**Draft Recommendation for
Space Data System Standards**

MISSION OPERATIONS MONITOR & CONTROL SERVICES

DRAFT RECOMMENDED STANDARD

CCSDS 522.1-R-4

RED BOOK

April 2017

CCSDS

The Consultative Committee for Space Data Systems

Draft Recommendation for
Space Data System Standards

MISSION OPERATIONS MONITOR & CONTROL SERVICES

DRAFT RECOMMENDED STANDARD

CCSDS 522.1-R-4

RED BOOK

April 2017

AUTHORITY

Issue:	Red Book, Issue 4
Date:	April 2017
Location:	Not Applicable

(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF AUTHORITY:)

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

(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, IT WILL CONTAIN THE FOLLOWING 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:

o 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.

o 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.

o 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

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 the *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.

PREFACE

This document is a draft CCSDS Recommended Standard. Its 'Red Book' status indicates that the CCSDS believes the document to be technically mature and has released it for formal review by appropriate technical organizations. As such, its technical contents are not stable, and several iterations of it may occur in response to comments received during the review process.

Implementers are cautioned **not** to fabricate any final equipment in accordance with this document's technical content.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 522.0-R-1	Spacecraft Monitor and Control—Core Services, Draft Recommended Standard, Issue 1	September 2007	Superseded
CCSDS 522.1-R-2	Mission Operations Monitor & Control Services, Draft Recommended Standard, Issue 2	December 2012	Superseded
CCSDS 522.1-R-3	Mission Operations Monitor & Control Services, Draft Recommended Standard, Issue 3	March 2014	Superseded
CCSDS 522.1-R-4	Mission Operations Monitor & Control Services, Draft Recommended Standard, Issue 4	April 2017	Current draft

NOTE – Changes from issue 3 are too numerous to permit markup.

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

1 INTRODUCTION

1.1 GENERAL

This Recommended Standard defines the Mission Operations (MO) Monitor and Control (M&C) services in conformance with the service framework specified in reference [D1], *Mission Operations Services Concept*.

The M&C services are a set of services that enables a mission to perform basic monitoring and control of a local or remote mission operations entity. It provides basic monitoring and control services that are expected, but not required, to be used in conjunction with higher level mission operations services, such as Automation and Planning, as identified in reference [D1].

These services are defined in terms of the Common Object Model (COM) (see reference [3], *Mission Operations Common Object Model*), and the Message Abstraction Layer (MAL) (see reference [2], *Mission Operations Message Abstraction Layer*).

1.2 PURPOSE AND SCOPE

This Recommended Standard defines, in an abstract manner, the M&C services in terms of:

   a) the operations necessary to provide the services;

   b) the parameter data associated with each operation;

   c) the required behaviour of each operation;

   d) the use of the COM.

It does not specify:

   a) individual implementations or products;

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

   c) the methods or technologies required for communications.

1.3 APPLICABILITY

This specification is applicable to any mission operations component that provides a control interface or provides monitoring information to other mission operations components. Nominally, but not exclusively, this applies to mission operations onboard software across the space link, mission operations ground control systems to external mission operations entities, and between external mission operations entities.

1.4   RATIONALE

The primary goal of CCSDS is to increase the level of interoperability among agencies. This Recommended Standard furthers that goal by providing a standard service specification for the basic monitor and control of a remote entity. This supports multi-agency missions by providing a single specification for the exchange of basic monitor and control information, which, when combined with the other MO specifications of COM, MAL, and any supported protocol and message transport, provides an interoperable service for the basic monitor and control of a remote entity.

1.5   DOCUMENT STRUCTURE

This Recommended Standard is organised as follows:

  a) section 1 provides purpose and scope, applicability, and rationale of this Recommended Practice and lists the definitions, conventions, and references used throughout the document;

  b) section 2 presents an overview of the concepts;

  c) section 3 presents the M&C specification;

  d) section 4 is a formal specification of the M&C data structures;

  e) section 5 is a formal specification of the M&C error codes;

  f) section 6 specifies the internet location of the formal service specification eXtensible Markup Language (XML).

1.6   DEFINITIONS

action: A single executable task of a service provider, a telecommand is an example of an Action.

aggregation: A collection of parameters provided as a set by a service provider.

alert: Any operationally significant event.

argument: A single part of either an action or an alert.

COM object: A thing which is recognised as being capable of an independent existence and which can be uniquely identified. An object may be a physical object such as a spacecraft, an event such as an eclipse, or a concept such as telemetry parameter. It forms the fundamental part of a service specification, e.g., a parameter definition, a parameter value at a given point in time, a command. (See subsection 2.2 of reference [3].)

consumed service interface: The API presented to the consumer component that maps from the Service operations to one or more Service Data Units (SDUs) contained in MAL messages that are transported to the provided service interface.

group: A collection of COM objects of the same type.

hardware component: A complex physical entity (such as a spacecraft or a control system) or an individual physical entity of a system (such as an instrument, a computer, or a piece of communications equipment). A hardware component may be composed from other hardware components. Each hardware component may host one or more software components. Each hardware component has one or more ports where connections to other hardware components are made. Any given port on the hardware component may expose one or more service interfaces.

management service interface: A service interface that exposes management functions of a service function contained in a component for use by service consumers.

parameter: A single unit of data reported by a service provider.

provided service interface: A service interface that exposes the service function contained in a component for use by service consumers. It receives the MAL messages from a consumed service interface and maps them into Application Program Interface (API) calls on the provider component.

service capability set: A grouping of service operations. The specification of services is based on the expectation that different deployments require different levels of complexity and functionality from a service. To this end a given service may be implementable at one of several distinct levels, corresponding to the inclusion of one or more capability sets. The capability sets define a grouping of the service operations that remains sensible and coherent; it also provides a service provider with an ability to communicate to a consumer its capability.

service consumer (consumer): A component that consumes or uses a service provided by another component. A component may be a provider of some services and a consumer of others.

service data unit, SDU: A unit of data that is sent by a service interface and is transmitted, semantically unchanged, to a peer service interface.

service extension: Addition of capabilities to a base service. A service may extend the capabilities of another service with additional operations. An extended service is indistinguishable from the base service to consumers such that consumers of the base service can also be consumers of the extended service without modification.

service interface: A set of interactions provided by a component for participation with another component for some purpose, along with constraints on how they can occur. A service interface is an external interface of a service where the behaviour of the service provider component is exposed. Each service will have one defined 'provided service

interface' and may have one or more 'consumed service interface' and one 'management service interface'.

service provider (provider): A component that offers a service to another by means of one of its provided service interfaces. A component may be a provider of some services and a consumer of others.

service: A set of capabilities that a component provides to another component via an interface. A service is defined in terms of the set of operations that can be invoked and performed through the service interface. Service specifications define the capabilities, behaviour, and external interfaces, but do not define the implementation.

software component (component): A software unit supporting the business function. Components offer their function as services, which can either be used internally or which can be made available for use outside the component through provided service interfaces. Components may also depend on services provided by other components through consumed service interfaces.

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 CONVENTIONS

1.8.1 FIGURES

In figures illustrating this document, Unified Modelling Language (UML) modelling diagrams are used. Reference [1] provides further information regarding diagrams types and their meaning.

COM object relationship diagrams can be found in subsection 2.2.2 of reference [3].

1.8.2 TABLES

The table format information presented here is extracted from section 2 of reference [2]. It is repeated here to aid in understanding tables as they are presented in this document. A full description of the table formats presented in this document can be found in section 2 of reference [2] and in section 2 of reference [3].

Each interaction pattern definition contains a table that defines the template for operations that use that pattern.

Table 1-1: Example Operation Template

Operation Identifier	<<Operation name>>	
Interaction Pattern	<<Interaction pattern name>>	
Pattern Sequence	Message	Body Type
<<Message direction>>	<<Message name>>	<<Message type>>
…	…	…

The message direction denotes the direction of the message relative to the provider of the pattern and is either IN or OUT. So all messages directed towards the provider are IN messages, and all messages directed away from the provider are OUT messages.

Blue cells (dark grey when printed on a monochrome printer) contain table headings, light grey cells contain fields that are fixed for a pattern, and white cells contain values that must be provided by the operation or structure.

Where services and types are required from other MO specifications, the following notation is used to denote the area and, optionally, service in which the referenced definition resides:

<area>::<service>::<type name>

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 document 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.

[1]   *Mission Operations Reference Model*. Issue 1. Recommendation for Space Data System Practices (Magenta Book), CCSDS 520.1-M-1. Washington, D.C.: CCSDS, July 2010.

[2]   *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.

[3]   *Mission Operations Common Object Model*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 521.1-B-1. Washington, D.C.: CCSDS, February 2014.

NOTE   –   Informative references are listed in annex D.

2 OVERVIEW

2.1 GENERAL

The M&C services provide operations for generic monitoring and control of a remote service provider (e.g., a spacecraft or ground system component) in terms of three fundamental mechanisms and associated data items:

– action invocation—*actions*;

– parameter status—*monitoring parameters;*

– alert notification—*alerts*.

The following diagram presents the set of standards documentation in support of the Mission Operations Services Concept. The M&C services belong to the 'Service Specifications' documentation referenced in figure 2-1.
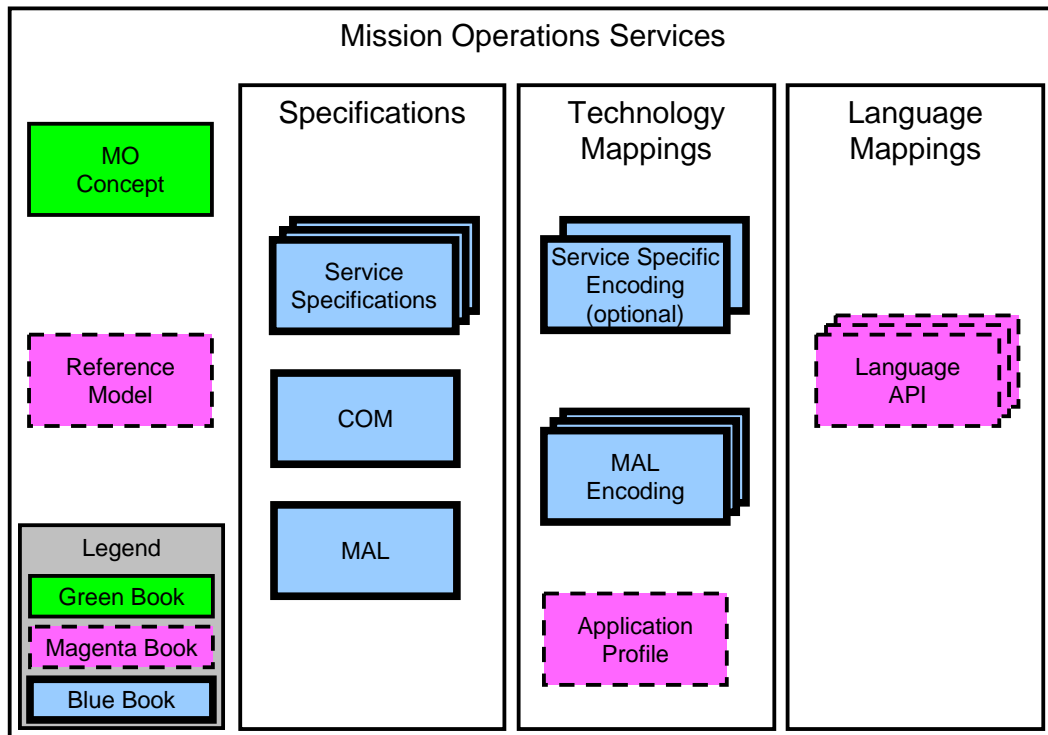
Figure 2-1: Mission Operations Services Concept Document Set

NOTE – References [D1] and [1] contain additional information about the MO Services Concept and the MO Reference Model, respectively.

The MO M&C services build upon the layering concept outlined in reference [D1] and are defined in terms of the MO MAL outlined in reference [2], so it is possible to deploy them over any supported protocol and message transport. The M&C services are also defined in

terms of the COM, as defined in reference [3], and builds upon the COM activity tracking service, the COM event service and the COM archive.

NOTE – The services defined in this specification are not dependent on any of the other services, other than the COM services, and are complete in their own right. An implementation is free to opt out of any service and also any capability set of these services; however, it may not make sense to support certain services without others.

2.2 MONITOR AND CONTROL CONCEPT

The M&C service provides the basic ability to monitor and control a remote entity. It provides three basic classes of information:

– **Actions** allow executable tasks to be invoked and their evolving status to be monitored: spacecraft telecommands are an example of an action.

– **Parameters** provide status monitoring capability but also may have their value set.

– **Alerts** provide a mechanism for asynchronous notification of operationally significant events or anomalies by the service provider to the service consumer.

In most modern systems the M&C services will be used in conjunction with higher level services such as automation and planning, which utilise the M&C services to fulfil their high level functions. For example, a planning service may take in requests to perform activities, these may get decomposed into automated procedures executed by the automation system, which in turn may monitor and control the remote entity using the M&C services:
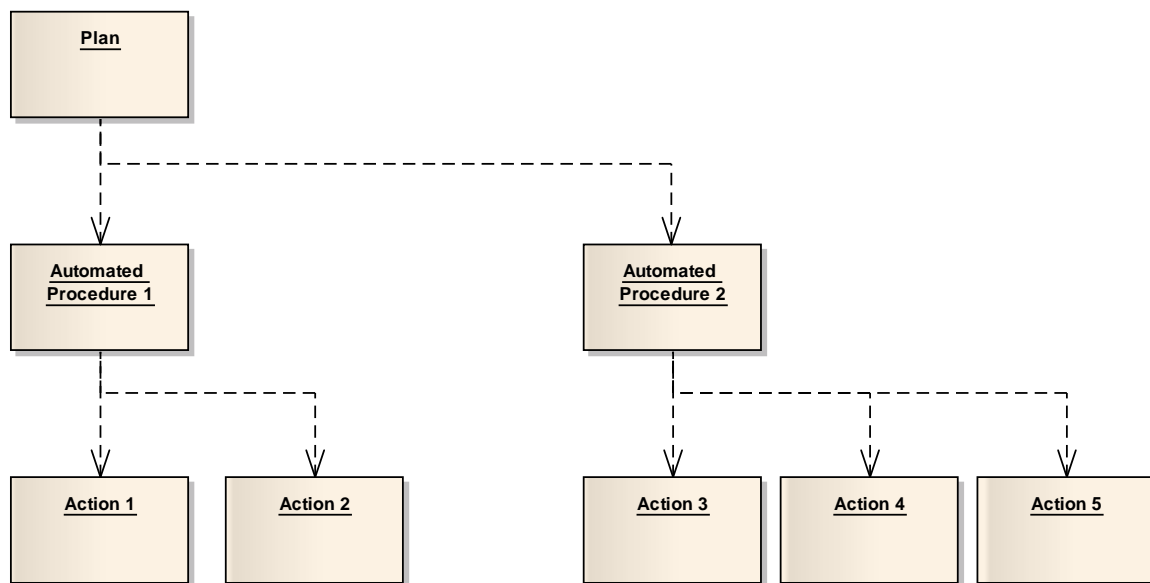
Figure 2–2: MO Service Layering Using Monitor and Control Services

Figure 2–2 shows an information view of the services and demonstrates how the services offered in one area may build upon the services from another area. However, the deployment of those services in a specific system may be different from one deployment to the next.

