage of the MAL message header. This includes errors generated locally at the MAL protocol level and errors generated as a MAL response.

3.2.3.2.1.2 Specific error codes as defined by HTTP (see references [2] and [3]) may also be used in case of communication errors and problems, or errors at the HTTP level.

NOTE – The mapping to MAL errors is given in table 3-6.

Table 3-4: Status Codes in Response Message

Status Code	Description	Usage
200 **OK**	Standard response for successful HTTP requests.	SUBMIT_ACK REQUEST_RESPONSE PROGRESS_ACK REGISTER_ACK PUBLISH_REGISTER_ACK DEREGISTER_ACK PUBLISH_DEREGISTER_ACK
202 **Accepted**	The request has been accepted for processing, but the processing has not been completed.	INVOKE_ACK
204 **No Content**	The server successfully processed the request but is not returning any content.	SEND PROGRESS_UPDATE_STAGE PROGRESS_RESPONSE_STAGE PUBSUB_NOTIFY_STAGE PUBSUB_PUBLISH_STAGE
4xx **Client errors**	Errors generated because of HTTP client errors.	SUBMIT_ERROR REQUEST_ERROR INVOKE_ACK_ERROR PROGRESS_ACK_ERROR REGISTER_ERROR PUBLISH_REGISTER_ERROR
5xx **Server Errors**	Errors generated because of HTTP server errors.	SUBMIT_ERROR REQUEST_ERROR INVOKE_ACK_ERROR PROGRESS_ACK_ERROR REGISTER_ERROR PUBLISH_REGISTER_ERROR

3.2.3.2.2 HTTP Errors

3.2.3.2.2.1 When an HTTP error is being returned in a MAL Error message, the mapping from HTTP error code to MAL error code in table 3-5 shall be used.

3.2.3.2.2.2 Other HTTP errors are not envisaged to be seen; however, in the case that they are, they shall be mapped to the MAL INTERNAL error.

3.2.3.2.2.3 Upon receiving an HTTP error code, the MAL message body shall be set to contain only the corresponding MAL error, formatted according to reference [1].

3.2.3.2.2.4 MAL Error messages shall not be mapped to HTTP errors; only a mapping from HTTP Error to MAL is required.

Table 3-5: HTTP Error Code to MAL Error Code Mapping

HTTP Error Code	Description	MAL Error Code
400	Bad Request	BAD_ENCODING
401	Unauthorized (RFC 7235) (reference [4])	AUTHORISATION_FAIL
403	Forbidden	AUTHORISATION_FAIL

HTTP Error Code	Description	MAL Error Code
404	Not Found	DESTINATION_UNKNOWN
405	Method Not Allowed	UNSUPPORTED_OPERATION
408	Request Timeout	DELIVERY_TIMEDOUT
410	Gone	DESTINATION_TRANSIENT
429	Too Many Requests (RFC 6585)	TOO_MANY
500	Internal Server Error	INTERNAL
501	Not Implemented	UNSUPPORTED_OPERATION
502	Bad Gateway	DELIVERY_FAILED
503	Service Unavailable	DESTINATION_TRANSIENT
504	Gateway Time-out	DELIVERY_TIMEDOUT
511	Network Authentication Required (RFC 6585)	AUTHENTICATION_FAIL

3.2.4 HEADER FIELDS

3.2.4.1 Discussion

Each HTTP message contains standard and custom fields. As defined in reference [2], each header field consists of a case-insensitive field name followed by a colon (‘:’), optional leading whitespace, the field value, and optional trailing whitespace. The field-name token labels the corresponding field-value as having the semantics defined by that header field.

3.2.4.2 Requirements

3.2.4.2.1 Header fields shall be those specified in table 3-6.

3.2.4.2.2 Fields identified as mandatory shall always be present.

NOTE – The fields are provided in table 3-6 in the same order as the one used for the MAL header. Some fields are optional (since they are optional at MAL level).

3.2.4.2.3 As specified in reference [2], the order in which HTTP header fields are received is not significant, and implementation of message consumer shall not rely on any specific order.

3.2.4.2.4 The X-MAL-URI-To field is optional; if present, it shall be set to the complete ‘URI To’ from the MAL message and shall supersede the construction of the ‘URI To’ using the Host and request-target HTTP fields.

Table 3-6: MAL HTTP Header Fields

Field Identifier	Mandatory / Optional	Standard / Custom	Description
Host	M	S	Destination host. Shall be set to the IP address (or host name) and TCP port number of the ‘URI To’ from the MAL message, in case the HTTP destination coincides with the destination specified in the ‘URI To’ MAL header field; otherwise it shall be set to the IP address (or host name) and TCP port number of the intermediate hop, in case of multi-hop transmission.
X-MAL-Authentication-Id	M	C	The source Authentication Identifier.
X-MAL-URI-From	M	C	Source of the message. Shall be set to the complete ‘URI From’ from the MAL message.
X-MAL-URI-To	O	C	Destination of the message. If present, it shall be set to the complete ‘URI To’ from the MAL message. If present, it supersedes the construction of the ‘URI To’ using the Host and request-target HTTP fields.
X-MAL-Timestamp	M	C	The message generation timestamp.
X-MAL-QoSlevel	M	C	The QoS level of the message.
X-MAL-Priority	M	C	The QoS priority of the message.
X-MAL-Domain	M	C	The domain of the message.
X-MAL-Network-Zone	M	C	The network zone of the message.
X-MAL-Session	M	C	The type of session of the message.
X-MAL-Session-Name	M	C	The name of the session of the message.
X-MAL-Interaction-Type	M	C	The interaction pattern type.
X-MAL-Interaction-Stage	M	C	The interaction pattern stage.
X-MAL-Transaction-Id	M	C	Unique to consumer.
X-MAL-Service-Area	M	C	The service area.
X-MAL-Service	M	C	The service.
X-MAL-Operation	M	C	The service operation.
X-MAL-Area-Version	M	C	The area version.
X-MAL-Is-Error-Message	M	C	The error message indication. ‘True’ if this is an error message; otherwise ‘False’.
X-MAL-Encoding	O	C	The encoding used for the HTTP body. It shall be present only if the value of the Content-Type header field is set to ‘application/mal’
X-MAL-Version-Number	M	C	The version of the message header and body structure.

Field Identifier	Mandatory / Optional	Standard / Custom	Description
Content-Type	M	S	Type of encoding used to encode the data in the message body. Its value shall be set to ‘application/mal-xml’ if the MAL body is encoded using the encoding rules defined in this Standard. If a different set of encoding rules is used, then its value shall be set to ‘application/mal’ and the additional HTTP header field ‘X-MAL-Encoding’ shall be used to determine the encoding of the HTTP body content.
Content-Length	M	S	Length of the encoded message body, in octets.

NOTE – The QoS properties allow setting the values of the header flags, which cannot be deduced from the MAL header fields.

3.3 DISCUSSION

The mapping of the MAL message is composed of the following specifications:

a) the URI format to be applied to the MAL header fields ‘URI From’ and ‘URI To’;

b) the mapping of the MAL header fields to the HTTP message header fields defined by this Recommended Standard;

c) the values to be assigned to the HTTP message fields that are not the result of the MAL message header and body mapping;

d) the mapping of the MAL message body to the HTTP message body defined by this Recommended Standard.

This Recommended Standard supports the use of other encoding formats using the custom X-MAL-Encoding HTTP header field. Subsection 3.7.3 and section 5 defines the encoding format to be used in case the XML Encoding is selected.

3.4 URI FORMAT

NOTES

1 The following statements are about the MAL abstraction called URI and not about how it is mapped to the MAL HTTP message fields.