2.3 MONITOR AND CONTROL SYSTEM COMPOSITION

A system as a whole may be composed of many components that support some, all, or none of the M&C services. Any reasonably complex system is also likely to include other services outside of the M&C area, such as planning and scheduling services, or even mission-specific services. It is also likely that more than one component is a provider of M&C services, so not just the end spacecraft, or some component of it, but also ground components such as the Mission Control System (MCS) in the case of mission operations (or a Central Checkout System [CCS] in the case of assembly Integration and Test) may be controllable via the M&C services. Figure 2–3 shows relationships between service consumers and providers in an M&C system and also illustrates how services can be run over different communication protocols appropriate to the link.
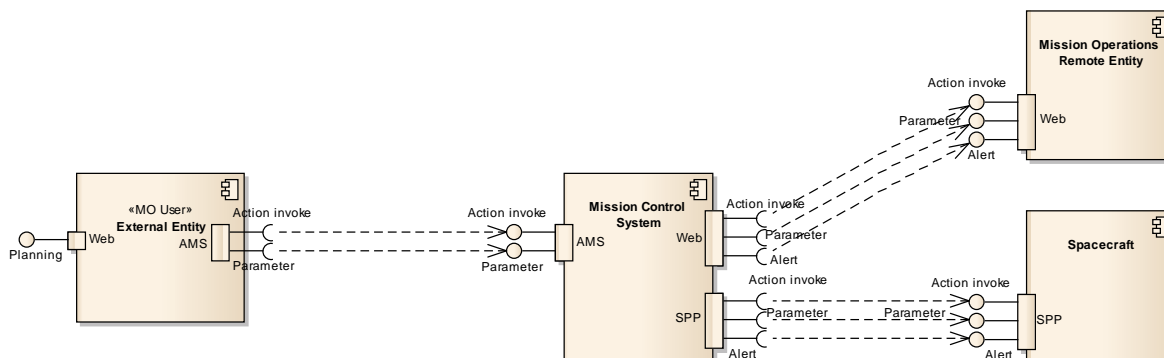
Figure 2–3: Example Service Consumers and Providers

Some components may augment the capabilities of other components by providing extra capabilities on top of those provided by the original component. An example of this is a ground system augmenting the basic parameter information obtained from a spacecraft with validity, check, and statistic information. Figure 2–4 shows a service data augmentation where the MCS augments the raw parameter values from the spacecraft with calibrated/converted values.
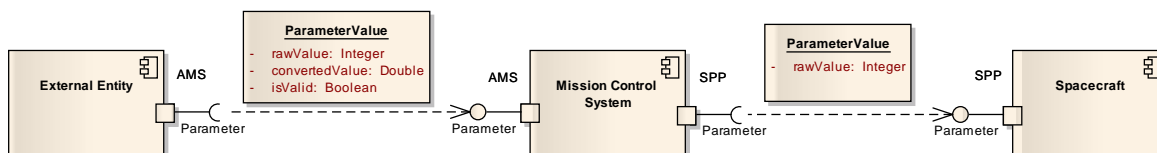
Figure 2–4: Service Data Augmentation

The system may also export M&C services outwards to other systems and components, providing extended capabilities over and above those of the controlled system, for example, providing a pre-transmission check of any actions before sending them on to a spacecraft or ground system. Figure 2-5 shows extension of services.
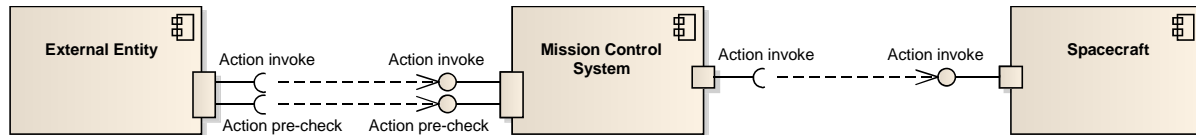
Figure 2-5: Service Extension

The services defined in this document provide the basis for M&C of a component, local or remote. The following subsections describe the basic uses of each of these services.

2.4 ACCESS CONTROL

For many situations there may be a deployment or mission specific policy for limiting either access to specific received data or the ability to execute specific operations. This concept of authentication and authorisation is provided by the MAL access control concept and is covered in subsections 3.6, 5.2, and 5.3 of the Reference Model (reference [1]).

Another area that access control can be used for is situations that are context specific rather than operator specific. For example, a normal access control situation might be to restrict the ability to execute specific operations to specific operators. However the MAL access control concept can be used to restrict certain operations from being executed depending on the state of the service provider. For example, if a service defines service objects that relate to each other and then also a delete operation for its service objects, access control can be used to ensure any attempt to delete a service object that is still referenced by another fails.

This use of the MAL access control concept to provide 'protection' of data allows deployment specific policies to be implemented without forcing operational restrictions on all through specification.

2.5 M&C DATA MODEL

All of the M&C services use a similar data model for their service specific objects. These follow a pattern of an Identity, a Definition, and optionally some form of evolving value.

The Identity object represents the unchanging aspect, the identity, of the service object; for example, in a Parameter it is the parameter name. It forms the single point in the data model that will not change for a service object.

The Definition object represents the details of the service object. It links to an Identity, so forms the definition of that Identity, and there may be several versions of that definition over time. However, there can only be one definition for an Identity in effect at any one time. The COM archive may hold all of the definitions to show what has changed over time, but in a

specific provider there must only be a single Definition in use for a specific Identity at any point in time.

The Value object represents a specific 'value' of the service object at a point in time. It links to a Definition object to indicate which definition was in effect at the time of the Value creation, and there may be several versions of that value over time. However, there can only be one value for an Identity in effect at any one time. The COM archive may hold all of the values to show what has changed over time, but in a specific provider there must only be a single Value for a specific Identity at any point in time.

2.6   ACTION SERVICE

2.6.1   GENERAL

Actions enable consumers of the service to control the remote system, typically a spacecraft; however, there is no restriction on what mission operations entity the remote system may be. Action invocation operations include issuing of action directives by an authorised client to the remote service provider, and the subsequent monitoring of the evolving execution status of that action by both the initiator and other client functions. It should be noted that the action service is concerned with the execution of actions; the control of when that execution happens is a function of the scheduling service outlined in reference [D1].

2.6.2   INTERLEAVING OF ACTIONS FROM SEPARATE CONSUMERS

There are some situations where having multiple sources of actions being submitted, in parallel, to an action provider can cause issues. In these situations it is a requirement that actions from a single source not be interleaved with those of another; this is sometimes referred to 'blocking' or 'grouping'. The action service as it is currently defined does not support these concepts.

One possible solution to this would be to modify the action service to take a list of actions to be executed sequentially without interleaving; however, there are many situations where the sequence of actions requires more complex internal relationships than just sequential execution, such as delays and conditions, which would need to be captured in the submission to the action service provider. At this point the responsibility of the action service would be migrating into automation and schedule execution and therefore this is not suitable.

Another possible solution to this would be to have operations that allow a consumer to 'lock' the action service provider to only its own actions until the consumer 'unlocks' it at a later time. However this solution is extremely error prone if, for example, the locking consumer terminates before unlocking the provider and liable to lead to deadlock situations, and therefore it also is not suitable.

Therefore, because of these limitations on the solutions, the action service does not support the 'blocking' concept itself and it is delegated to a higher service such as schedule execution. It should be noted that the implementation of these higher services may require

deployment specific solutions to support 'blocking' of actions but that is an issue for the higher services.

2.6.3 ACTION PRE-TRANSMISSION CHECKING

When dealing with a remote service provider such as a spacecraft, it is normal to perform some sort of pre-check before transmitting an action. This includes checks such as link status, spacecraft state, and action argument checks. These pre-checks would most often be performed by a proxy component that is present in the ground segment.

It is possible that the system being deployed requires no link checks if it is fully onboard or ground-based. In this case, the pre-checks would either be nonexistent or limited to spacecraft state and action argument checks.

The action service provides a mechanism for the checking of an action to ensure it is okay to send. The actual content and detail of the checks are implementation- and context-specific; the action service only provides the mechanism, not the specifics of the checking. It is also possible to not use this mechanism and just attempt to send the action; in this case, the status of the action should be monitored to determine its success.

The submission of actions therefore has two main patterns of behaviour:

a) Check and then send: An action is submitted to a local component to determine if it would be executed if subsequently sent. This allows a client application to provide feedback to an operator about the likely success of an action without requiring it to be sent to the remote provider.

b) Just send: An action is just sent for execution and the returned execution status is checked to see if the action succeeds. This pattern is likely to be used by automated clients.

2.6.4 ACTION PROVIDER ADDRESSING

Where more than one component is involved in the transmission of an action from a consumer to the provider, or where there are multiple action service providers, ensuring that the action is sent/relayed to the correct service provider is critical.

A service provider is located using its MAL URI address, where this is directly addressable by a consumer there are no issues. However, where there may be more than one component involved in the relay of the action the addressing scheme becomes central to the success of the transmission from consumer to provider.

There are two possible types of addressing scheme, absolute or relative. In absolute addressing the URI used uniquely addresses one and one only service provider. It is the responsibility of the underlying network to transfer the action request from the consumer to the provider. Email is an example of absolute addressing.

In relative addressing the address used for the transmission between two intermediate components is not necessarily the same address as used by either the next stage or the final provider. In this case, how the action request is routed from one component to the next must be defined using some out-of-band agreement. TCP/IP address translation is an example of relative addressing.

2.6.5   ACTION PRE-EXECUTION CHECKING

Before the action is executed in the remote provider it most probably has some more, local, checks performed on it. These pre-execution checks allow a provider to ensure that the received action is still valid given the possibly lengthy delay between initial transmission and execution. The MO action service allows an action service provider to perform these checks, as appropriate for that implementation, and report the result in a COM activity tracking Acceptance event (see subsection 2.6.2 of reference [3]).

2.6.6   ACTION AUTHENTICATION

Authentication of received Actions is an essential step of pre-execution checking. It is vital to the security of missions that authentication is covered in the action concept.

For the action service, authentication information is provided by the MAL layer in the header of the submitAction operation. How this is used is a deployment specific decision; however, the concept of authentication is covered in subsections 3.6, 5.2, and 5.3 of the Reference Model (reference [1]).

2.6.7   ACTION VERIFICATION

The action service reports to consumers the progress of actions. This verification information provides feedback to consumers about both the transmission progress (from consumer to provider) and execution progress (inside the provider). It should be noted that the action service does not model the predicted execution state of an action; it only reports on progress when each stage is reached.

The reporting of action transmission and execution verification is provided by the COM activity tracking service. The action service uses the standard activity service transport reporting and then extends the activity execution model for the reporting of action execution information.

The standard COM activity subscription process allows a client to specify the filter for the status information returned.

2.6.8 ACTION FORWARDING

Action forwarding is when one component acts as a relay for another and forwards actions on to the final component. The action service uses the COM activity multi-hop support to transport actions from a consumer to a provider via an intermediate node as required.

2.6.9 ACTION CHAINING

When the transmission and execution of one action or service operation is linked to another, for example, the execution of one Action causes the execution of another Action, the COM object model allows the implicit relationship between the two activities to be shown. This does not mean that Actions can be linked together before execution (i.e., the building of a sequence of Actions, as that is part of another service outside the M&C area), but it does allow the traceability of the results of an Action to be shown (i.e., that Action 'B' was triggered by the execution of Action 'A'). The link between the two objects shows the relationship between these two separate activities and allows the full chain of actions and operations to be traceable.

2.7 PARAMETER SERVICE

Parameter status monitoring is performed by the reception of updates to a set of predefined monitoring parameters that are published by a monitoring parameter service provider.

Monitoring parameters have an evolving status represented by a chronological sequence of status updates over an unbounded lifetime. Status updates may be periodic, change-based, or a mixture of the two.

The M&C parameter service does not define the provided parameter information in the service specification; it delegates the definition of the provided parameter information to the runtime configuration of the provider. This does not mean that a component that provides the M&C parameter service has to support dynamic definition of parameters (that may be impossible if the component is a temperature sensor), but that the service specification of the M&C parameter service does not fix what parameters are to be returned for a component, only what the structures look like, and how the parameter information is returned.

Monitoring parameters are basic types such as strings or integers. Composition of parameters is supported by defining an appropriate aggregation (see Aggregation service in 2.11).

In most cases it is expected that the service consumer will register with the service provider for one or more parameters for a period of time and will perform this operation as a set of the following well-defined stages:

    a) subscribe for parameters;

    b) receive updates;

c) (optional) modify requested parameter list;

d) unsubscribe.

It is also possible, when supported, for a consumer of the service to set the value of parameters.

Monitoring parameters will acquire new values by one of the following three methods:

a) Physical changes in the properties of a real-world device. This is the case for a spacecraft parameter whose values directly indicate a physical state of the spacecraft.

b) Calculations arising from changes to the value of other monitoring parameters. This is the case for a 'derived' or 'synthetic' parameter that takes the values of other monitoring parameters as inputs and applies a function to produce its value.

c) The value being supplied by a consumer of the service to the provider using the setValue operation.

Parameter definitions are managed using the operations of the Parameter service and may also be managed via the common configuration service if the configuration service is supported by a deployment.

2.8   ALERT SERVICE

Alerts are raised asynchronously to report a significant event or anomaly. Alerts may originate within the remote service provider (spacecraft or other controlled system) itself or within an associated ground-based component in response to a transition in some monitored status.

Alerts are characterised by an identifier and a set of arguments. However, it is possible that some systems require text-only alert messages where the body of the alert is a free-form text message. For these systems it is possible to define a single alert definition that contains a single string argument; however, it should be noted that translation software shall be required when moving argument-based messages and string-based messages between the two formats.

The alert service uses the COM event capability to publish its alerts. An alert is a specialised COM event; it allows the definition of the alert to be defined dynamically, whereas COM events are defined statically in the relevant service specification. The alert service defines operations to define its alerts.

2.9   CHECK SERVICE

The check service allows the consumer to define a set of checks to be applied to parameters and then have the check provider periodically sample the values of those parameters and check them. If a check is violated (for example, goes outside of the specified limits), the consumer is notified by the generation of a COM event.

A minimal list of check types is supported; however, service implementations may support additional custom types. The following is the list of checks that must be supported:

– *limit check*—the value lies within a specified range;

– *constant check*—the value is checked against a specified value or value of another parameter;

– *delta check*—the change in value is checked against a pair of thresholds.

The check service defines operations to specify the checks and associate parameters to those checks. The association also contains a check condition that must evaluate to TRUE before the check is applied. Any check state transitions are reported as a COM event.

2.10 STATISTICS SERVICE

The statistics service allows the consumer to associate parameters to defined statistical evaluations (e.g., *min*, *max*, *mean*, *standard deviation*) and have the service provider periodically sample the values of those parameters and evaluate them. The resultant statistics evaluations are provided to consumers who register interest in them.

The service allows parameters to be associated with a statistic function and have those parameters evaluated by the function; however, it does not allow new statistic algorithms not currently supported by a service provider to be defined. For example, if a service provider supports *min* and *max* statistics, it is possible to request parameters to be evaluated by these statistical functions, but it is not possible to add a new statistic function 'standard deviation', as this is not supported by the service provider.

NOTE – The actual statistical functions supported by a statistics service provider are an implementation detail. It should also be noted that service providers may not provide this service.

2.11 AGGREGATION SERVICE

2.11.1 GENERAL

A logical extension to basic parameter monitoring is data aggregation. The aggregation service provides the capability to acquire several parameter values in a single request. The aggregation might be one of the following:

– predefined by the service provider, e.g., housekeeping parameters;

– predefined at runtime by the consumer, e.g., a diagnostic report.

For example, the user may request a data product that comprises the accelerations and angular rates of the spacecraft. This would be acquired by reading the appropriate gyros and accelerometers onboard and returning the data as a set. However, if the service provider is

actually on the ground, for example an MCS kernel acting as a proxy, then it may collate the set from its current 'state vector' of all reported parameters.

The aggregation service provides operations to define which parameters to aggregate and reports the current values of those parameters.

2.11.2 AUTOMATIC PUBLISH OF AGGREGATIONS

In many situations it is expected that basic summary or overview information about the state of a system or spacecraft be published without an initial consumer request. For example, for safety reasons it is normal for a spacecraft to broadcast basic status information when in recovery/safe mode to help diagnose any communication issues that might exist.

Whilst no operations for this situation have been explicitly defined it should be noted that it is completely valid for a service provider to publish these aggregation updates without having received a subscription request from a consumer. In this situation the registration request from the consumer is hardcoded into the provider application, the MO publish/subscribe concept is not modified but in effect parts of it have been hardcoded into the deployment.

2.12 CONVERSION SERVICE

A functional extension of the other services is to add the engineering unit conversion capability. The conversion service defines a basic set of conversion patterns and allows instances of these conversion to be defined using the COM archive. Other services then can reference these conversions as appropriate to those services.

For example, using this conversion service with the parameter service, firstly an appropriate conversion has to be defined in the conversion service and then associated to a parameter. When consumers register for updates for that parameter, they will receive both a raw and a converted value such as a quantised temperature value with a unit.

The conversion service does not provide any operations but specifies the structures used to define the conversions, how they use the COM structures, and how it should be used with the COM archive. Which conversions are applied to which entities of a particular service is a detail of the definition of that service, not the conversion service, the conversion service only defines the conversion rules.

NOTE – The actual conversion functions supported by a service provider are an implementation detail; however, this standard defines a minimum set of these that must be supported for compliance.

2.13 GROUP SERVICE

To simplify the administration of the services, and to reduce the bandwidth required for certain operations, a group concept is provided. The group service defines the concept of a group; a group is a simple object which holds a list of links to other objects. The objects a group links to must all be of the same type and in the same domain. Services then reference these groups in their operations rather than their own objects.

It is also possible to define a group of groups, where one group holds links to other groups which in turn either contain groups or other objects. All operations that support groups also support this concept of groups of groups; however, the objects that they finally link to must still all be the same type and be supported by the specific operation. For example, if an operation states that it can reference objects of type X or groups of X then it must also support groups of groups of X. The depth of the grouping is not limited by this specification.

Instances of these groups are defined and maintained using the COM archive. Other services then can reference these groups as appropriate to those services.

For example, using this group service with the parameter service, firstly an appropriate group has to be defined in the group service. When a consumer wants to disable generation of updates for all the parameters in the group a single operation with the single group identifier is all that is required rather than a large list of parameter identifiers.

The group service does not provide any operations but specifies the structures used to define the groups, how they use the COM structures, and how they should be used with the COM archive. How groups are used by a particular service is a detail of the definition of that service, not the group service, the group service only defines the concept of groups.

3   SPECIFICATION: MC

3.1   OVERVIEW

This section details the M&C services; the structures used by the services are detailed in section 4. Both the services and structures are defined in terms of the MO MAL, which is defined in reference [2], so it is possible to deploy them over any supported protocol and message transport.

The services defined here are also specified in terms of the COM, which is defined in reference [3].

To aid comprehension, several tables are included for each service and operation definition. The table formats are briefly described in 1.8.2 and are the same as those used for the specification of the MO COM services in reference [3].

All service specifications in this document are part of the M&C area, which has a short form number of '4'.

3.2   SERVICE: ACTION

3.2.1   OVERVIEW

The action service allows consumers to submit an action for execution and to subsequently monitor the execution progress of these actions via the COM activity tracking pattern. The progress of the action is split into two parts, firstly transfer from the consumer to the provider, and secondly execution in the provider.

An action is submitted to the provider using the submitAction operation, the progress of which can be monitored using the COM activity tracking pattern, which completes when the action has been delivered to the provider. The action is then executed in the provider which can also be monitored using the activity tracking pattern. The submitAction operation takes the object instance identifier of the submitted action and uses that to populate the source fields of the activity tracking events. Coordination may be required between action service consumers to ensure the action object instance identifiers are unique. How this is done is outside the scope of this specification; however, a good approach is the use of a central COM archive as that can provide the unique instance identifiers.

The nominal sequence of action submission and execution monitoring are shown in figure 3-1.
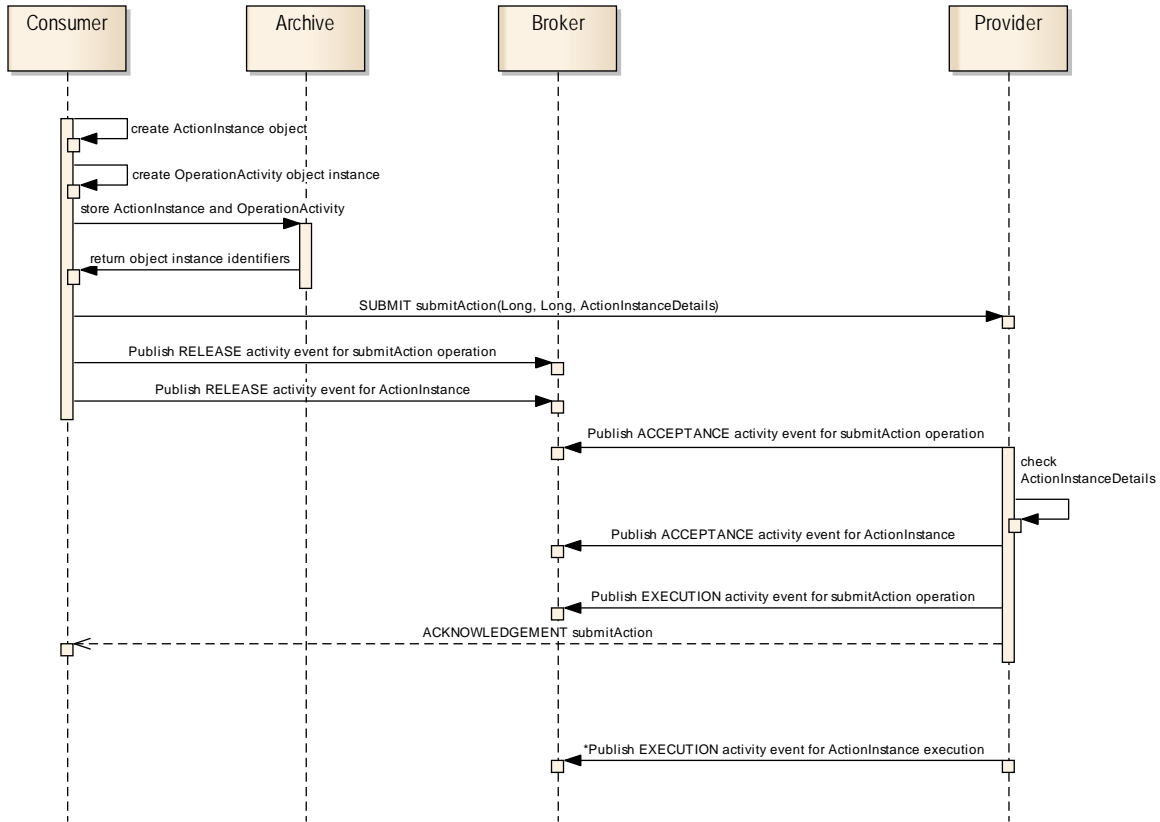
Figure 3-1: Nominal Sequence of Action Submission and Monitoring

The sequence above shows the activity tracking events being published to a MAL broker (see reference [2]). The sequence does not show the broker passing the events to any subscribed consumers, as that is outside the intent of the figure.

The consumer is responsible for creating and archiving the action instance object and then using the submitAction operation to submit it to the action provider for execution. The action provider reports, using the COM ActivityTracking service, both the execution progress of the submitAction operation and also the execution of the action itself. Once the supplied action instance details have been checked and execution of the action started by the submitAction operation, that operation finishes, whilst the execution of the action possibly continues (depends on the actual action). The final interaction in the sequence shows the execution events of the executing action instance, the '*' at the start of that line indicates zero to many events being published.

If the execution of an action fails with an error, the action service provider can publish an ActionFailure COM event to hold the error code being reported by the failure. If no error code is required to be reported then there is no need to publish the event as the normal COM ActivityTracking event contains a success indication.

The service also includes an operation, preCheckAction, which checks that an action would be accepted for execution without actually submitting it for execution. It is expected to be provided by local action proxies, rather than the remote system, to allow for localised checking of things such as link state, argument values, action safety, before sending the action over long and slow space links.

The action service defines three types of objects. The first type is the ActionIdentity object that holds the name of an action. The second type is the ActionDefinition object that holds the description of an action with the list of required/optional arguments. The third type is the ActionInstance object that holds details of a specific action instance namely a value for each of the arguments of the action.

Table 3-1: Action Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MC	Action	4	1	1
Interaction Pattern	**Operation Identifier**	**Operation Number**	**Support in Replay**	**Capability Set**
SUBMIT	submitAction	1	No	1
REQUEST	preCheckAction	2	No	
REQUEST	listDefinition	3	Yes	2
REQUEST	addAction	4	No	3
REQUEST	updateDefinition	5	No	
SUBMIT	removeAction	6	No	

3.2.2   HIGH-LEVEL REQUIREMENTS

3.2.2.1   The action service shall provide:

   a) the capability for submitting actions for execution;

   b) the capability for submitting actions for check without execution;

   c) the capability for listing the numeric identifiers for the action definitions;

   d) the capability for defining a new definition;

   e) the capability for updating an existing definition;

   f) the capability for removing an existing definition.

3.2.2.2   The list of actions that are supported by the action service shall be declared when deploying that service.

3.2.2.3   Each action submission shall include:

a)   the action definition numeric identifier;

b)   the action instance numeric identifier;

c)   the ordered list of action arguments.

3.2.3   FUNCTIONAL REQUIREMENTS

The submitAction operation shall be used to submit a new action instance for execution.

3.2.4   COM USAGE

3.2.4.1   Each action of a provider shall be represented by an ActionIdentity COM object.

3.2.4.2   The body of the ActionIdentity COM object shall hold the name of the action.

3.2.4.3   The definitions of the actions shall be represented as ActionDefinition COM objects.

3.2.4.4   The ActionDefinition object shall use the related link to indicate which ActionIdentity object it uses.

3.2.4.5   The source link of the ActionIdentity object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.2.4.6   The source link of the ActionDefinition object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.2.4.7   Instances of an action that are submitted to a provider using the submitAction operation shall be represented as ActionInstance COM object.

3.2.4.8   The object instance identifier for an ActionInstance object shall be populated by the consumer.

3.2.4.9   The ActionInstance object shall use the related link to indicate which ActionDefinition object it uses.

3.2.4.10   Instances of an action that are submitted to a provider using the submitAction operation shall link to the submitAction OperationActivity object using the source link.

3.2.4.11   The source link of the submitAction OperationActivity object should be the object that caused it to be created, most likely an operator login or automated procedure object.

Table 3-2: Action Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
ActionIdentity	1	MAL::Identifier	Set to NULL	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
ActionDefinition	2	ActionDefinitionDetails	1	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
ActionInstance	3	ActionInstanceDetails	2	The object that caused it to be created, most likely a COM OperationActivity object if created by the submitAction operation.

3.2.5   COM EVENT SERVICE USAGE

3.2.5.1   Execution progress of an action shall be reported using COM ActivityTracking Execution events.

3.2.5.2   The Execution event may be reported many times as it is used to report each stage in the execution of the action. It is implementation specific when a progress step is reached and the Execution event will be reported.

3.2.5.3   If the execution of an action fails then that failure shall be reported using the normal COM ActivityTracking event pattern, specifically an ActivityTracking Execution event with the success field set to FALSE.

3.2.5.4   If the execution of an action fails with an error code then that error code shall be reported using an ActionFailure COM event in addition to the ActivityTracking event used to report the action failure.

3.2.5.5   The source of the ActionFailure event shall be the ActivityTracking Execution event that represents the failure stage of the action execution.

3.2.5.6   The related field of the ActionFailure event shall link to the ActionInstance object that failed.

3.2.5.7   The body of the event shall hold a numeric failure code that is either a MAL error number or a deployment specific code.

3.2.5.8   The specification of the deployment specific failure code is outside the scope of this specification and is expected to be agreed out of band.

Table 3-3: Action Service Events

Event Name	Object Number	Object Body Type	Related points to	Source points to
ActionFailure	6	MAL::UInteger	3	COM::ActivityTracking::5

3.2.6   DISCUSSION—COM OBJECT RELATIONSHIPS

Figure 3-2 below shows the COM object and event relationships for this service:
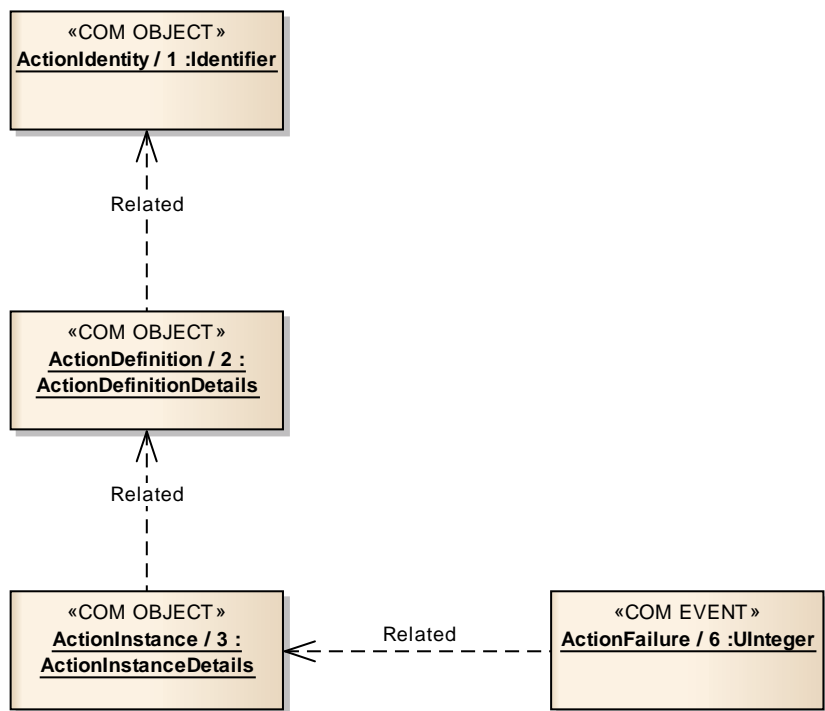
Figure 3-2: Action Service COM Object and Event Relationships

3.2.7   COM ARCHIVE SERVICE USAGE

3.2.7.1   ActionIdentity objects should be stored in the COM archive.

3.2.7.2   ActionDefinition objects should be stored in the COM archive.

3.2.7.3   Before an action is sent with the submitAction operation, the ActionInstance object should be stored in the COM archive by the consumer.