2 This Recommended Standard considers that the underlying layer of HTTP is a TCP/IP Transport Layer.

3.4.1 The format of the MAL header fields ‘URI From’ and ‘URI To’ shall comply with the following rules:

a) The URI scheme name shall be ‘malhttp’.

b) The scheme name shall be followed by a colon separator ':' and a double slash '//'.

c) The double slash shall be followed by the IP address or host name.

d) If version 6 of the Internet Protocol is used, the IP address shall be represented using the textual representation specified in reference [6], subsection 2.2. The IP address shall be enclosed in square brackets '[' and ']'; if version 4 of the Internet Protocol is used, the IP address shall be represented in dot-decimal notation.

e) The IP address or host name shall be followed by a colon separator ':' and the TCP port number, an integer represented in decimal.

f) The TCP port number shall be a positive integer, excluding zero, strictly less than 65536.

g) The TCP port number may be followed by a slash separator '/' and a non-empty string which is called the 'Source Id' for the field 'URI From' and the 'Destination Id' for the field 'URI To'.

NOTES

1 The source or destination identifier is optional if the host address and port is enough to completely identify the MAL application (where there is a single MAL service provider or consumer using that address).

2 An example of a URI using an Internet Protocol version 4 address is 'malhttp://192.168.0.1:2534/Service'. This URI references the source or destination 'Service' provided by the application, accessible from the TCP port '2534' on the host located at address '192.168.0.1'.

3 An example of a URI using an Internet Protocol version 6 address is 'malhttp://[2001:0db8:85a3:0000:0000:8a2e:0370:7334]:972/Service'. This URI references the source or destination 'Service' provided by the application accessible from the TCP port '972' on the host located at address '2001:0db8:85a3:0000:0000:8a2e:0370:7334'.

3.4.2 The IP address or host name and TCP port number shall uniquely identify a MAL application.

3.4.3 The source or destination identifier shall be unique for a given MAL application identified by its IP address or host name and TCP port number.

NOTE – A single application, which is identified by a single IP address and a single TCP port number, may represent several MAL entities. In order to uniquely address a single MAL entity, source and destination identifiers are used to refine the IP address and TCP port number.

3.4.4 The scheme name 'malhttp' shall be added to the SANA registry 'MAL Binding URI Scheme Name' and shall refer to the Mission Operations HTTP Transport and XML Encoding document 'CCSDS 524.3-B-1'.

3.5 MAL HEADER MAPPING

3.5.1 OVERVIEW

The following subsections provide the mapping of each field of the MAL message header to the MAL HTTP message header fields.

3.5.2 URI FROM

3.5.2.1 The MAL header field ‘URI From’ shall be assigned to the ‘X-MAL-URI-From’ HTTP message header field.

3.5.2.2 The content of the ‘X-MAL-URI-From’ HTTP message header field shall be encoded according to the rules defined in reference [9], section 2.

3.5.3 AUTHENTICATION ID

3.5.3.1 The MAL header field ‘Authentication Id’ shall be assigned to the ‘X-MAL-Authentication-Id’ HTTP message header field.

3.5.3.2 The MAL:Blob defining the MAL header field ‘Authentication Id’ shall be encoded into the HTTP message header field ‘X-MAL-Authentication-Id’ using hexadecimal characters.

3.5.3.3 The Blob octets shall be encoded in the same order as they appear in the ‘Authentication Id’ field.

3.5.4 URI TO

3.5.4.1 If the final destination of the MAL message as specified in the MAL header field ‘URI To’ coincides with the HTTP destination endpoint, the MAL ‘URI To’ is mapped as follows.

3.5.4.1.1 The IP address (or host name) and TCP port number of the MAL header field ‘URI To’ shall be assigned to ‘Host’ message header field.

3.5.4.1.2 If the MAL header field ‘URI To’ contains a Source Id, then this identifier preceded with a ‘/’ shall be assigned to the ‘request-target’ field of the MAL HTTP request-line.

3.5.4.1.3 If the MAL header field ‘URI To’ does not contain a Source Id, then ‘/’ shall be assigned to the ‘request-target’ field of the MAL HTTP request-line.

3.5.4.2 If the HTTP MAL application is being used to route to another MAL node, where the final destination of the MAL message as specified in the MAL header field ‘URI To’ does not coincide with the HTTP destination endpoint, the MAL header field ‘URI To’ shall be assigned to ‘X-MAL-URI-To’ HTTP message header field.

3.5.4.3 The content of the ‘X-MAL-URI-To’ HTTP message header field shall be encoded according to the rules defined in reference [9], section 2.

3.5.4.3.1 The IP address (or host name) and TCP port number of the HTTP destination endpoint shall be assigned to ‘Host’ message header field.

3.5.4.3.2 The request-line is implementation-specific.

3.5.5 TIMESTAMP

3.5.5.1 The MAL header field ‘Timestamp’ shall be assigned to the MAL HTTP message header field ‘X-MAL-Timestamp’.

3.5.5.2 The HTTP message header field ‘X-MAL-Timestamp’ shall contain the time in CCSDS ASCII Calendar Segmented Time Code format (reference [5]), code B (Year/Day of Year variation), with a precision up to milliseconds, and without the optional time code terminator (Z character).

3.5.6 QOSLEVEL

3.5.6.1 The MAL header field ‘QoSlevel’ shall be assigned to the MAL HTTP message header field ‘X-MAL-QoSlevel’.

3.5.6.2 The enumeration literal value of the field shall be assigned to the HTTP message header field ‘X-MAL- QoSlevel’, encoded as ASCII string as specified in table 3-7.

Table 3-7: QoS Level Encoding

MAL	Encoded ASCII String
BESTEFFORT	BESTEFFORT
ASSURED	ASSURED
QUEUED	QUEUED
TIMELY	TIMELY

NOTE – For instance:

```
X-MAL-QoSlevel: BESTEFFORT
```

3.5.7 PRIORITY

3.5.7.1 The MAL header field ‘Priority’ shall be assigned to the MAL HTTP message header field ‘X-MAL-Priority’, encoded as a decimal-formatted ASCII string without ‘0’ padding.

NOTE – For instance:

```
X-MAL-Priority: 10
```

3.5.8 DOMAIN

3.5.8.1 The MAL header field ‘Domain’ shall be assigned to the MAL HTTP message header field ‘X-MAL-Domain’.

3.5.8.2 The identifiers part of the MAL header field ‘Domain’ shall be encoded as an ASCII string containing dot-separated values.

3.5.8.3 The identifiers part of the MAL header field ‘Domain’ shall be encoded in the same order as they appear in the MAL list.

3.5.8.4 Each identifier part of the MAL header field ‘Domain’ that contains characters not present in the US-ASCII table or using a different charset shall be encoded using the MIME encoding rules specified in reference [10].

NOTE – For instance, the following encodings are all equivalent:

```
X-MAL-Domain: domainA.domainB

X-MAL-Domain: domainA.=?US-ASCII?Q?domain=20B?=

X-MAL-Domain: domainA.=?UTF-8?B?ZG9tYWluQg==?=
```

3.5.9 NETWORK ZONE

3.5.9.1 The MAL header field ‘Network Zone’ shall be assigned to the MAL HTTP message header field ‘X-MAL-Network-Zone’.

3.5.9.2 If the MAL header field ‘Network Zone’ contains characters not present in the US-ASCII table or using a different charset, then it shall be encoded using the MIME encoding rules specified in reference [10].

NOTE – For instance, the following encodings are all equivalent:

```
X-MAL-Network-Zone: network zone A

X-MAL-Network-Zone: =?US-ASCII?Q?network=20zone=20A?=

X-MAL-Network-Zone: =?UTF-8?B?bmV0d29yayB6b25lIEE=?=
```

3.5.10 SESSION

3.5.10.1 The MAL header field ‘Session’ shall be assigned to the HTTP message header field ‘X-MAL-Session’.

3.5.10.2 The enumeration literal value of the field shall be assigned to the HTTP message header field ‘X-MAL-Session’, encoded as an ASCII string as specified in table 3-8.

Table 3-8: Session Encoding

MAL	Encoded ASCII String
LIVE	LIVE
SIMULATION	SIMULATION
REPLAY	REPLAY

NOTE – For instance:

```
X-MAL-Session: REPLAY
```

3.5.11 SESSION NAME

3.5.11.1 The MAL header field ‘Session Name’ shall be assigned to the HTTP message header field ‘X-MAL-Session-Name’.

3.5.11.2 If the MAL header field ‘Session Name’ contains characters not present in the US-ASCII table or using a different charset, then it shall be encoded using the MIME encoding rules specified in reference [10].