3.2.7.4   The COM activity tracking events associated with the submission and execution of the action should be stored in the COM archive.

3.2.7.5   Any ActionFailure COM event objects should be stored in the COM archive.

3.2.8   COM ACTIVITY SERVICE USAGE

3.2.8.1   The COM Activity service should be used to monitor the transfer and execution of the submitAction operation.

3.2.8.2   The COM Activity service should be used to monitor the transfer and execution of ActionInstance objects.

3.2.8.3   The COM activity events for the submitAction operation shall be generated as defined in the COM activity tracking service.

3.2.8.4   A Release activity event for the action instance shall be generated when an ActionInstance object is released from a consumer using the submitAction operation.

3.2.8.5   The Acceptance activity event for the ActionInstance object, separate from the Acceptance activity event generated for the submitAction operation, shall be generated when an action is received by the destination provider and after any pre-execution checks have been performed.

3.2.8.6   The body of the ActionInstance ActivityAcceptance event object shall be populated with the pre-execution checks result.

3.2.8.7   The source link of the activity events for the ActionInstance shall be the ActionInstance object.

3.2.8.8   The related link of the activity events for the ActionInstance shall be set to NULL.

3.2.8.9   In all activity execution events for the action instance the ActivityExecution.stageCount field shall be set to 2 plus the total number of progress stages.

3.2.8.10   In the activity execution event, when notification of an action starting is requested, the ActivityExecution.executionStage field shall be set to 1.

3.2.8.11   In the activity execution event, when notification of an action progress is requested, the ActivityExecution.executionStage field shall be set to 1 plus the current progress stage.

3.2.8.12   In the activity execution event, when notification of an action completion is requested, the ActivityExecution.executionStage field shall be set to 2 plus the total number of progress stages.

3.2.9   OPERATION: SUBMITACTION

3.2.9.1   Overview

The submitAction operation allows a consumer to submit an action to a provider for remote execution.

Operation Identifier	submitAction	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	actionInstId : (MAL::Long) actionDetails : (ActionInstanceDetails)

3.2.9.2   Structures

3.2.9.2.1   The actionInstId field of the submission shall contain the object instance identifier of the ActionInstance to be used for activity tracking events.

3.2.9.2.2   The actionDetails part of the submission shall contain the argument values and related information of the action instance to be executed.

3.2.9.2.3   If the defInstId of the supplied actionDetails field does not match a known ActionDefinition object then an UNKNOWN error shall be returned.

3.2.9.2.4   The size of the argumentValues list of the ActionInstanceDetails structure shall be compared to the size of the argument list in the matched ActionDefinition object and an INVALID error shall be returned if they are not the same.

3.2.9.2.5   If the ActionInstanceDetails structure contains an argumentIds field value then this shall be compared to the same field in the matched ActionDefinition object and must be the same size and contain the same values, an INVALID error shall be returned if this is not the case.

3.2.9.2.6   If the ActionInstanceDetails structure contains an isRawValue field value then the size of this list shall be compared to the size of the argument list in the matched ActionDefinition object and an INVALID error shall be returned if they are not the same.

3.2.9.2.7   If the supplied argument values do not match the attribute type specified in the action definition then an INVALID error shall be returned.

3.2.9.2.8   A service provider may apply some deployment specific checks to the action instance and can return an INVALID error if they fail.

3.2.9.2.9   If an error is raised then no action shall be executed.

3.2.9.2.10   The SUBMIT acknowledgement shall be returned once the action has been accepted for execution but before execution starts.

3.2.9.3   Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

    1) the list sizes held in the ActionInstanceDetails do not match the argument definitions or it contains one or more invalid argument values;

    2) if the two lists are not the same length then the extra information field shall contain the first index of the element in the largest list which does not have corresponding element in the other list;

    3) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: UNKNOWN: submitted action definition is unknown.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	Not Used

3.2.10 OPERATION: PRECHECKACTION

3.2.10.1 Overview

The preCheckAction operation allows a consumer to check that an action would be successfully accepted for execution without actually submitting the action. The operation is expected to be provided by local action proxies rather than the remote system to allow for quick local checks before sending the action over long and slow space links.

Operation Identifier	preCheckAction	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	actionDetails : (ActionInstanceDetails)
OUT	RESPONSE	accepted : (MAL::Boolean)

3.2.10.2 Structures

3.2.10.2.1 The actionDetails part of the submission shall contain the argument values and related information of the action instance to be executed.

3.2.10.2.2 If the ActionInstanceDetails structure contains an argumentIds field value then this shall be compared to the same field in the matched ActionDefinition object and must be the same size and contain the same values, an INVALID error shall be returned if this is not the case.

3.2.10.2.3 If the supplied argument values do not match the attribute type specified in the action definition then an INVALID error shall be returned.

3.2.10.2.4 A service provider may apply some deployment specific checks to the action instance and can return an INVALID error if they fail.

3.2.10.2.5 The returned Boolean shall be set to TRUE if the action would be accepted successfully; otherwise the operation shall return FALSE.

3.2.10.3 Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

  1) the argument list contains one or more invalid arguments;

  2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: UNKNOWN: submitted action definition is unknown.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	Not Used

3.2.11 OPERATION: LISTDEFINITION

3.2.11.1 Overview

The listDefinition operation allows a consumer to request the object instance identifiers of the latest ActionIdentity and ActionDefinition objects for the supported actions of the provider.

Operation Identifier	listDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	actionNames : (List<MAL::Identifier>)
OUT	RESPONSE	actionInstIds : (List<ObjectInstancePair>)

3.2.11.2 Structures

3.2.11.2.1 The actionNames field shall contain a list of action names to retrieve the ActionIdentity and ActionDefinition object instance identifiers for.

3.2.11.2.2 The request may contain the wildcard value of '*' to return all supported ActionIdentity and ActionDefinition objects.

3.2.11.2.3 The wildcard value should be checked for first, if found no other checks of supplied identifiers shall be made.

3.2.11.2.4 If a provided identifier does not include a wildcard and does not match an existing ActionIdentity object then this operation shall fail with an UNKNOWN error.

3.2.11.2.5 The response shall contain a list of matching ActionIdentity and ActionDefinition object instance identifiers.

3.2.11.2.6 The returned list shall maintain the same order as the submitted list unless the wildcard value was included in the request.

3.2.11.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

a) one of the supplied identifiers is unknown;

b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.2.12 OPERATION: ADDACTION

3.2.12.1 Overview

The addAction operation allows a consumer to define one or more actions that do not currently exist. The new ActionIdentity and ActionDefinition objects are expected to be stored in the COM archive by the provider of the action service.

Operation Identifier	addAction	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	actionDefDetails : (List<ActionCreationRequest>)
OUT	RESPONSE	newObjInstIds : (List<ObjectInstancePair>)

3.2.12.2 Structures

3.2.12.2.1 The actionDefDetails field shall hold the name and definitions to be added.

3.2.12.2.2 The name field must not be the wildcard '*', or empty (an INVALID error shall be returned in this case).

3.2.12.2.3 The supplied name must be unique among all ActionIdentity objects for the domain of the provider; otherwise a DUPLICATE error shall be raised.

3.2.12.2.4 If an error is raised then no new identities and definitions shall be added as a result of this operation call.

3.2.12.2.5 If the supplied name matches an existing, but removed, ActionIdentity then that ActionIdentity shall be reused; otherwise a new ActionIdentity shall be created.

3.2.12.2.6 The provider shall create a new ActionDefinition object and store it, and any new ActionIdentity objects, in the COM archive.

3.2.12.2.7 The response shall contain the list of object instance identifiers for the ActionIdentity and new ActionDefinition objects.

3.2.12.2.8 The returned list shall maintain the same order as the submitted definitions.

3.2.12.3 Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

   1) one of the supplied ActionIdentity objects contains an invalid action name;

2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: DUPLICATE:

1) one or more of the ActionIdentity objects being added has supplied an action name that is already in use in the domain;

2) the extra information field contains a list of the indexes of the erroneous values from the originating request list.

Error	Error #	ExtraInfo Type
DUPLICATE	Defined in COM	List<MAL::UInteger>

3.2.13 OPERATION: UPDATEDEFINITION

3.2.13.1 Overview

The updateDefinition operation allows a consumer to update a definition for one or more actions.

This differs from deleting an existing action and adding a new definition with the same name in the fact that the ActionIdentity object is not changed between the two definitions.

The replacement definition is expected to be stored in the COM archive by the service provider. The operation does not remove the previous ActionDefinition object from the COM archive, merely removes the object from the provider. This permits existing, and completed, ActionInstance objects to continue to reference the correct ActionIdentity and ActionDefinition objects in the COM archive.

Operation Identifier	updateDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	actionObjInstIds : (List<MAL::Long>) actionDefDetails : (List<ActionDefinitionDetails>)
OUT	RESPONSE	newDefInstIds : (List<MAL::Long>)

3.2.13.2 Structures

3.2.13.2.1 The actionObjInstIds field shall contain the list of object instance identifiers of the ActionIdentity objects to be updated.

3.2.13.2.2 The supplied object instance identifiers shall match existing identity objects, an UNKNOWN error shall be raised if this is not the case.

3.2.13.2.3 If the actionObjInstIds list contains either NULL or '0' an INVALID error shall be raised.

3.2.13.2.4 The actionDefDetails field shall contain the replacement ActionDefinitionDetails.

3.2.13.2.5 The two lists shall be ordered the same.

3.2.13.2.6 The number of entries in the two lists shall be the same size; otherwise an INVALID error shall be returned.

3.2.13.2.7 If an error is raised then no definitions shall be modified as a result of this operation call.

3.2.13.2.8 The provider shall create a new ActionDefinition object and store it in the COM archive.

3.2.13.2.9 The new ActionDefinition object shall be the current ActionDefinition used for the specific ActionIdentity.

3.2.13.2.10 The response shall contain the list of object instance identifiers for the new ActionDefinition objects.

3.2.13.2.11 The returned list shall maintain the same order as the submitted definitions.

3.2.13.3 Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

1) the supplied object instance identifiers list contains either a NULL or '0' or the two supplied lists are not the same length;

2) if the two lists are not the same length then the extra information field shall contain the first index of the element in the largest list which does not have corresponding element in the other list;

3) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: UNKNOWN:

1) one of the supplied ActionIdentity object instance identifiers is unknown;

2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.2.14 OPERATION: REMOVEACTION

3.2.14.1 Overview

The removeAction operation allows a consumer to remove one or more actions from the list of actions supported by the action provider.

The operation does not remove the ActionIdentity or ActionDefinition object from the COM archive, merely removes the objects from the provider. This permits existing, and completed, ActionInstance objects to continue to reference the correct ActionIdentity and ActionDefinition objects in the COM archive.

Operation Identifier	removeAction	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	actionInstIds : (List<MAL::Long>)

3.2.14.2 Structures

3.2.14.2.1 The actionInstIds field shall hold the object instance identifiers of the ActionIdentity objects to be removed from the provider.

3.2.14.2.2 The wildcard value of '0' in the list of object instance identifiers shall be supported and matches all actions of the provider.

3.2.14.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.2.14.2.4 If a provided ActionIdentity object instance identifier does not include a wildcard and does not match an existing ActionIdentity object then this operation shall fail with an UNKNOWN error.

3.2.14.2.5 If a matched definition is still being used by an executing action instance then this operation shall not fail because of this reason.

3.2.14.2.6 Matched ActionIdentity objects shall not be removed from the COM archive only the list of ActionIdentity objects in the provider.

3.2.14.2.7 Removed ActionIdentity object shall not be allowed to be referenced by new action instances.

3.2.14.2.8 If an error is raised then no actions shall be removed as a result of this operation call.

3.2.14.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

a) one of the supplied ActionIdentity object instance identifiers is unknown;

b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.3 SERVICE: PARAMETER

3.3.1 OVERVIEW

The parameter service allows the user to subscribe to parameter value report and optionally be able to set new values. A single PUBSUB operation is provided for monitoring and publishing of parameter values.

A parameter value also contains a calculation of the validity of the parameter, the flow chart for this calculation is provided in figure 3-3.
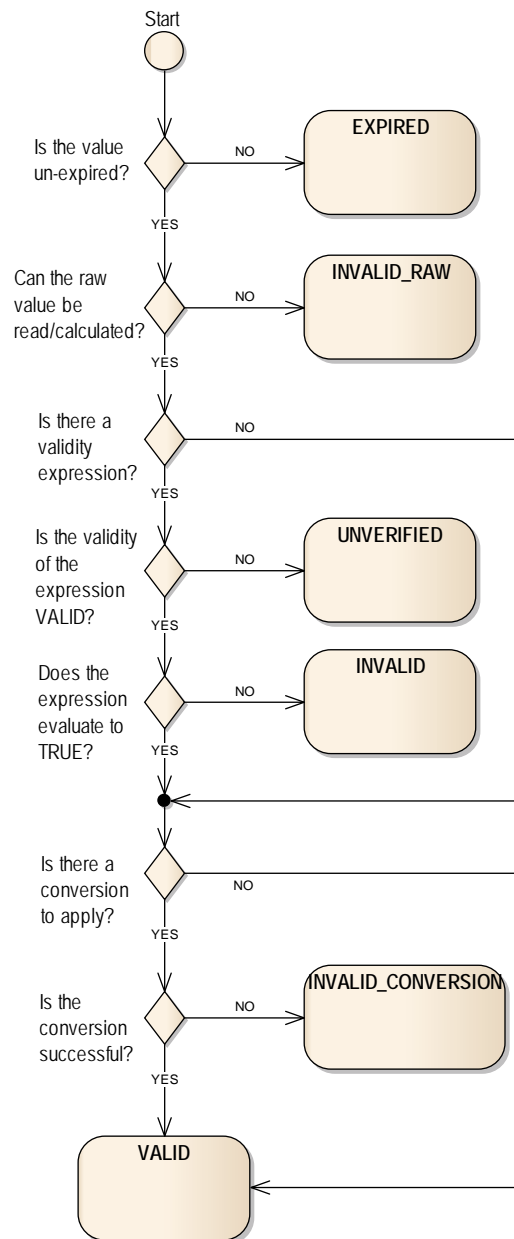
Figure 3-3: Flow Chart for Determining the Validity of a Parameter

This standard supports the concept of non-standard invalidity states but the meaning and calculation of these is outside the scope of this standard.