NOTE – For instance, the following encodings are all equivalent:

```
X-MAL-Session-Name: session A

X-MAL-Session-Name: =?US-ASCII?Q?session=20A?=

X-MAL-Session-Name: =?UTF-8?B?c2Vzc2lvbiBB?=
```

3.5.12 INTERACTION TYPE

3.5.12.1 The MAL header field 'Interaction Type' shall be assigned to the HTTP message header field 'X-MAL-Interaction-Type'.

3.5.12.2 The enumeration literal value of the field shall be assigned to the HTTP message header field 'X-MAL-Interaction-Type', encoded as an ASCII string as specified in table 3-9.

Table 3-9: Interaction Type Encoding

MAL	Encoded ASCII String
SEND	SEND
SUBMIT	SUBMIT
REQUEST	REQUEST
INVOKE	INVOKE
PROGRESS	PROGRESS
PUBSUB	PUBSUB

NOTE – For instance:

```
X-MAL-Interaction-Type: SEND
```

3.5.13 INTERACTION STAGE

The value of the MAL header field 'Interaction Stage' shall be assigned to the MAL HTTP message header field 'X-MAL-Interaction-Stage', encoded as a decimal-formatted ASCII string, without '0' padding.

3.5.14 TRANSACTION ID

The value of the MAL header field 'Transaction Id' shall be assigned to the MAL HTTP message header field 'X-MAL-Transaction-Id', encoded as a decimal-formatted ASCII string, without '0' padding.

3.5.15 SERVICE AREA

The value of the MAL header field 'Service Area' shall be assigned to the MAL HTTP message header field 'X-MAL-Service-Area', encoded as a decimal-formatted ASCII string, without '0' padding.

3.5.16 SERVICE

The value of the MAL header field ‘Service’ shall be assigned to the MAL HTTP message header field ‘X-MAL-Service’, encoded as a decimal-formatted ASCII string, without ‘0’ padding.

3.5.17 OPERATION

The value of the MAL header field ‘Operation’ shall be assigned to the MAL HTTP message header field ‘X-MAL-Operation’, encoded as a decimal-formatted ASCII string, without ‘0’ padding.

3.5.18 AREA VERSION

The value of the MAL header field ‘Area Version’ shall be assigned to the MAL HTTP message header field ‘X-MAL-Area-Version’, encoded as a decimal-formatted ASCII string, without ‘0’ padding.

3.5.19 IS ERROR MESSAGE

The value of the MAL header field ‘Is Error Message’ shall be assigned to the MAL HTTP message header field ‘X-MAL-Is-Error-Message’. Allowed values are ‘True’ or ‘False’.

NOTE – For instance:

```
X-MAL-Is-Error-Message: False
```

3.6 MAL HTTP SPECIFIC FIELDS

3.6.1 OVERVIEW

The following subsections specify the values to be assigned to the MAL HTTP header fields that are not the result of the MAL header mapping.

3.6.2 VERSION NUMBER

3.6.2.1 The HTTP header field ‘X-MAL-Version-Number’ shall identify the structure of the MAL HTTP header as defined by this Recommended Standard.

3.6.2.2 The HTTP header field ‘X-MAL-Version-Number’ shall be assigned with the value ‘1’.

3.6.2.3 The version number ‘1’ shall be added to the SANA registry ‘MAL HTTP Binding Version Number’ and shall refer to the Mission Operations HTTP Transport and XML Encoding document ‘CCSDS 524.3-B-1.

3.6.3 CONTENT TYPE

3.6.3.1 The HTTP header field ‘Content-Type’ shall identify the encoding rules used to encode the MAL HTTP message body data.

3.6.3.2 The HTTP header field ‘Content-Type’ shall be assigned with the value ‘application/mal-xml’, and the ‘X-MAL-Encoding’ field shall be left out, when the MAL HTTP message body is encoded using the XML encoding defined by this Recommended Standard in section 5. (For alternate encodings, see 3.6.5.)

3.6.4 CONTENT LENGTH

The length in octets of the encoded MAL HTTP message body shall be assigned to the HTTP header field ‘Content-Length’, encoded as a decimal-formatted ASCII string.

3.6.5 SPECIFYING AN ALTERNATE ENCODING

3.6.5.1 The HTTP header field ‘Content-Type’ shall be assigned with the value ‘application/mal’ when the MAL HTTP message body is not encoded using the XML encoding defined by this Recommended Standard in section 5.

3.6.5.2 If the HTTP header field ‘Content-Type’ is assigned with the value ‘application/mal’, then the presence of the HTTP header field ‘X-MAL-Encoding’ is mandatory.

3.6.5.3 The HTTP header field ‘X-MAL-Encoding’ shall identify the encoding of the MAL HTTP body and be assigned with the value corresponding to the encoding used to encode the MAL body.

3.6.5.4 The value assigned to the HTTP header field ‘X-MAL-Encoding’ shall be encoded as a decimal-formatted ASCII string.

3.7 MAL MESSAGE BODY MAPPING

3.7.1 OVERVIEW

The following subsections specify how the MAL message body is mapped to the MAL HTTP message body.

Subsection 3.7.3 specifies an encoding format that can be used to encode the MAL HTTP message body.

NOTE – This Recommended Standard does not define the adoption of the encoding format specified in 3.7.3 as the mandatory encoding format to be adopted for the MAL HTTP Transport binding; others may be used.

3.7.2 BODY MAPPING

3.7.2.1 The MAL message body shall be encoded using the selected encoding format and assigned to the MAL HTTP message body.

3.7.2.2 The length in octets of the encoded MAL HTTP message body shall be assigned to the MAL HTTP message header field ‘Content-Length’.

3.7.2.3 The identifier of the selected encoding format for the MAL message body shall be assigned to the MAL HTTP message header fields ‘Content-Type’ and ‘X-MAL-Encoding’, according to the mapping rules specified in 3.6.3 and 3.6.5.

3.7.3 BODY ENCODING

3.7.3.1 Overview

This subsection specifies how the MAL message body is encoded using the XML Encoding.

The encoding formats are defined in this subsection. The encoding formats for data are in section 5. The encoding provides an unambiguous one-to-one mapping from MAL service definition to XML Schema (specified in references [7] and [8]), and allows validation of MAL messages against an XML schema.

NOTE – The body encoding format can be re-used by a MAL binding to a messaging technology that is not HTTP.

3.7.3.2 Body

3.7.3.2.1 The XML namespace ‘http://www.ccsds.org/schema/malxml/MAL’ with prefix ‘malxml’ shall be used as target namespace for the definition of all XML types, elements and attributes included in the encoded MAL message body, defined as part of this standard.

3.7.3.2.2 The MAL message body shall be encoded using the malxml ‘Body’ element, defined by the following XSD Schema definition:

```
<element name="Body" type="malxml:Body" />
<complexType name="malxml:Body">
 <sequence>
   <any processContents="lax" minOccurs="0" maxOccurs="unbounded" />
 </sequence>
</complexType>
```

3.7.3.2.3 The malxml:Body element shall contain each body element of the MAL message body in the same order as it is declared in the operation definition.

3.7.3.2.4 A MAL body element defined as MAL Element shall be encoded using the MAL Element encoding rule (5.3).

3.7.3.2.5 The name of the XML element for body elements defined as MAL Element shall be set equal to the name of the MAL Element to be encoded.

3.7.3.2.6 A MAL Body element defined as List of MAL elements shall be encoded using the MAL List encoding rule (5.7).

3.7.3.2.7 The name of the XML element for body elements defined as MAL List shall be set equal to the name of the MAL Element enclosed by the list, suffixed with the word ‘List’.

3.7.3.2.8 The XML encoded MAL Body shall be preceded by the following XML Declaration:

```
<?xml version="1.0" encoding="UTF-8"?>
```

3.7.3.2.9 The root element of the XML encoded MAL Body shall be the MAL Body element.

4 MAL TRANSPORT INTERFACE MAPPING

4.1 OVERVIEW

The MAL specification (reference [1]) ‘Transport Interface’ section defines the interface to be provided by the MAL HTTP Transport binding layer. The following subsections specify the expected behaviour for each of the MAL transport interface request and indication primitives. If an indication is a response to a request, then the behaviour of the indication is specified in the same subsection as the request.

The following primitives are defined in the MAL transport interface and need to be provided by the MAL HTTP Transport binding layer:

a) SUPPORTEDQOS request;

b) SUPPORTEDQOS RESPONSE indication;

c) SUPPORTEDIP request;

d) SUPPORTEDIP RESPONSE indication;

e) TRANSMIT request;

f) TRANSMIT ACK indication;

g) TRANSMIT ERROR indication;

h) TRANSMITMULTIPLE request;

i) TRANSMITMULTIPLE ACK indication;

j) TRANSMITMULTIPLE ERROR indication;

k) RECEIVE indication;

l) RECEIVEMULTIPLE indication.

The MAL primitives with their parameters are listed in table 4-1.

The following primitives defined by HTTP (references [2] and [3]) are used by the mapping:

a) ‘HTTP REQUEST - POST;

b) ‘HTTP RESPONSE’.

The HTTP primitives with their parameters are listed in table 4-2.

Table 4-1: MAL Transport Interface Primitives

Primitive	**Parameters**
SUPPORTEDQOS request	QoS Level
SUPPORTEDQOS RESPONSE indication	Boolean
SUPPORTEDIP request	Interaction Type
SUPPORTEDIP RESPONSE indication	Boolean
TRANSMIT request	MAL Message QoS Properties
TRANSMIT ACK indication	
TRANSMIT ERROR Indication	MAL Message Header Error Number Extra Information QoS Properties
TRANSMITMULTIPLE request	List of: 1 MAL Message 2 QoS Properties
TRANSMITMULTIPLE ACK indication	
TRANSMITMULTIPLE ERROR indication	List of: a) MAL Message Header b) Error Number c) Extra Information d) QoS Properties
RECEIVE indication	MAL Message QoS Properties
RECEIVEMULTIPLE indication	List of: a) MAL Message b) QoS Properties

Table 4-2: HTTP Interface Primitives

Primitive	**Parameters**
HTTP REQUEST - POST	HTTP request-line, header, and body
HTTP RESPONSE	HTTP status-line, header, and body

4.2 SUPPORTEDQOS REQUEST

4.2.1 The SUPPORTEDQOS request primitive shall be provided.

4.2.2 Support for the Quality of Service (QoS) levels defined by MAL shall depend on the capabilities of the underlying layer used to convey the HTTP messages. For example, if TCP/IP is used as transport protocol, then QoS depends on the capability of TCP/IP.

4.3 SUPPORTEDIP REQUEST

4.3.1 The SUPPORTEDIP request primitive shall be provided.

4.3.2 The SUPPORTEDIP request primitive shall return TRUE for the interaction patterns SEND, SUBMIT, REQUEST, INVOKE, and PROGRESS.

4.3.3 The SUPPORTEDIP request primitive shall return FALSE for the interaction pattern PUBLISH-SUBSCRIBE.

4.3.4 The MAL layer shall support PUBLISH-SUBSCRIBE itself.

NOTE – The MAL specification (reference [1]) requires that implementations of the MAL layer support the Publish-Subscribe pattern, but that they can delegate this pattern to a transport that supports the pattern natively. The HTTP protocol does not support the Publish-Subscribe pattern natively; therefore a MAL implementation has to support this pattern itself.

4.4 TRANSMIT REQUEST

4.4.1 The TRANSMIT request primitive shall be provided in order to translate a MAL message into one MAL HTTP request message and send it by calling the HTTP primitive ‘REQUEST POST’.

4.4.2 If any of the MAL header fields is NULL, then the TRANSMIT ERROR primitive shall be called with the error number MAL::INTERNAL.

4.4.3 The MAL message header fields and body elements shall be mapped to the MAL HTTP message according to the specification given in section 3 of this Recommended Standard.

4.4.4 If either of the fields ‘URI From’ or ‘URI To’ is not compliant with the URI format defined in 3.4, then the TRANSMIT ERROR primitive shall be called with the error number MAL::INTERNAL.

4.4.5 If an error is returned by the invocation of the HTTP ‘REQUEST POST’ primitive, then the TRANSMIT ERROR primitive shall be called with the error number MAL::INTERNAL.

4.4.6 If the invocation of the HTTP 'REQUEST POST' primitive successfully returns, then the TRANSMIT ACK primitive shall be called.

4.5 TRANSMITMULTIPLE REQUEST

4.5.1 The TRANSMITMULTIPLE request primitive shall be provided by calling the TRANSMIT request primitive for every MAL message.

4.5.2 If the TRANSMIT ERROR indication is called for any of the MAL messages, the TRANSMIT ERROR indications should be collected, and the TRANSMITMULTIPLE ERROR indication should be called with the content of the collected TRANSMIT ERROR indications.

4.5.2.1 The individual TRANSMIT ERROR indications shall not be transmitted to MAL.

4.5.2.2 Only the TRANSMITMULTIPLE ERROR indication shall be called.

4.6 RECEIVE INDICATION

4.6.1 The RECEIVE indication primitive shall be provided in order to receive one MAL HTTP message response and translate it into a MAL message.

4.6.2 The RECEIVE indication primitive shall be called once a complete MAL HTTP message response is received.

4.6.3 The HTTP 'RESPONSE' primitive shall be called once for each HTTP 'REQUEST POST' sent.

4.6.4 The MAL message header fields and body elements shall be generated according to the specifications given in section 3 of this Recommended Standard, by using the following input data:

a) the MAL HTTP message;

b) the XML specification of the MO service (see section 6 of reference [1]) identified by the MAL header fields 'Service Area', 'Service', and 'Area Version'.

4.6.5 If the field 'URI To' is unknown, then the error MAL::DESTINATION_UNKNOWN shall be returned if the Interaction Pattern allows a MAL error message to be returned. The MAL header field 'URI From' of the returned error message shall be assigned with the 'URI To' field of the initial message, even if this URI is unknown.

4.7 RECEIVEMULTIPLE INDICATION

The RECEIVEMULTIPLE indication primitive shall not be provided.

5 MAL DATA ENCODING

5.1 OVERVIEW

This section specifies a complete and unambiguous mapping of the MAL data types to XML Schema types.

The encoding is performed along the following **fundamental rules**:

a) A MAL composite is mapped to a XSD complex type;

b) A MAL attribute is mapped to a XSD complex type;

c) A MAL abstract composite is mapped to a XSD abstract complex type;

d) A MAL composite field is mapped to a XSD element in a sequence belonging to the XSD complex type related to the MAL composite;

e) A MAL nullable field is mapped to a XSD element with 'nillable' attribute set to 'true';

f) A list of MAL elements is mapped to a XSD complex type, defined as an XSD sequence of a single element, having the 'minOccurs' attribute set to '0' and the 'maxOccurs' attribute set to 'unbounded';

The encoding for **attribute** is performed along the following rules:

a) A MAL attribute is a complex type, which inherit from malxml XSD Complex Type 'Attribute';

b) Every attribute has a single element called with the same name of the Attribute name and having a specific data type;

c) When possible, natively supported XSD data types (defined in reference [8]) shall be used;

d) When not possible, ad-hoc data types shall be defined;

e) Enumeration types shall be encoded using restriction on XSD simple type 'string'.

The type of an element can designate either the declared type of the field the element is assigned to or the actual type of the element. In order to avoid any ambiguity, the word 'type' is always qualified as follows:

a) declared type: the type of the field the element is assigned to; if the field belongs to the MAL message header, then the declared type is given by table 3-1; if the field belongs to the MAL message body, then the declared type and more generally the declaration context is given by the XML specification of the service (see section 6 of reference [1]);

b) actual type: the type of the element.