The generation of value reports can be controlled using the enableGeneration operation, which supports the use of groups. Groups must reference parameter identities or groups of parameter identities only.

The parameter service does not include any value checking, this is delegated to the check service.

Parameter definitions are maintained using the operations defined in this service but storage of definitions is delegated to the COM archive.

Table 3-4: Parameter Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MC	Parameter	4	2	1
Interaction Pattern	Operation Identifier	Operation Number	Support in Replay	Capability Set
PUBLISH-SUBSCRIBE	monitorValue	1	Yes	1
REQUEST	getValue	2	Yes	2
SUBMIT	setValue	3	No	3
SUBMIT	enableGeneration	4	No	4
REQUEST	listDefinition	5	Yes	5
REQUEST	addParameter	6	No	
REQUEST	updateDefinition	7	No	6
SUBMIT	removeParameter	8	No	

3.3.2   HIGH-LEVEL REQUIREMENTS

3.3.2.1   The parameter service shall provide:

a) the capability for periodic reporting of predefined parameter values;

b) the capability for setting the raw parameter value of a set of parameters;

c) the capability for requesting the current value of a set of parameters;

d) the capability for controlling reporting of the parameter values;

e) the capability for maintaining the list of parameter definitions.

3.3.2.2   Each parameter value report shall contain:

a) The validity of the parameter value;

b) The raw value of the parameter;

c) The converted value of the parameter, if applicable.

3.3.2.3 The validity of parameter shall be calculated against validity criteria in the parameter definition.

3.3.2.4 It shall be possible to update, delete, and add parameter definitions.

3.3.2.5 The list of the parameters definitions may be requested.

3.3.3 FUNCTIONAL REQUIREMENTS

3.3.3.1 The generation of reports of the parameter shall be controlled by the generationEnabled Boolean value in the ParameterDefinitionDetails.

3.3.3.2 Setting the generationEnabled field to TRUE shall enable generation of reports for a specific parameter.

3.3.3.3 If a parameter is required to send periodic reports then the time between these reports shall be controlled using the reportInterval Duration value in the ParameterDefinitionDetails.

3.3.3.4 A reportInterval value of '0' shall mean no periodic reports shall be sent; reports must be triggered by another implementation-specific mechanism in this case.

3.3.3.5 The generationEnabled values can be set by using the enableGeneration operation.

3.3.3.6 The generationEnabled and reportInterval values can be set by using the updateDefinition operation.

3.3.3.7 For onboard parameters, the interval should be a multiple of the minimum sampling interval of those parameters.

3.3.3.8 If an interval that is not supported by the provider is requested, then an INVALID error shall be returned, and the change rejected.

3.3.3.9 If the parameter is periodic and reported via a periodic aggregation, and a new report has not been received in the aggregation period, then the validity shall be set to EXPIRED by setting the ParameterValue validityState field to '1'.

3.3.3.10 If the parameter raw value cannot be obtained, or calculated for synthetic parameters, then the validity state shall be INVALID_RAW by setting the ParameterValue validityState field to '2' and the rawValue set to NULL.

3.3.3.11 If the validityExpression field is NULL, or the validity expression evaluates to TRUE, and the ParameterDefinitionDetails either does not contain a conversion or the

conversion is successful, then the validity state shall be set to VALID by setting the ParameterValue validityState field to '0'.

3.3.3.12 If the validityExpression field is NULL, or the validity expression evaluates to TRUE, and the conversion of the parameter value fails (for example an unexpected value for a discrete conversion), then the validity state shall be INVALID_CONVERSION by setting the ParameterValue validityState field to '3'.

3.3.3.13 If the validityExpression field is not NULL and the parameters used to determine the expression are not VALID, then the validity state shall be set to UNVERIFIED by setting the ParameterValue validityState field to '4'.

3.3.3.14 If the validityExpression field is not NULL and the validity expression evaluates to FALSE, then the validity state shall be set to INVALID by setting the ParameterValue validityState field to '5'.

3.3.3.15 If the validity of the parameter is INVALID_CONVERSION then the convertedValue of the ParameterValue shall be set to NULL.

3.3.3.16 If the validity of the parameter is either UNVERIFIED or INVALID and the ParameterDefinitionDetails contains a conversion, then the convertedValue of the ParameterValue shall contain the converted value.

3.3.3.17 It is supported that other, implementation specific, mechanisms may be used to determine parameter validity.

3.3.3.18 It is supported that other, deployment specific, validityState values be used; however, the meaning of those values is deployment specific and outside the scope of this standard. They must use values greater than 127.

3.3.4 COM USAGE

3.3.4.1 Each parameter of a provider shall be represented by a ParameterIdentity COM object.

3.3.4.2 The body of the ParameterIdentity COM object shall hold the name of the parameter.

3.3.4.3 The definitions of the parameters shall be represented as ParameterDefinition COM objects.

3.3.4.4 Parameter value report shall be represented as ParameterValueInstance COM objects.

3.3.4.5 The ParameterDefinition object shall use the related link to indicate which ParameterIdentity object it uses.

3.3.4.6 The ParameterValueInstance object shall use the related link to indicate which ParameterDefinition object it uses.

3.3.4.7 The source link of the ParameterIdentity object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.3.4.8 The source link of the ParameterDefinition object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.3.4.9 The source link of the ParameterValueInstance object should be the object that caused the report to be generated.

3.3.4.10 The source link of the ParameterValueInstance object shall be NULL for periodic reports.

Table 3-5: Parameter Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
ParameterIdentity	1	MAL::Identifier	Set to NULL	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
ParameterDefinition	2	ParameterDefinitionDetails	1	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
ParameterValueInstance	3	ParameterValue	2	The object that caused the value to be generated or NULL for periodic reports.

3.3.5 DISCUSSION—COM OBJECT RELATIONSHIPS

Figure 3-4 below shows the COM object relationships for this service:
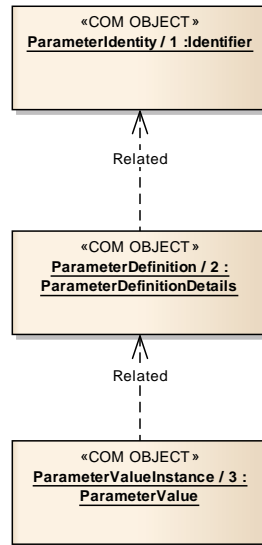
Figure 3-4: Parameter Service COM Object Relationships

3.3.6 COM ARCHIVE SERVICE USAGE

3.3.6.1 ParameterIdentity and ParameterDefinition objects should be stored in the COM archive.

3.3.6.2 When a parameter value report is published, the ParameterValueInstance object should be stored in the COM archive.

3.3.7   OPERATION: MONITORVALUE

3.3.7.1   Overview

The monitorValue operation allows a consumer to subscribe for parameter value reports.

Operation Identifier	monitorValue	
Interaction Pattern	PUBLISH-SUBSCRIBE	
Pattern Sequence	Message	Body Signature
OUT	PUBLISH/NOTIFY	objId : (COM::ObjectId) newValue : (ParameterValue)

3.3.7.2   Structures

3.3.7.2.1   The MAL EntityKey.firstSubKey shall contain the parameter name.

3.3.7.2.2   The MAL EntityKey.secondSubKey shall contain the ParameterIdentity object instance identifier.

3.3.7.2.3   The MAL EntityKey.thirdSubKey shall contain the ParameterDefinition object instance identifier.

3.3.7.2.4   The   MAL   EntityKey.fourthSubKey   shall   contain   the   new ParameterValueInstance object instance identifier.

3.3.7.2.5   The timestamp of the ParameterValueInstance report shall be taken from the publish message and shall be the time of the parameter value update.

3.3.7.2.6   The publish message shall include the ObjectId of the source link of the report.

3.3.7.2.7   If no source link is needed then the ObjectId shall be replaced with a NULL.

3.3.7.2.8   The second part of the publish message shall be the ParameterValueInstance object value.

3.3.7.3   Errors

The operation does not return any errors.

3.3.8 OPERATION: GETVALUE

3.3.8.1 Overview

The getValue operation returns the latest received value for a requested parameter.

Operation Identifier	getValue	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	paramInstIds : (List<MAL::Long>)
OUT	RESPONSE	paramValDetails : (List<ParameterValueDetails>)

3.3.8.2 Structures

3.3.8.2.1 The paramInstIds field shall provide the list of ParameterIdentity object instance identifiers.

3.3.8.2.2 The wildcard value of '0' shall be supported and matches all parameters of the provider.

3.3.8.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.3.8.2.4 If a requested parameter is unknown then an UNKNOWN error shall be returned.

3.3.8.2.5 If a parameter is being reported periodically, using the operation shall not reset the reportInterval timer.

3.3.8.2.6 The response shall contain a list of returned ParameterIdentity and ParameterDefinition object instance identifier pairs and a matching list of parameter values.

3.3.8.2.7 The new value shall not be published via the monitorValue operation.

3.3.8.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

    a) one or more of the requested parameters is unknown;

    b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.3.9 OPERATION: SETVALUE

3.3.9.1 Overview

The setValue operation allows a consumer to set the raw value for one or more parameters.

Operation Identifier	setValue	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	newRawValues : (List<ParameterRawValue>)

3.3.9.2 Structures

3.3.9.2.1 The submitted newRawValues shall hold a list of ParameterRawValues that contain the ParameterIdentity object instance identifier and the respective raw value to be set.

3.3.9.2.2 If the paramInstId field contains the wildcard value of '0' then an INVALID error shall be returned.

3.3.9.2.3 If a requested ParameterIdentity is unknown then an UNKNOWN error shall be returned.

3.3.9.2.4 If a request ParameterIdentity is not settable because of its being read only, then a READONLY error shall be returned.

3.3.9.2.5 The rawValue shall contain the new parameter raw value to be set.

3.3.9.2.6 If the supplied new parameter raw value does not match the defined type for the ParameterIdentity then an INVALID error shall be returned.

3.3.9.2.7 If an error is raised then no modifications shall be made as a result of this operation call.

3.3.9.2.8 The parameter values shall be set concurrently, by this it is meant that all values are set at the same time without interleaving of other values being (ATOMIC behaviour). How this is implemented is an implementation detail.

3.3.9.2.9 The service provider shall create new ParameterValueInstance objects for the updated parameter values, store these in the COM Archive, and publish these new values.

3.3.9.3 Errors

The operation may return one of the following errors:

    a) ERROR: UNKNOWN:

        1) one or more of the referenced parameters is unknown;

2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

b) ERROR: INVALID:

1) one of the supplied parameter values contains an invalid value;

2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

c) ERROR: READONLY:

1) one or more of the parameters being set is read only;

2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
READONLY	70020	List<MAL::UInteger>

3.3.10 OPERATION: ENABLEGENERATION

3.3.10.1 Overview

The enableGeneration operation allows a consumer to control whether reports for specific parameters are generated or not. The operation allows the consumer to select the parameters directly or indirectly using groups.

Operation Identifier	enableGeneration	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	isGroupIds : (MAL::Boolean) enableInstances : (List<COM::InstanceBooleanPair>)

3.3.10.2 Structures

3.3.10.2.1 If the isGroupIds field is TRUE then the enableInstances field shall contain GroupIdentity object instance identifiers; otherwise the field contains ParameterIdentity object instance identifiers.

3.3.10.2.2 The ParameterIdentity objects referenced, either directly or indirectly via groups, by the enableInstances field shall be the ParameterIdentity objects to match.

3.3.10.2.3 The id of the enableInstances field shall support the wildcard value of '0' and matches all ParameterIdentity objects of the provider.

3.3.10.2.4 The service provider shall check for the wildcard value in the list of object instance identifiers in the enableInstances field first and if found no other checks of supplied object instance identifiers shall be made.

3.3.10.2.5 If the enableInstances field contains a value of TRUE then reports for matching ParameterIdentity objects shall be generated, a value of FALSE requests that reports will not be generated.

3.3.10.2.6 No error shall be raised if the enableInstances Boolean value supplied is the same as the current generationEnabled field of the definition for a matched ParameterIdentity object; i.e., enabling an already enabled parameter will not result in an error.

3.3.10.2.7 If a requested ParameterIdentity or GroupIdentity object is unknown then an UNKNOWN error shall be returned.

3.3.10.2.8 If a requested Group, or the Group objects referenced by that Group, does not contain ParameterIdentity objects then an INVALID error shall be returned.

3.3.10.2.9 If an error is raised then no modifications shall be made as a result of this operation call.

3.3.10.2.10 The provider shall create and store a new ParameterDefinition object in the COM archive if the generationEnabled field is changed.

3.3.10.2.11 If a new ParameterDefinition object is created then that new object shall be the current ParameterDefinition used for the specific ParameterIdentity.

3.3.10.2.12 If the generation of reports is being enabled, and the parameter is defined as being periodic, then the provider shall generate a report immediately and start the report interval from that report.

3.3.10.3 Errors

The operation may return one of the following errors:

a) ERROR: UNKNOWN:

   1) one or more of the requested parameters or groups is unknown;

   2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

b) ERROR: INVALID:

   1) one of the supplied groups is either not a group of groups or a group of ParameterIdentity objects;

   2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

3.3.11 OPERATION: LISTDEFINITION

3.3.11.1 Overview

The listDefinition operation allows a consumer to request the latest object instance identifiers of the ParameterIdentity and ParameterDefinition objects for the supported parameters of the provider.

Operation Identifier	listDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	paramNames : (List<MAL::Identifier>)
OUT	RESPONSE	objInstIds : (List<ObjectInstancePair>)

3.3.11.2 Structures

3.3.11.2.1 The paramNames field shall contain a list of parameter names to retrieve the ParameterIdentity and ParameterDefinition object instance identifiers for.

3.3.11.2.2 The paramNames field may contain the wildcard value of '*' to return all supported ParameterIdentity and ParameterDefinition objects.

3.3.11.2.3 The wildcard value should be checked for first, if found no other checks of supplied identifiers shall be made.

3.3.11.2.4 If a provided identifier does not include a wildcard and does not match an existing ParameterIdentity object then this operation shall fail with an UNKNOWN error.

3.3.11.2.5 The response shall contain a list of matching ParameterIdentity and ParameterDefinition object instance identifier pairs.

3.3.11.2.6 The returned list shall maintain the same order as the submitted list unless the wildcard value was included in the request.

3.3.11.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

   a) one of the supplied identifiers is unknown;

   b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.3.12 OPERATION: ADDPARAMETER

3.3.12.1 Overview

The addParameter operation allows a consumer to define one or more parameters that do not currently exist.

The new ParameterIdentity and ParameterDefinition objects are expected to be stored in the COM archive by the provider of the parameter service.

Operation Identifier	addParameter	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	paramDefDetails : (List<ParameterCreationRequest>)
OUT	RESPONSE	newObjInstIds : (List<ObjectInstancePair>)

3.3.12.2 Structures

3.3.12.2.1 The paramDefDetails field shall hold the name and the ParameterDefinitionDetails to be added.

3.3.12.2.2 The name field must not be the wildcard '*', or empty (an INVALID error shall be returned in this case).

3.3.12.2.3 If the supplied reportInterval value is not supported by the provider then an INVALID error shall be returned.