MAL only specifies non-abstract types that are final, i.e., that cannot be extended. As a consequence, if the declared type of an element is non-abstract, then the actual type is the same as the declared type.

The following subsections specify the rules to be applied when encoding an element. These rules depend on the element declaration context (e.g., the declared type) and the element actual type.

NOTE – The encoding format shall be flexible in the use or lack thereof of carriage returns, line endings, and whitespaces.

The EntityRequest structure is a composite type defined inside the MAL area, as specified in reference [1]. Its definition is specified as follows:

```
<mal:composite name="EntityRequest" shortFormPart="24">
  <mal:extends>
    <mal:type name="Composite" area="MAL"/>
  </mal:extends>
  <mal:field name="subDomain">
    <mal:type name="Identifier" area="MAL" list="true"/>
  </mal:field>
  <mal:field name="allAreas" canBeNull="false">
    <mal:type name="Boolean" area="MAL"/>
  </mal:field>
  <mal:field name="allServices" canBeNull="false">
    <mal:type name="Boolean" area="MAL"/>
  </mal:field>
  <mal:field name="allOperations" canBeNull="false">
    <mal:type name="Boolean" area="MAL"/>
  </mal:field>
  <mal:field name="onlyOnChange" canBeNull="false">
    <mal:type name="Boolean" area="MAL"/>
  </mal:field>
  <mal:field name="entityKeys" canBeNull="false">
    <mal:type name="EntityKey" area="MAL" list="true"/>
  </mal:field>
</mal:composite>
```

According to the rules defined in this section, the EntityRequest type shall be encoded as a complex type as follows:

```
<xsd:complexType name="EntityRequest">
  <complexContent>
          <extension base="malxml:Composite" />
  </complexContent>
  <sequence>
    <element name="subDomain" type="malxml:IdentifierList" nillable="true" />
    <element name="allAreas" type="malxml:Boolean" />
    <element name="allServices" type="malxml:Boolean" />
    <element name="allOperations" type="malxml:Boolean" />
    <element name="onlyOnChange" type="malxml:Boolean" />
    <element name="entityKeys" type="malxml:EntityKeyList" />
  </sequence>
</complexType>
```

An example of an XML fragment derived from the complex type is reported as follows, where the target namespace is ‘http://www.ccsds.org/schema/malxml/MAL’:

```
<...>
  <subDomain>
          <Identifier><Identifier>Id1</Identifier></Identifier>
          <Identifier><Identifier>Id2</Identifier></Identifier>
          <Identifier xsi:nil="true" />
  </subDomain>
  <allAreas><Boolean>true</Boolean></allAreas>
  <allServices><Boolean>true</Boolean></allServices>
  <allOperations><Boolean>true</Boolean></allOperations>
  <onlyOnChange><Boolean>true</Boolean></onlyOnChange>
  <entityKeys>
    <EntityKey>
      <firstSubKey>
        <Identifier>IDK1</Identifier>
      </firstSubKey>
      <secondSubKey>
        <Long>0</Long>
      </secondSubKey>
      <thirdSubKey xsi:nil="true" />
      <fourthSubKey xsi:nil="true" />
    </EntityKey>
    <EntityKey xsi:nil="true" />
  </entityKeys>
</...>
```

5.2 GENERAL

5.2.1 The XML namespace 'urn:ccsds:schema:mo:malxml' with prefix 'malxml' shall be used as the target namespace for the definition of all XML types, elements, and attributes included in the encoded MAL message body, defined as part of this Recommended Standard and belonging to the 'MAL' area.

5.2.2 XML types defined in the context of MAL areas, services, and operations shall be assigned to a different XML namespace.

5.2.3 The XML namespace for XML types not defined in this standard shall be constructed using the following rules:

a) the namespace shall begin with 'http://www.ccsds.org/schema/malxml';

b) if the XML types is defined as part of an area, then the namespace shall be extended with a '/' and with the area name;

c) if the XML types is defined as part of a service, then the area-extended namespace shall be extended with a '/' and with the service name;

NOTE – As an example, the type QueryFilter, defined in the context of the Archive service part of the COM area, shall belong to the namespace 'http://www.ccsds.org/schema/malxml/COM/Archive'.

5.2.4 An abstract empty XML Complex Type 'malxml:Element' shall serve as the base for the extension of XML Complex Type used for MAL elements.

5.2.5 The XML Complex Type 'malxml:Element' shall be defined as follows:

```
<complexType name="Element" abstract="true" />
```

5.2.6 An abstract empty XML Complex Type 'malxml:Attribute' shall serve as the base for the extension of XML Complex Type used for MAL attributes.

5.2.7 The XML Complex Type 'malxml:Attribute' shall be defined as follows:

```
<complexType name="Attribute" abstract="true">
 <complexContent>
        <extension base="malxml:Element" />
 </complexContent>
</complexType>
```

5.2.8 An abstract empty XML Complex Type 'malxml:Composite' shall serve as the base for the extension of XML Complex Type used for MAL composite.

5.2.9 The XML Complex Type ‘malxml:Composite’ shall be defined as follows:

```
<complexType name="Composite" abstract="true">
  <complexContent>
        <extension base="malxml:Element" />
  </complexContent>
</complexType>
```

5.3 ELEMENT

5.3.1 A MAL::Element shall be encoded as follows:

a) if the actual type of the element is a MAL::Attribute, then the element shall be encoded as specified by its actual type;

b) if the actual type of the element is a MAL::Enumeration, then the element shall be encoded as an Enumeration;

c) if the actual type of the element is a MAL::List, then the element shall be encoded as a List;

d) if the actual type of the element is a MAL::Composite, then the element shall be encoded as a Composite.

5.4 ATTRIBUTE

5.4.1 A MAL attribute shall be encoded as an XML element, with type extending the XML Complex Type ‘malxml:Attribute’.

5.4.2 If the MAL attribute is defined as part of an area or service, the corresponding XML Complex Type shall belong to the namespace associated to the area or service.

5.4.3 The name of the XML Complex Type of the MAL attribute shall be equal to the name of the MAL attribute it maps.

5.4.4 The XML Complex Type of the MAL attribute shall be defined as an XML sequence containing a single element:

a) the ‘name’ attribute of the XML element shall be set equal to the MAL attribute name;

b) the ‘type’ attribute of the XML element shall be set equal to the XML type associated to the actual attribute type.

NOTE – The actual attribute type can be a predefined XML simple data type, an enumeration, or an XML complex type.

5.5 ENUMERATION

5.5.1 A MAL enumeration shall be encoded as an XML element, with type extending the XML Complex Type ‘malxml:Element’.

5.5.2 The name of the XML Complex Type of the MAL enumeration shall be equal to the name of the MAL enumeration.

5.5.3 In order to represent the MAL enumeration values, an XML Simple Type restriction of the simple type ‘xsd:string’ shall be defined.

5.5.4 The restriction shall include all the MAL enumeration values in the order provided in the MAL enumeration definition.

5.5.5 The name of the XML Simple Type restriction shall be equal to the name of the MAL enumeration, plus the suffix ‘Enum’.

5.5.6 If the MAL enumeration is defined as part of an area, service, or operation, the corresponding XML Complex Type and restricted XML Simple Type shall belong to the namespace associated to the area, service, or operation.

5.5.7 The XML Complex Type of the MAL enumeration shall be defined as an XML sequence containing a single element:

a) the ‘name’ attribute of the XML element shall be set equal to the name of the MAL enumeration;

b) the ‘type’ attribute of the XML element shall be set equal to the restricted XML Simple Type used to define the enumeration.

NOTES

1 Each element in an enumeration is assigned with one enumeration string value and two integer values: the ordinal value and the numeric value. The ordinal value is a sequential counter, starting at zero for the first element, and incremented by one in the same order as the elements of the enumeration. The numeric value is defined in reference [1], and it is not reported in the type definition.

2 As an example, the MAL Enumeration ‘SessionType’ is encoded using the following type definitions:

```
<complexType name="SessionType">
<complexContent>
         <extension base="malxml:Element">
    <sequence>
     <element name="SessionType" type="malxml:SessionTypeEnum" />
    </sequence>
   </extension>
</complexContent>
</complexType>
```

```
<simpleType name="SessionTypeEnum">
        <restriction base="xsd:string">
   <enumeration value="LIVE" />
   <enumeration value="SIMULATION" />
   <enumeration value="REPLAY" />
  </restriction>
</simpleType>
```

5.6 COMPOSITE

5.6.1 A MAL composite shall be encoded as an XML element, with type extending the XML Complex Type 'malxml:Composite'.

5.6.2 If the MAL composite inherits from another MAL composite, then the type of the XML element shall extend the inherited MAL composite's XML Complex Type.

5.6.3 If the MAL composite is defined as part of an area, service, or operation, the corresponding XML Complex Type shall belong to the namespace associated to the area, service, or operation.

5.6.4 If the MAL composite is marked as 'abstract', then the corresponding XML Complex Type shall be marked as 'abstract'.

5.6.5 The name of the XML Complex Type of the MAL composite shall be equal to the name of the MAL composite it maps.

5.6.6 Each field of the MAL composite shall be encoded as an XML element of the sequence defining the XML Complex Type of the composite, in the same order as it is declared in the MAL composite definition, as follows:

a) the 'name' attribute of the XML element shall be set equal to the name of the composite field, without blank spaces between words (if any);

b) the 'type' attribute of the XML element shall be set equal to the XML Complex Type associated to the field type;

c) if the field is nullable, then the 'nillable' attribute shall be set equal to 'true'.

5.6.7 The XML element representing the MAL composite shall contain an attribute named 'malxml:type', whose value is equal to the short-form-part of the MAL composite it is representing.

5.6.8 The XML Complex Type of the MAL Composite shall contain an 'xsd:attribute' element defining the 'malxml:type' attribute of the XML Element as follows:

a) the 'name' attribute of the XML element shall be set to 'malxml:type';

b) the 'type' attribute of the XML element shall be set equal to 'xsd:long'.

5.7 LIST

5.7.1 A MAL list shall be encoded as an XML element, with type extending the XML Complex Type ‘malxml:Composite’.

5.7.2 If the MAL list is defined as part of an area or service, the corresponding XML Complex Type shall belong to the namespace associated to the area or service.

5.7.3 The name of the XML Complex Type of the MAL list shall be equal to the name of the MAL element it contains, plus the suffix ‘List’.

5.7.4 The XML Complex Type of the MAL list shall be defined as an XML sequence containing a single element:

a) the ‘name’ attribute of the XML element shall be set equal to the name of the MAL element enclosed by the list;

b) the ‘type’ attribute of the XML element shall be set equal to the corresponding XML Complex Type of the MAL element enclosed by the list;

c) the ‘minOccurs’ attribute shall be set equal to ‘0’;

d) the ‘maxOccurs’ attribute shall be set equal to ‘unbounded’;

e) the ‘nullable’ attribute shall be set equal to ‘true’.

5.7.5 The list elements shall be encoded in the same order as in the list.

5.8 BLOB

5.8.1 A MAL::Blob shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Blob">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Blob" type="xsd:hexBinary" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.8.2 The Blob octets shall be encoded in the same order as in the Blob.

5.9 BOOLEAN

A MAL::Boolean shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Boolean">
 <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Boolean" type="xsd:boolean" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.10 DURATION

A MAL::Duration shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Duration">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Duration" type="xsd:duration" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.11 FLOAT

A MAL::Float shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Float">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Float" type="xsd:float" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.12 DOUBLE

A MAL::Double shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Double">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Double" type="xsd:double" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.13 IDENTIFIER

A MAL::Identifier shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Identifier">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Identifier" type="xsd:string" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.14 OCTET

A MAL::Octet shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Octet">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Octet" type="xsd:byte" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.15 UOCTET

A MAL::UOctet shall be encoded as defined by the following XML Complex Type:

```
<complexType name="UOctet">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="UOctet" type="xsd:unsignedByte" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.16 SHORT

A MAL::Short shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Short">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Short" type="xsd:short" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.17 USHORT

A MAL::UShort shall be encoded as defined by the following XML Complex Type:

```
<complexType name="UShort">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="UShort" type="xsd:unsignedShort" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.18 INTEGER

A MAL::Integer shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Integer">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Integer" type="xsd:int" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.19 UINTEGER

A MAL::UInteger shall be encoded as defined by the following XML Complex Type:

```
<complexType name="UInteger">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="UInteger" type="xsd:unsignedInt" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.20 LONG

A MAL::Long shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Long">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Long" type="xsd:long" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.21 ULONG

A MAL::ULong shall be encoded as defined by the following XML Complex Type:

```
 <complexType name="ULong">
   <complexContent>
         <extension base="malxml:Attribute">
    <sequence>
     <element name="ULong" type="xsd:unsignedLong" />
    </sequence>
   </extension>
   </complexContent>
</complexType>
```

5.22 STRING

A MAL::String shall be encoded as defined by the following XML Complex Type:

```
<complexType name="String">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="String" type="xsd:string" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.23 TIME

A MAL::Time shall be encoded as defined by the following XML Complex Type:

```
<complexType name="Time">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="Time" type="xsd:dateTime" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.23.1 The resolution of the Time complexType represented by the xsd:dateTime element shall be as specified in reference [1].

5.23.2 The format used by the xsd:dateTime element shall be:

```
CCYY-MM-DDThh:mm:ss.sss
```

5.24 FINETIME

A MAL::FineTime shall be encoded as defined by the following XML Complex Type:

```
<complexType name="FineTime">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="FineTime" type="xsd:dateTime" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

5.24.1 The resolution of the FineTime complexType represented by the xsd:dateTime element shall be in nanoseconds.

5.24.2 The maximum number of digits used for the fractional part of the second unit, used in the format, shall be nine (see reference [8]).