3.3.12.2.4 The supplied name must be unique among all ParameterIdentity objects for the domain of the provider; otherwise a DUPLICATE error shall be raised.

3.3.12.2.5 If an error is raised then no new identities and definitions shall be added as a result of this operation call.

3.3.12.2.6 If the supplied name matches an existing, but removed, ParameterIdentity then that ParameterIdentity shall be reused; otherwise a new ParameterIdentity shall be created.

3.3.12.2.7 The provider shall create a new ParameterDefinition object and store it, and any new ParameterIdentity objects, in the COM archive.

3.3.12.2.8 The response shall contain the list of object instance identifiers for the ParameterIdentity and new ParameterDefinition objects.

3.3.12.2.9 The returned list shall maintain the same order as the submitted definitions.

3.3.12.3 Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

1) one of the supplied ParameterIdentity objects contains an invalid name or a supplied interval is not supported by the provider;

2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: DUPLICATE:

1) one or more of the ParameterIdentity objects being added has supplied a parameter name that is already in use in the domain;

2) the extra information field contains a list of the indexes of the erroneous values from the originating request list.

Error	Error #	ExtraInfo Type
DUPLICATE	Defined in COM	List<MAL::UInteger>

3.3.13 OPERATION: UPDATEDEFINITION

3.3.13.1 Overview

The updateDefinition operation allows a consumer to update a definition for one or more parameters.

This differs from deleting an existing parameter and adding a new definition with the same parameter name in the fact that the ParameterIdentity object is not changed between the two definitions.

The replacement definition is expected to be stored in the COM archive by the service provider. The operation does not remove the previous object from the COM archive, but merely removes the object from the provider.

Operation Identifier	updateDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	paramInstIds : (List<MAL::Long>) paramDefDetails : (List<ParameterDefinitionDetails>)
OUT	RESPONSE	newObjInstIds : (List<MAL::Long>)

3.3.13.2 Structures

3.3.13.2.1 The paramInstIds field shall contain the object instance identifiers of the ParameterIdentity objects to be updated.

3.3.13.2.2 The supplied object instance identifiers shall match existing identity objects, an UNKNOWN error shall be raised if this is not the case.

3.3.13.2.3 If the paramInstIds list contains either NULL or '0' an INVALID error shall be raised.

3.3.13.2.4 The paramDefDetails field shall contain the replacement ParameterDefinitionDetails.

3.3.13.2.5 The two lists shall be ordered the same.

3.3.13.2.6 The number of entries in the two lists shall be the same size; otherwise an INVALID error shall be returned.

3.3.13.2.7 If the supplied reportInterval value is not supported by the provider then an INVALID error shall be returned.

3.3.13.2.8 If an error is raised then no definitions shall be updated as a result of this operation call.

3.3.13.2.9 The provider shall create a new ParameterDefinition object and store it in the COM archive.

3.3.13.2.10 The new ParameterDefinition object shall be the current ParameterDefinition used for the specific ParameterIdentity.

3.3.13.2.11 The response shall contain the list of object instance identifiers for the new ParameterDefinition objects.

3.3.13.2.12 The returned list shall maintain the same order as the submitted definitions.

3.3.13.3 Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

1) the supplied object instance identifiers list contains either a NULL or '0' or the two supplied lists are not the same length or a supplied interval is not supported by the provider;

2) if the two lists are not the same length then the extra information field shall contain the first index of the element in the largest list which does not have corresponding element in the other list;

3) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: UNKNOWN:

1) one of the supplied ParameterIdentity object instance identifiers is unknown;

2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.3.14 OPERATION: REMOVEPARAMETER

3.3.14.1 Overview

The removeParameter operation allows a consumer to remove one or more parameters from the list of parameters supported by the parameter provider.

The operation does not remove the ParameterIdentity or ParameterDefinition objects from the COM archive, merely removes the objects from the provider. This permits existing parameter values to continue to reference the correct ParameterIdentity and ParameterDefinition objects in the COM archive.

Operation Identifier	removeParameter	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	paramInstIds : (List<MAL::Long>)

3.3.14.2 Structures

3.3.14.2.1 The paramInstIds field shall hold the object instance identifiers of the ParameterIdentity objects to be removed from the provider.

3.3.14.2.2 The list may contain the wildcard value of '0'.

3.3.14.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.3.14.2.4 If a provided ParameterIdentity object instance identifier does not include a wildcard and does not match an existing parameter identity object then this operation shall fail with an UNKNOWN error.

3.3.14.2.5 Matched ParameterIdentity and ParameterDefinition objects shall not be removed from the COM archive only the list of ParameterIdentity and ParameterDefinition objects from the provider.

3.3.14.2.6 If an error is raised then no parameters shall be removed as a result of this operation call.

3.3.14.2.7 If the operation succeeds then the provider shall not publish parameter values for the deleted ParameterIdentity objects anymore.

3.3.14.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

    a) one of the supplied ParameterIdentity object instance identifiers is unknown;

    b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.4   SERVICE: ALERT

3.4.1   OVERVIEW

The alert service defines the structures and patterns for the publishing and monitoring of alerts. The alert service uses COM event service to monitor and publish alert events.

The generation of alerts can be controlled using the enableGeneration operation, which supports the use of groups. Groups must reference either other groups or alerts only.

Alert definitions are maintained using the operations defined in this service but storage of definitions is delegated to the COM archive.

Table 3-6: Alert Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MC	Alert	4	3	1
Interaction Pattern	**Operation Identifier**	**Operation Number**	**Support in Replay**	**Capability Set**
SUBMIT	enableGeneration	1	No	1
REQUEST	listDefinition	2	Yes	2
REQUEST	addAlert	3	No	
REQUEST	updateDefinition	4	No	3
SUBMIT	removeAlert	5	No	

3.4.2   HIGH-LEVEL REQUIREMENTS

3.4.2.1   The alert service shall provide:

   a) the capability for reporting of alerts;

   b) the capability for controlling reporting of alerts;

   c) the capability for maintaining the list of alert definitions.

3.4.2.2   The alert reporting service shall provide the capability to enable and disable the generation of alert reports.

3.4.2.3   It shall be possible to update, delete, and add alert definitions.

3.4.2.4   The list of the alert definitions may be requested.

3.4.3 FUNCTIONAL REQUIREMENTS

3.4.3.1   The generation of alerts shall be controlled by the generationEnabled Boolean value in the AlertDefinitionDetails.

3.4.3.2   Setting the generationEnabled field to TRUE shall enable generation of instances of that alert.

3.4.3.3   The generationEnabled value can be set by using either the enableGeneration or updateDefinition operations.

3.4.3.4   The alert instances shall be published as events using the COM event operations.

3.4.4 COM USAGE

3.4.4.1   Each alert of a provider shall be represented by an AlertIdentity COM object.

3.4.4.2   The body of the AlertIdentity COM object shall hold the name of the alert.

3.4.4.3   The definitions of the alerts shall be represented as AlertDefinition COM objects.

3.4.4.4   Instances of an alert shall be represented as an AlertEvent COM event.

3.4.4.5   The AlertDefinition object shall use the related link to indicate which AlertIdentity object it uses.

3.4.4.6   The AlertEvent object shall use the related link to indicate which AlertDefinition object it uses.

3.4.4.7   The source link of the AlertIdentity object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.4.4.8   The source link of the AlertDefinition object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

Table 3-7: Alert Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
AlertIdentity	1	MAL::Identifier	Set to NULL	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
AlertDefinition	2	AlertDefinitionDetails	1	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.4.5   COM EVENT SERVICE USAGE

3.4.5.1   When an alert is generated it shall be represented as a COM event.

3.4.5.2   The event shall be published using the COM event service.

3.4.5.3   The source link of the AlertEvent object shall be the object that caused the event to be generated.

Table 3-8: Alert Service Events

Event Name	Object Number	Object Body Type	Related points to	Source points to
AlertEvent	3	AlertEventDetails	2	The object that caused the event to be generated.

3.4.6   DISCUSSION—COM OBJECT RELATIONSHIPS

Figure 3-5 below shows the COM object and event relationships for this service.
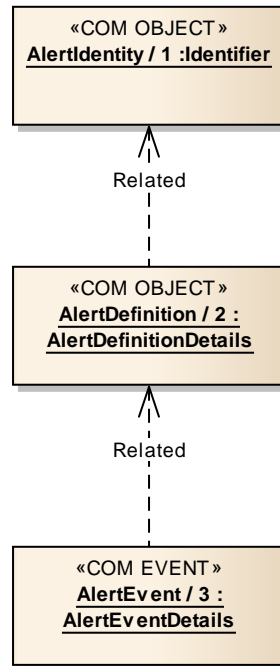
Figure 3-5: Alert Service COM Object and Event Relationships

3.4.7   COM ARCHIVE SERVICE USAGE

3.4.7.1   AlertIdentity and AlertDefinition objects should be stored in the COM archive.

3.4.7.2   When an alert event is published, the AlertEvent object should be stored in the COM archive by the publisher.

3.4.8   OPERATION: ENABLEGENERATION

3.4.8.1   Overview

The enableGeneration operation allows a consumer to control whether updates for specific alerts are generated or not. The operation allows the consumer to select the alerts directly or indirectly using groups.

Operation Identifier	enableGeneration	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	isGroupIds : (MAL::Boolean) enableInstances : (List<COM::InstanceBooleanPair>)

3.4.8.2   Structures

3.4.8.2.1   If the isGroupIds field is TRUE then the enableInstances field shall contain GroupIdentity object instance identifiers; otherwise the field contains AlertIdentity object instance identifiers.

3.4.8.2.2   The AlertIdentity objects referenced, either directly or indirectly via groups, by the enableInstances field shall be the AlertIdentity objects to match.

3.4.8.2.3   The id of the enableInstances field shall support the wildcard value of '0' and matches all AlertIdentity objects of the provider.

3.4.8.2.4   The service provider shall check for the wildcard value in the list of object instance identifiers in the enableInstances field first and if found no other checks of supplied object instance identifiers shall be made.

3.4.8.2.5   If the enableInstances field contains a value of TRUE then instances of matching AlertIdentity objects shall be generated, a value of FALSE requests that instances will not be generated.

3.4.8.2.6   No error shall be raised if the enableInstances Boolean value supplied is the same as the current generationEnabled field for an alert object; i.e., enabling an already enabled alert will not result in an error.

3.4.8.2.7   If a requested AlertIdentity or GroupIdentity object is unknown then an UNKNOWN error shall be returned.

3.4.8.2.8   If a requested Group, or the Group objects referenced by that Group, does not contain AlertIdentity objects then an INVALID error shall be returned.

3.4.8.2.9   If an error is raised then no modifications shall be made as a result of this operation call.

3.4.8.2.10 The provider shall create and store a new AlertDefinition object in the COM archive if the generationEnabled field is changed.

3.4.8.2.11 If a new AlertDefinition object is created then that new object shall be the current AlertDefinition used for the specific AlertIdentity.

3.4.8.3 Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

1) one of the supplied groups is either not a group of groups or a group of AlertIdentity objects;

2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: UNKNOWN:

1) one or more of the requested alerts or group objects is unknown;

2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.4.9   OPERATION: LISTDEFINITION

3.4.9.1   Overview

The listDefinition operation allows a consumer to request the latest object instance identifiers of the AlertIdentity and AlertDefinition objects for the supported alerts of the provider.

Operation Identifier	listDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	alertNames : (List<MAL::Identifier>)
OUT	RESPONSE	alertObjInstIds : (List<ObjectInstancePair>)

3.4.9.2   Structures

3.4.9.2.1   The alertNames field shall contain a list of alert names to retrieve the AlertIdentity and AlertDefinition object instance identifiers for.

3.4.9.2.2   The alertNames field may contain the wildcard value of '*' to return all supported AlertIdentity and AlertDefinition objects.

3.4.9.2.3   The wildcard value should be checked for first, if found no other checks of supplied identifiers shall be made.

3.4.9.2.4   If a provided identifier does not include a wildcard and does not match an existing AlertIdentity object then this operation shall fail with an UNKNOWN error.

3.4.9.2.5   The response shall contain a list of matching AlertIdentity and AlertDefinition object instance identifiers.

3.4.9.2.6   The returned list shall maintain the same order as the submitted list unless the wildcard value was included in the request.

3.4.9.3   Errors

The operation may return the following error: ERROR: UNKNOWN:

   a)   one of the supplied identifiers is unknown;

   b)   a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.4.10 OPERATION: ADDALERT

3.4.10.1 Overview

The addAlert operation allows a consumer to define one or more alerts that do not currently exist.

The new AlertIdentity and AlertDefinition objects are expected to be stored in the COM archive by the provider of the alert service.

Operation Identifier	addAlert	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	alertDefDetails : (List<AlertCreationRequest>)
OUT	RESPONSE	newObjInstIds : (List<ObjectInstancePair>)

3.4.10.2 Structures

3.4.10.2.1 The alertDefDetails field shall hold the name and the AlertDefinitionDetails to be added.

3.4.10.2.2 The name field must not be the wildcard '*', or empty (an INVALID error shall be returned in this case).

3.4.10.2.3 The supplied name must be unique among all AlertIdentity objects for the domain of the provider; otherwise a DUPLICATE error shall be raised.

3.4.10.2.4 If an error is raised then no new identities and definitions shall be added as a result of this operation call.

3.4.10.2.5 If the supplied name matches an existing, but removed, AlertIdentity then that AlertIdentity shall be reused; otherwise a new AlertIdentity shall be created.

3.4.10.2.6 The provider shall create a new AlertDefinition object and store it, and any new AlertIdentity objects, in the COM archive.

3.4.10.2.7 The response shall contain the list of object instance identifiers for the AlertIdentity and new AlertDefinition objects.

3.4.10.2.8 The returned list shall maintain the same order as the submitted definitions.

3.4.10.3 Errors

The operation may return one of the following errors:

   a) ERROR: DUPLICATE:

1) one or more of the AlertIdentity objects being added has supplied an alert name that is already in use in the domain;

2) the extra information field contains a list of the indexes of the erroneous values from the originating request list;

Error	Error #	ExtraInfo Type
DUPLICATE	Defined in COM	List<MAL::UInteger>

b) ERROR: INVALID:

1) one of the supplied AlertIdentity objects contains an invalid name;

2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

3.4.11 OPERATION: UPDATEDEFINITION

3.4.11.1 Overview

The updateDefinition operation allows a consumer to update a definition for one or more alerts.

This differs from deleting an existing alert and adding a new definition with the same alert name in the fact that the AlertIdentity object is not changed between the two definitions.

The replacement definition is expected to be stored in the COM archive by the service provider. The operation does not remove the previous object from the COM archive, merely removes the object from the provider.

Operation Identifier	updateDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	alertObjInstIds : (List<MAL::Long>) alertDefDetails : (List<AlertDefinitionDetails>)
OUT	RESPONSE	newObjInstIds : (List<MAL::Long>)

3.4.11.2 Structures

3.4.11.2.1 The alertObjInstIds field shall contain the object instance identifiers of the AlertIdentity objects to be updated.

3.4.11.2.2 The supplied object instance identifiers shall match existing identity objects, an UNKNOWN error shall be raised if this is not the case.