5.24.3 The format used by the xsd:dateTime element shall be:

```
 CCYY-MM-DDThh:mm:ss.sssssssss
```

5.25 URI

A MAL::URI shall be encoded as defined by the following XML Complex Type:

```
<complexType name="URI">
  <complexContent>
        <extension base="malxml:Attribute">
   <sequence>
    <element name="URI" type="xsd:anyURI" />
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

ANNEX A

PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT PROFORMA

(NORMATIVE)

A1 INTRODUCTION

A1.1 OVERVIEW

This annex provides the Protocol Implementation Conformance Statement (PICS) Requirements List (RL) for an implementation of the Mission Operations HTTP Transport and XML Encoding standard. The PICS for an implementation is generated by completing the RL in accordance with the instructions below. An implementation claiming conformance must satisfy the mandatory requirements referenced in the RL.

An implementation’s completed RL is called the PICS. The PICS states which protocol features have been implemented. The following entities can use the PICS:

- the protocol implementer, as a checklist to reduce the risk of failure to conform to the standard through oversight;
- the supplier and acquirer or potential acquirer of the implementation, as a detailed indication of the capabilities of the implementation, stated relative to the common basis for understanding provided by the standard PICS proforma;
- the user or potential user of the implementation, as a basis for initially checking the possibility of interworking with another implementation (while interworking can never be guaranteed, failure to interwork can often be predicted from incompatible PICSes);
- a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for conformance of the implementation.

A1.2 NOTATION

A1.2.1 Status Column Symbols

The following are used in the RL to indicate the status of features:

Symbol	Meaning
M	Mandatory
O	Optional

A1.2.2 Support Column Symbols

The support of every item as claimed by the implementer is stated by entering the appropriate answer (Y, N, or N/A) in the support column.

Symbol	Meaning
Y	Yes, supported by the implementation
N	No, not supported by the implementation
N/A	Not applicable

A2 GENERAL INFORMATION

A2.1 IDENTIFICATION OF PICS

Ref	Question	Response
1	Date of Statement (DD/MM/YYYY)	
2	CCSDS document number containing the PICS	
3	Date of CCSDS document containing the PICS	

A2.2 IDENTIFICATION OF IMPLEMENTATION UNDER TEST (IUT)

Ref	Question	Response
1	Implementation name	
2	Implementation version	
3	Machine name	
4	Machine version	
5	Operating System name	
6	Operating System version	
7	Special Configuration	
8	Other Information	

A2.3 USER IDENTIFICATION

Supplier	
Contact Point for Queries	
Implementation name(s) and Versions	
Other Information Necessary for full identification —e.g., name(s) and version(s) for machines and/or operating systems; System Name(s)	

A2.4 INSTRUCTIONS FOR COMPLETING THE RL

An implementer shows the extent of compliance to the protocol by completing the RL; the resulting completed RL is called a PICS.

A3 MO HTTP TRANSPORT AND XML ENCODING PICS

A3.1 MESSAGE ABSTRACTION LAYER

Item	Protocol Feature	Reference	Status	Support
1-1	Transaction Handling	[1] subsection 3.2	M	
1-2	State Transitions	[1] subsection 3.3	M	
1-3	Message Composition	[1] subsection 3.4	M	
1-4	MAL Service Interface	[1] subsection 3.5	M	
1-5	Access Control Interface	[1] subsection 3.6	M	
1-6	Transport Interface	[1] subsection 3.7	M	
1-7	MAL Data Type Specification	[1] section 4	M	
1-8	MAL Errors	[1] section 5	M	

A3.2 MAL MESSAGE MAPPING

Item	Protocol Feature	Reference	Status	Support
2-1	URI Format	3.4	M	
2-2	MAL Header Mapping	3.5	M	
2-3	Field ‘Timestamp’	3.5.5	M	
2-4	Fields ‘Priority’, ‘Domain’, ‘Network Zone’, ‘Session Name’	3.5.7 3.5.8 3.5.9 3.5.11	M	
2-5	Field ‘Authentication Id’	3.5.3	M	
2-6	MAL HTTP Specific Fields	3.6	M	
2-7	MAL Message Body Mapping	3.6.5	M	

A3.3 MAL TRANSPORT INTERFACE MAPPING

Item	Protocol Feature	Reference	Status	Support
3-1	SupportedQoS Request	4.2	M	
3-2	SupportedIP Request	4.3	M	
3-3	Transmit Request	4.4	M	
3-4	TransmitMultiple Request	4.5	M	
3-5	Receive Indication	4.6	M	
3-6	ReceiveMultiple Indication	4.7	M	

A3.4 MAL DATA ENCODING

Item	Protocol Feature	Reference	Status	Support
4-1	MAL Data Encoding	section 5	O	

ANNEX B

SECURITY, SANA, AND PATENT CONSIDERATIONS

(INFORMATIVE)

B1 SECURITY CONSIDERATIONS

B1.1 OVERVIEW

This annex subsection discusses various aspects of security with respect to the MAL HTTP Transport protocol.

It is strongly recommended that HTTPS be used to communicate the encoded HTTP messages.

B1.2 SECURITY BACKGROUND

The following security aspects are typically separated:

a) data and data origin authentication: corroboration of the source of information that is contained in a message;

b) authorization: conveyance, to another entity, of official sanction to do or be something;

c) confidentiality: keeping information secret from all but those who are authorized to see it;

d) integrity: detecting that information has not been altered by unauthorized or unknown means.

The MAL HTTP Transport protocol is not responsible for ensuring all these security aspects; however, it has to fulfil the security criteria expected by the MAL layer from every transport binding. These criteria are:

a) the Transport Layer is responsible for the transmission of the authentication identifier assigned by the MAL layer to every consumer;

b) the Transport Layer has to provide authentication, confidentiality, and integrity of the transmitted messages.

B1.3 SECURITY CONCERNS WITH RESPECT TO THE CCSDS DOCUMENT

B1.3.1 Data Privacy

It is assumed that message authentication and confidentiality are provided beneath the HTTP layer and are transparent to the MAL HTTP Transport binding and above. As a consequence, once a message rises above the HTTP protocol layer, the message has been authenticated and all encryption has been removed.

B1.3.2 Data Integrity

Integrity is required to be provided by the underlying transport protocol that conveys the HTTP messages. While underlying Transport Layer security is advised, it cannot protect the Application Layer fully, and Application Layer protection such as HTTPS should be implemented in addition.

B1.3.3 Authentication of Communicating Entities

Authentication of the consumers is done above the MAL layer through a specific service that enables a consumer to get an authentication identifier. The meaning of that authentication identifier is dependent on the security system used for the deployment. This identifier must allow the MAL access control implementation to perform a lookup for authorization purposes.