3.4.11.2.3 If the alertObjInstIds list contains either NULL or '0' an INVALID error shall be raised.

3.4.11.2.4 The alertDefDetails field shall contain the replacement AlertDefinitionDetails.

3.4.11.2.5 The two lists shall be ordered the same.

3.4.11.2.6 The number of entries in the two lists shall be the same size; otherwise an INVALID error shall be returned.

3.4.11.2.7 If an error is raised then no definitions shall be updated as a result of this operation call.

3.4.11.2.8 The provider shall create a new AlertDefinition object and store it in the COM archive.

3.4.11.2.9 The new AlertDefinition object shall be the current AlertDefinition used for the specific AlertIdentity.

3.4.11.2.10 The response shall contain the list of object instance identifiers for the new AlertDefinition objects.

3.4.11.2.11 The returned list shall maintain the same order as the submitted definitions.

3.4.11.3 Errors

The operation may return one of the following errors:

    a) ERROR: INVALID:

        1) one of the supplied AlertDefinition objects contains an invalid value or the two supplied lists are not the same length;

        2) if the two lists are not the same length then the extra information field shall contain the first index of the element in the largest list which does not have corresponding element in the other list;

        3) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

    b) ERROR: UNKNOWN:

        1) one of the supplied AlertIdentity object instance identifiers is unknown;

        2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.4.12 OPERATION: REMOVEALERT

3.4.12.1 Overview

The removeAlert operation allows a consumer to remove one or more definitions from the list of alerts supported by the alert provider.

The operation does not remove the AlertIdentity or AlertDefinition objects from the COM archive, merely removes the objects from the provider. This permits existing AlertEvent objects to continue to reference the correct AlertDefinition object in the COM archive.

Operation Identifier	removeAlert	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	alertInstIds : (List<MAL::Long>)

3.4.12.2 Structures

3.4.12.2.1 The alertInstIds field shall hold the object instance identifiers of the AlertIdentity objects to be removed from the provider.

3.4.12.2.2 The list may contain the wildcard value of '0'.

3.4.12.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.4.12.2.4 If a provided AlertIdentity object instance identifier does not include a wildcard and does not match an existing AlertIdentity object then this operation shall fail with an UNKNOWN error.

3.4.12.2.5 Matched AlertIdentity objects shall not be removed from the COM archive only the list of AlertIdentity objects in the provider.

3.4.12.2.6 If an error is raised then no alerts shall be removed as a result of this operation call.

3.4.12.2.7 If the operation succeeds then the provider shall not publish AlertEvent events for the deleted AlertIdentity objects anymore.

3.4.12.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

    a)   one of the supplied AlertIdentity object instance identifiers is unknown;

    b)   a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.5   SERVICE: CHECK

3.5.1   OVERVIEW

The check service allows the user to monitor and control the checking of parameters values against check definitions including limit sets. Violations of these defined checks are published using the COM event service.

The check service allows the consumer to define checks and then link the check definition to a parameter to be monitored. Figure 3-6 shows the nominal sequence of operations for the check service:
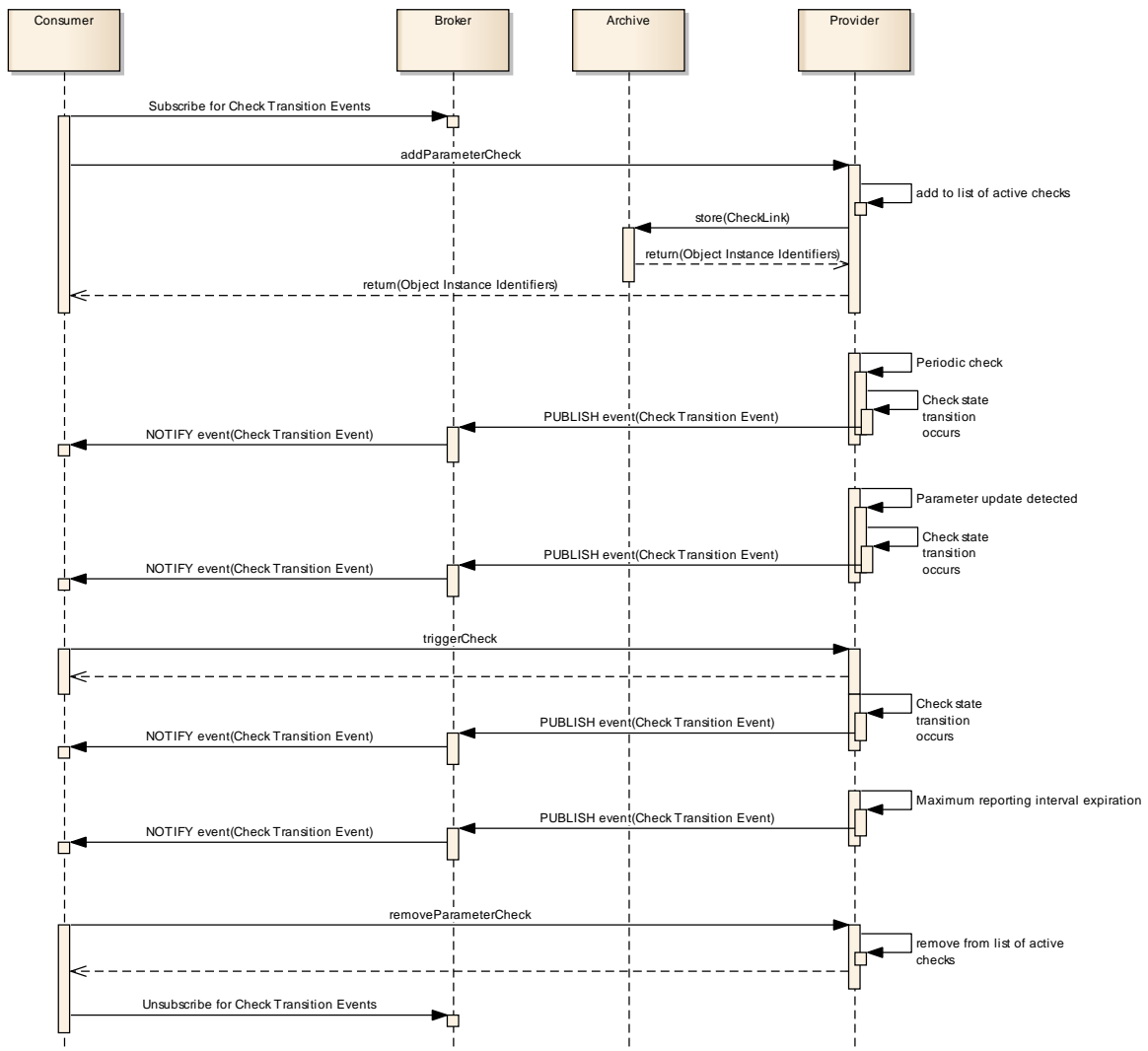
Figure 3-6:   Check Service Nominal Sequence

A consumer first subscribes for check events before linking some already existing checks to some parameters. The figure shows how check evaluation is normally triggered by either a

periodic interval or a change in a monitored value; however, the service also supports the optional ability for consumers to trigger the evaluation of checks using the triggerCheck operation. Finally, if a check event has not been generated for a configurable period of time, known as the maximum reporting interval, an event is generated regardless of whether the check is in violation or not. This maximum reporting interval supports the situation where regular confirmation of the non-violating state of a check is required.

The list of currently violating checks can be obtained using the getCurrentTransitionList operation; to get a full report including non-violating checks the getSummaryReport operation should be used.

Figure 3-7 shows the flow chart for determining the status of a check.
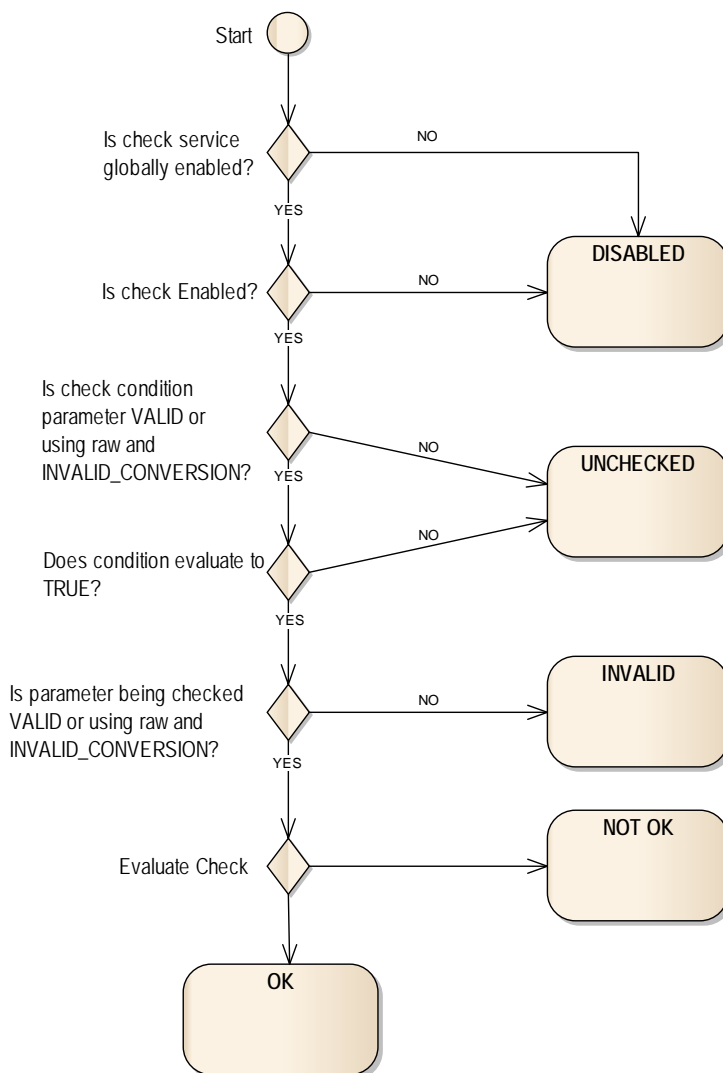
Figure 3-7: Flow Chart for Determining the Status of a Check

Each time a check is evaluated, the procedure indicated in the flow chart depicted in figure 3-7 is performed to calculate the check result. Each check link can have a condition

included which determines whether the check should be applied or not; this allows several checks to be associated to a single parameter and applied in different conditions.

The service defines five basic check types, constant, reference, delta, limit, and compound. A constant check is used to ensure that a parameter value either does not change or does not change from a set of specific values.

In the Reference and Delta checks, a value is compared against another value which serves as a reference value. The reference value can be taken from either another parameter or the parameter being checked, and can be the previous value or a value in the past, specified by using a delta time in the ReferenceValue composite. This does not affect the logic of the flow chart above.

Limit checks monitor parameters to see if the parameter value is either inside or outside a range of values.

The final check type, compound, monitors a set of previously defined checks and only violates when a specified number of those monitored checks violate themselves.

The service only publishes check results (as check transition events) when either the result changes from the previous value (for example a check starts to violate), or the maximum reporting interval expires. This reduces the amount of 'no change' reporting data being distributed. For situations where regular reporting of the check result is required, even when no change is detected, a low maximum reporting interval value should be specified.

Table 3-9: Check Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MC	Check	4	4	1
Interaction Pattern	Operation Identifier	Operation Number	Support in Replay	Capability Set
PROGRESS	getCurrentTransitionList	1	Yes	1
PROGRESS	getSummaryReport	2	Yes	
SUBMIT	enableService	3	No	2
REQUEST	getServiceStatus	4	No	
SUBMIT	enableCheck	5	No	3
SUBMIT	triggerCheck	6	No	4
REQUEST	listDefinition	7	No	5
REQUEST	listCheckLinks	8	No	
REQUEST	addCheck	9	No	6
REQUEST	updateDefinition	10	No	
SUBMIT	removeCheck	11	No	
REQUEST	addParameterCheck	12	No	7
SUBMIT	removeParameterCheck	13	No	

3.5.2 HIGH-LEVEL REQUIREMENTS

3.5.2.1   The check service shall provide:

a)   the capability for reporting the check transitions;

b)   the capability for requesting the latest check transitions;

c)   the capability for controlling the global check evaluation;

d)   the capability for controlling the individual check reporting;

e)   the capability for triggering the evaluation of checks;

f)   the capability for adding check definitions;

g)   the capability for modifying check definitions;

h)   the capability for removing check definitions;

i)   the capability for listing check definitions;

j)   the capability for modifying the association of checks to parameters.

3.5.2.2   The check service shall use the event service for issuing the corresponding event generated by the check service.

3.5.2.3   The check may support the evaluation of the following check types:

a)   Constant check;

b)   Reference check;

c)   Delta check;

d)   Limit check;

e)   Compound check.

3.5.2.4   When performing a limit check, the check service shall:

a)   check that the value of a parameter lies within or on a pair of limit values if set to violate outside the range, or outside a pair of limits if set to violate in range;

b)   declare the check successful when the expression evaluates to true for the correct number of successive valid samples.

3.5.2.5   When performing a constant check, the check service shall:

a)   check that the parameter value evaluates to one of the expected values;

b)   declare the check successful when the expression evaluates to true for the correct number of successive valid samples.

3.5.2.6   When performing a referenced check, the check service shall:

a)   check that the parameter value evaluates to the referenced value;

b)   declare the check successful when the expression evaluates to true for the correct number of successive valid samples.

3.5.2.7   When performing a delta check, the check service shall:

a)   calculate the difference, i.e., the delta value, as either an absolute or percentage depending on the check definition between two parameter values;

b)   check that the difference lies within or on a pair of threshold values if set to violate in range, or outside a pair of thresholds if set to violate outside the range;

c)   declare the check successful when the expression evaluates to true for the correct number of successive valid samples.

3.5.2.8   When performing a compound check, the check service shall:

a)   monitor the check links referenced by the compound check definition and count the number of referenced checks in violation;

b)   declare the compound check violated when the number of referenced checks in violation equals or exceeds the defined count value.

3.5.2.9   The check service shall provide the capability to process several check definitions for the same parameter.

3.5.2.10 Each check definition shall contain:

a)   the identifier of the check definition;

b)   the description of the check definition;

c)   the details specific to that check type.

3.5.2.11 A check definition shall be applied to a specific parameter using a check link.

3.5.2.12 Each check link shall contain:

a)   the identifier of the check definition to use;

b)   the identifier of the parameter to monitor;

c)   a Boolean that controls whether the check is evaluated;

d)   a Boolean that indicates whether the check should be evaluated when the parameter value changes;

e)   a Boolean indicating whether the check should apply to the raw or converted value;

   f) the checking interval;

   g) an optional check validity condition that yielding false prevents the check being performed.

3.5.2.13 Each check validity condition shall contain:

   a) the identifier of a parameter to use as a validity parameter;

   b) an expression operator;

   c) a Boolean indicating whether the expression should apply to the raw or converted value;

   d) a value to evaluate against.

3.5.3 FUNCTIONAL REQUIREMENTS

3.5.3.1 Global check evaluation shall be controlled by the enableService operation.

3.5.3.2 Disabling the check service globally shall pause all check evaluations.

3.5.3.3 Enabling the check service globally shall resume all check evaluations as they were when they were paused.

3.5.3.4 The addParameterCheck operation shall be used to create a link between a check definition and the parameter to be checked.