The authentication identifier is transmitted by the MAL HTTP Transport protocol in the parameter ‘Authentication Id’ of the MAL HTTP message; however, this parameter is optional and may be omitted.

The MAL authentication identifier is an implementation and technology specific security credential created at a higher layer by MAL access control.

HTTP provides a general framework for access control and authentication, via an extensible set of challenge-response authentication schemes, which can be used by a server to challenge a client request and by a client to provide authentication information (reference [4]). It is possible to use this mechanism for authentication, but that is not specified here.

B1.3.4 Control of Access to Resources

Authorization is done by the MAL access control that performs any required authorization checks and converts the consumer identifier into technology dependent security credentials.

B1.4 POTENTIAL THREATS AND ATTACK SCENARIOS

Many common potential threats and attack scenarios exist for HTTP (such as spoofing or interception), and these also apply here; but mitigation of those common threats can be performed using common HTTP protection approaches such as the application of secure communications and firewalls.

A discussion of all possible HTTP and XML attack vectors is outside the scope of this document; however, support for standard HTTP security such as secure sockets (which can provide security algorithms ensuring authentication, confidentiality, and integrity) is discussed in B1.3 and firewall considerations in B1.6.

B1.5 CONSEQUENCES OF NOT APPLYING SECURITY TO THE TECHNOLOGY

For the ground segment, unprotected transmission of MAL messages can result in many attack vectors that involve violation of integrity and/or confidentiality as listed in reference [D3]. It is therefore recommended that HTTPS be used and that the CCSDS Security Green Books (references [D2], [D3], and [D4]) be read.

B1.6 FIREWALL COMPATIBILITY

This annex subsection explicitly addresses firewall compatibility and does not recommend firewalls as the only line of defence.

The mapping of the MAL operations over the HTTP protocol, as specified by this Recommended Standard, requires the presence of an HTTP server located on the consumer side when using services defining operations with interaction pattern INVOKE, PROGRESS, or PUBSUB; an effect exists if either side is protected by a firewall (see figure B-1).

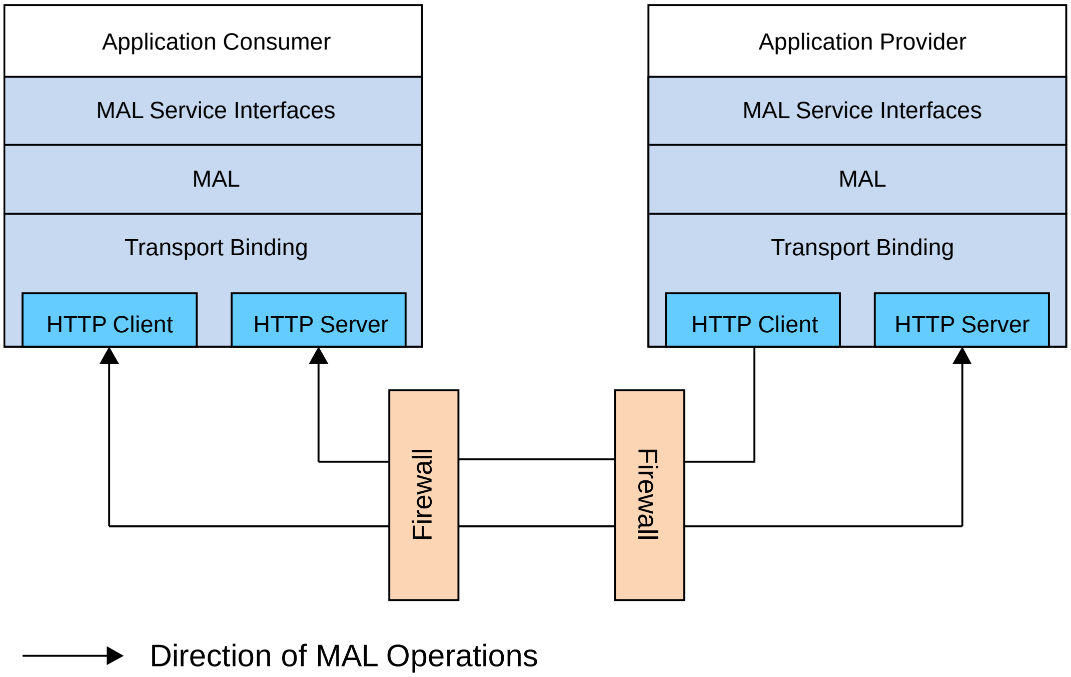

Figure B-1: Deployment Using the MAL HTTP Transport Binding

In the envisaged scenario, both parties are known to each other and have control over their respective firewalls, as modification of firewall rules is expected to allow HTTP connections in both directions.

As a consequence of the requirement for an HTTP server on the consumer side, this Recommended Standard is not suitable for deployment scenarios in which the configuration of the deployed firewalls is outside the control of the organizations involved in the service provision/consumption (see figure B-2).

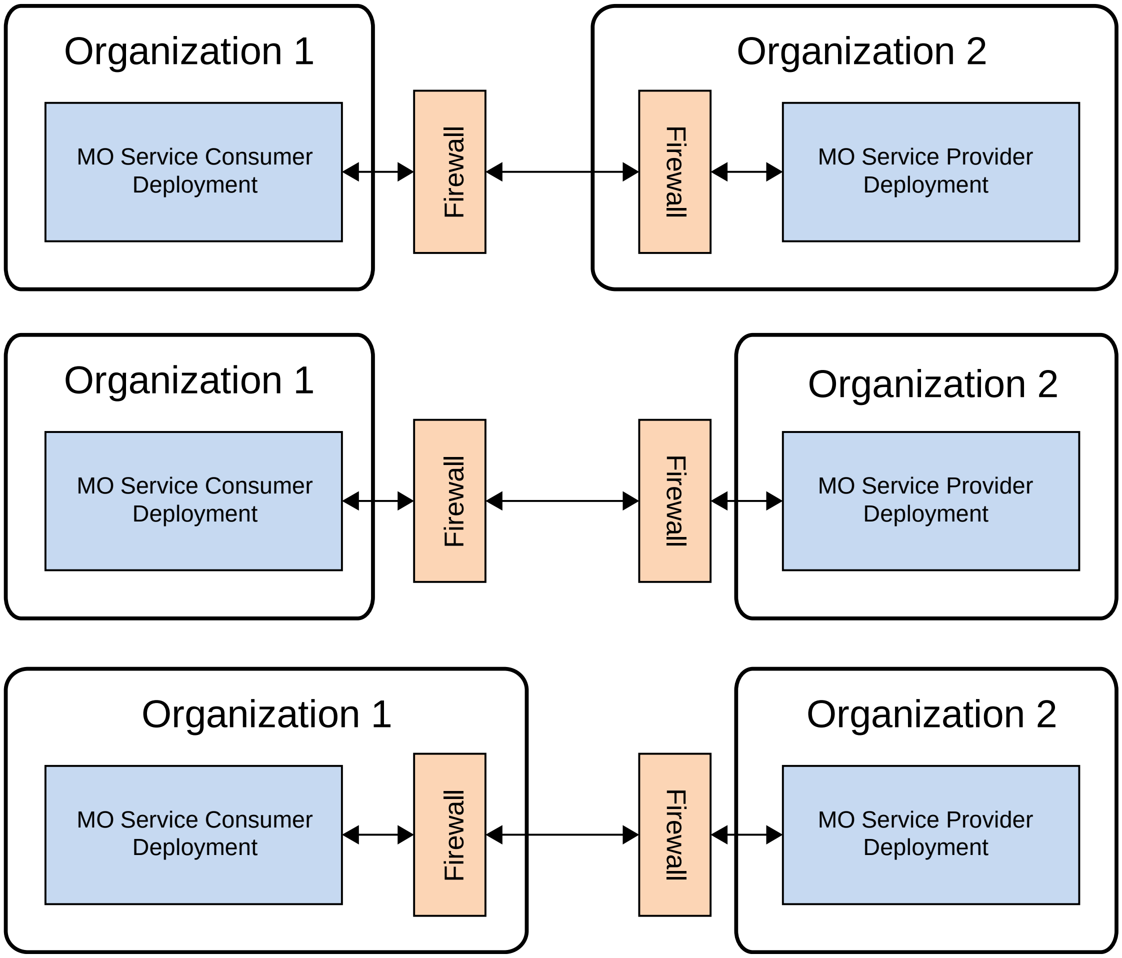
Organization 1
MO Service Consumer Deployment
Firewall
Firewall
Organization 2
MO Service Provider Deployment
Organization 1
MO Service Consumer Deployment
Firewall
Firewall
Organization 2
MO Service Provider Deployment
Organization 1
MO Service Consumer Deployment
Firewall
Firewall
Organization 2
MO Service Provider Deployment

Figure B-2: No Firewall Control

This Recommended Standard is also not suitable for deployment scenarios involving unknown consumers and a provider (typical Internet scenario) if the provided MO services contain operations with interaction pattern INVOKE, PROGRESS, and PUBSUB, because the return HTTP connection from the MAL provider to the MAL consumer will not be possible, in that it will likely be blocked by the consumer's firewall.

B2 SANA CONSIDERATIONS

This Recommended Standard does not require any specific use of the SANA registries, as the standard may be used without the involvement of SANA. However, if a provider of mission operations services using the encoding and transport defined in this standard wishes to make their capabilities and addressing information known and/or available publicly, then the use of standard, public SANA registries would be recommended.

The SANA Registry Management Policy (reference [D5]) provides comprehensive information about key SANA and other CCSDS registries and their relationships, and defines a consistent set of policies, rules, and procedures that can be applied to the creation, control, and management of the CCSDS-wide enterprise registries and the global and local registries in the SANA.

For the purposes of mission operations service provision, it is expected that the SANA Service Site and Aperture registries should be examined and used with complimentary contact information held in the Persons registry.

B3 PATENT CONSIDERATIONS

No patents are known to apply to this Recommended Standard.

ANNEX C

ACRONYMS

(INFORMATIVE)

This annex lists the acronyms used in this Recommended Standard.

ASCII	American Standard Code for Information Interchange
CCSDS	Consultative Committee for Space Data Systems
CRC	cyclic redundancy check
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
MAL	Message Abstraction Layer
MSB	most significant bit
PDU	protocol data unit
QoS	quality of service
SANA	Space Assigned Number Authority
SM&C	Spacecraft Monitoring and Control
TCP	Transmission Control Protocol
URI	Universal Resource Identifier

ANNEX D

INFORMATIVE REFERENCES

(INFORMATIVE)

[D1] *Mission Operations Services Concept*. Issue 3. Report Concerning Space Data System Standards (Green Book), CCSDS 520.0-G-3. Washington, D.C.: CCSDS, December 2010.

[D2] *The Application of CCSDS Protocols to Secure Systems*. Issue 2. Report Concerning Space Data System Standards (Green Book), CCSDS 350.0-G-2. Washington, D.C.: CCSDS, January 2006.

[D3] *Security Threats against Space Missions*. Issue 2. Report Concerning Space Data System Standards (Green Book), CCSDS 350.1-G-2. Washington, D.C.: CCSDS, December 2015.

[D4] *CCSDS Guide for Secure System Interconnection*. Issue 1. Report Concerning Space Data System Standards (Green Book), CCSDS 350.4-G-1. Washington, D.C.: CCSDS, November 2007.

[D5] *CCSDS SANA Registry Management Policy*. Issue 1. CCSDS Record (Yellow Book), CCSDS 313.1-Y-1. Washington, D.C.: CCSDS, May 2016.