3.5.3.5 Individual check evaluation shall be controlled by the checkEnabled Boolean value in the CheckLinkDefinition object.

3.5.3.6 Setting the checkEnabled field to TRUE shall enable check evaluation for a specific CheckLink object.

3.5.3.7 The checkEnabled value may be set by using the enableCheck operation.

3.5.3.8 If the checkOnChange value is TRUE, any change in the state (such as validity becoming VALID) or value of a parameter shall cause an evaluation of the parameter value against the check.

3.5.3.9 If the checkOnChange value is TRUE, any change in the state (such as validity becoming VALID) or the value of the parameter referenced by a check condition shall cause an evaluation of the parameter value against the check.

3.5.3.10 If a check is required to be evaluated periodically, then the time between these evaluations shall be controlled using the checkInterval Duration value in the CheckLinkDefinition object.

3.5.3.11 For periodic check evaluations to occur the checkOnChange value must be FALSE.

NOTE – It is not possible to request periodic- and change-based checking concurrently.

3.5.3.12 A checkInterval value of '0' shall mean that no periodic check evaluation shall be performed, and checks will be triggered by another mechanism in this case.

3.5.3.13 The checkInterval value shall only be set when the CheckLinkDefinition object is created.

3.5.3.14 If a checkInterval or maxReportingInterval that is not supported by the provider is requested (both on creation and update of the check), then an INVALID error shall be returned, and the change shall be rejected.

3.5.3.15 The check shall be evaluated if the CheckLink object is enabled, the check condition as defined in the check link evaluates to TRUE, and the checked parameter is in a valid state.

3.5.3.16 The triggerCheck operation shall trigger a check evaluation and immediately publish the result in a CheckTransition event if the check violates.

3.5.3.17 If the latest CheckResult cannot be determined, for example a getSummaryReport operation call for a CheckLink object that has not been evaluated yet, the associated CheckResult shall contain UNCHECKED for the check states and a checked value of NULL.

3.5.3.18 In a CheckDefinitionDetails only one of nominalTime and nominalCount is permitted to be zero, an INVALID error shall be returned from the addCheck or updateDefinition operation if this is not the case.

3.5.3.19 In a CheckDefinitionDetails only one of violationTime and violationCount is permitted to be zero, an INVALID error shall be returned from the addCheck or updateDefinition operation if this is not the case.

3.5.3.20 If nominalTime is not zero, only when the nominalCount number of valid samples including the current one over a nominalTime time period of the checked parameter have passed the check evaluation shall a new check state and check result be calculated.

3.5.3.21 If nominalTime is zero, only when the nominalCount number of successive valid samples including the current one of the checked parameter have passed the check evaluation shall a new check state and check result be calculated.

3.5.3.22 If nominalTime is not zero, but nominalCount is zero, a new check result is always calculated (which then has a new timestamp).

3.5.3.23 If violationTime is not zero, only when the violationCount number of valid samples including the current one over a violationTime time period of the checked parameter have failed the check evaluation shall a new check state and check result be calculated.

3.5.3.24 If violationTime is zero, only when the violationCount number of successive valid samples including the current one of the checked parameter have failed the check evaluation shall a new check state and check result be calculated.

3.5.3.25 If violationTime is not zero, but violationCount is zero, a new check result is always calculated (which then has a new timestamp).

3.5.3.26 For reference and delta checks it shall be possible to define the check to compare against a reference value.

3.5.3.27 The reference value checked against shall be updated once validCount samples of that referenced parameter exist at deltaTime in the past from time now.

3.5.3.28 For a delta check, the delta value shall be calculated as the difference between the reference value and the check parameter and supports both positive and negative deltas.

3.5.3.29 For percentage based delta thresholds, the percentage is calculated as the arithmetic difference between the parameter value and the reference value, as a percentage of the first valid sample where the result of the calculation equal to 1.0 is interpreted as 100 per cent. For example, if the parameter value is 14 and the reference value is 8 then the calculation is (14 - 8) / 8 or 0.75 or 75%.

3.5.3.30 For percentage based delta thresholds, if reference value is zero then a divide by zero condition will occur, implementations shall detect this and the calculated delta will be +/- the maximum Float value.

3.5.3.31 For a constant check, the result of the constant check shall be declared successful only if the parameter value passes evaluation against one of constant values in the definition.

3.5.3.32 For a compound check, if the minimumChecksInViolation field is greater than '0', the result of the compound check shall be declared successful only if the count of referenced CheckLink objects currently in violation is less than the minimumChecksInViolation field.

3.5.3.33 For a compound check, if the minimumChecksInViolation field is set to '0', the result of the compound check shall be declared successful if any of the referenced CheckLink objects is nominal, i.e., not currently in violation.

3.5.3.34 If the maxReportingInterval is not zero and if a CheckResult has not been generated in that time a CheckResult with the current check states shall be generated.

3.5.3.35 The maxReportingInterval value shall be set by using the updateDefinition operation.

3.5.3.36 Each transition in the state or the result of a check shall cause the generation of a CheckResult and a CheckTransition event.

3.5.4 COM USAGE

3.5.4.1 Each check of a provider shall be represented by a CheckIdentity COM object.

3.5.4.2 The body of the CheckIdentity COM object shall hold the name of the check.

3.5.4.3 The check definition details shall be represented as one of the specific check definition COM objects (ConstantCheck, ReferenceCheck, DeltaCheck, LimitCheck, CompoundCheck).

3.5.4.4 The check definition object (ConstantCheck, ReferenceCheck, DeltaCheck, LimitCheck, CompoundCheck) shall use the related link to indicate which CheckIdentity object it uses.

3.5.4.5 The link between which parameters are being evaluated for each check shall be represented as CheckLink COM objects.

3.5.4.6 The CheckLink object shall use the related link to indicate which CheckIdentity object it uses.

3.5.4.7 If the CheckLink belongs to a compound check then the source link shall be set to NULL.

3.5.4.8 If the CheckLink object does not belong to a compound check then the source link shall be used to indicate which ParameterIdentity object it is checking.

3.5.4.9 The definitions of the check links shall be represented as CheckLinkDefinition COM objects.

3.5.4.10 The CheckLinkDefinition object shall use the related link to indicate which CheckLink object it defines.

3.5.4.11 Changes in the state or result of a check shall be represented as CheckTransition COM events.

3.5.4.12 The CheckTransition event objects shall use the related link to indicate which CheckLinkDefinition object it uses.

3.5.4.13 The CheckTransition event objects shall use the source link to indicate what object (if any) caused it to be evaluated, most likely the relevant ParameterValueInstance object or the COM OperationActivity object of a triggerCheck operation.

3.5.4.14 In the case of the maximum reporting interval expiring, the source link of the CheckTransition event object shall be set to NULL.

Table 3-10: Check Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
CheckIdentity	1	MAL::Identifier	Set to NULL	Set to NULL
CheckLink	2	No body	1	Parameter::1
CheckLinkDefinition	3	CheckLinkDetails	2	Set to NULL
ConstantCheck	5	ConstantCheckDefinition	1	Set to NULL
ReferenceCheck	6	ReferenceCheckDefinition	1	Set to NULL
DeltaCheck	7	DeltaCheckDefinition	1	Set to NULL
LimitCheck	8	LimitCheckDefinition	1	Set to NULL
CompoundCheck	9	CompoundCheckDefinition	1	Set to NULL

3.5.5   COM EVENT SERVICE USAGE

3.5.5.1   When a check is evaluated and its state changes, the change in state shall be represented as CheckTransition COM events.

3.5.5.2   When the maximum reporting interval expires the generated check result shall be represented as CheckTransition COM event.

Table 3-11: Check Service Events

Event Name	Object Number	Object Body Type	Related points to	Source points to
CheckTransition	4	CheckResult	3	The object that caused the check evaluation to occur, most likely the relevant ParameterValueInstance object, but also be the COM OperationActivity object of a triggerCheck operation or NULL in the case of the maximum reporting interval expiring.

3.5.6   DISCUSSION—COM OBJECT RELATIONSHIPS

Figure 3-8 below shows the COM object and event relationships for this service.
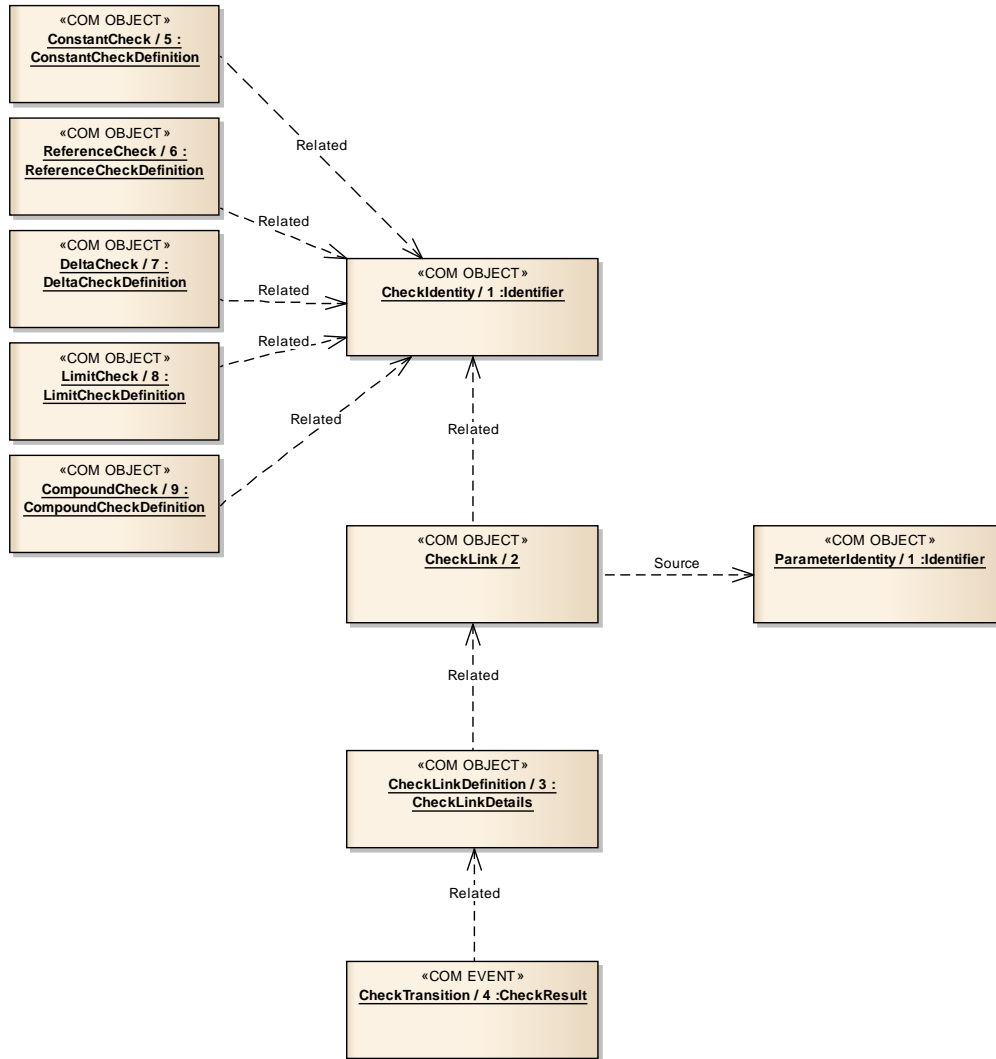
Figure 3-8: Check Service COM Object and Event Relationships

3.5.7   COM ARCHIVE SERVICE USAGE

3.5.7.1   CheckIdentity objects should be stored in the COM archive.

3.5.7.2   The COM objects that hold the check definition details (such as ConstantCheck, ReferenceCheck, DeltaCheck, LimitCheck, CompoundCheck) should be stored in the COM archive.

3.5.7.3   CheckLink objects should be stored in the COM archive.

3.5.7.4   CheckLinkDefinition objects should be stored in the COM archive.

3.5.7.5   When a check transition event is published, the event should be stored in the COM archive.

3.5.8 OPERATION: GETCURRENTTRANSITIONLIST

3.5.8.1 Overview

The getCurrentTransitionList operation allows a consumer to obtain the latest result of a number of checks filtering on check state.

Operation Identifier	getCurrentTransitionList	
Interaction Pattern	PROGRESS	
Pattern Sequence	Message	Body Signature
IN	PROGRESS	filter : (CheckResultFilter)
OUT	ACK	Empty
OUT	UPDATE	updateSummaries : (List<CheckResultSummary>)
OUT	RESPONSE	responseSummaries : (List<CheckResultSummary>)

3.5.8.2 Structures

3.5.8.2.1 The filter field shall contain a set of object instance identifiers for which the check result is required.

3.5.8.2.2 If the checkFilterViaGroups field is TRUE then the checkFilter field shall contain GroupIdentity object instance identifiers; otherwise the field contains CheckIdentity object instance identifiers.

3.5.8.2.3 The CheckIdentity objects referenced, either directly or indirectly via groups, by the checkFilter field shall be the CheckIdentity objects to match.

3.5.8.2.4 The checkFilter field shall support the wildcard value of '0' and shall match all CheckIdentity objects of the provider.

3.5.8.2.5 The service provider shall check for the wildcard value in the list of object instance identifiers in the checkFilter field first and if found no other checks of supplied CheckIdentity object instance identifiers shall be made.

3.5.8.2.6 If the parameterFilterViaGroups field is TRUE then the parameterFilter field shall contain GroupIdentity object instance identifiers; otherwise the field contains ParameterIdentity object instance identifiers.

3.5.8.2.7 The ParameterIdentity objects referenced, either directly or indirectly via groups, by the parameterFilter field shall be the ParameterIdentity objects to match.

3.5.8.2.8 The parameterFilter field shall support the wildcard value of '0' and shall match all ParameterIdentity objects of the provider.

3.5.8.2.9 The service provider shall check for the wildcard value in the list of object instance identifiers in the parameterFilter field first and if found no other checks of supplied ParameterIdentity object instance identifiers shall be made.

3.5.8.2.10 If a referenced GroupIdentity object is unknown then an UNKNOWN error shall be returned.

3.5.8.2.11 If a requested Group, or the Group objects referenced by that Group, does not contain CheckIdentity objects for the checkFilter or ParameterIdentity for the parameterFilter then an INVALID error shall be returned.

3.5.8.2.12 If a referenced CheckIdentity object, either directly or indirectly via groups, is unknown then an UNKNOWN error shall be returned.

3.5.8.2.13 If a referenced ParameterIdentity object, either directly or indirectly via groups, is unknown then an UNKNOWN error shall be returned.

3.5.8.2.14 The filter field shall also contain a list of CheckState enumerations of which states to filter on.

3.5.8.2.15 The supplied lists shall be ANDed together to form the complete filter.

3.5.8.2.16 If a CheckLink object matches the CheckIdentity filter, and the ParameterIdentity filter, and its state matches any of the supplied CheckState enumerations, then its latest CheckResult value shall be returned.

3.5.8.2.17 To report all checks, the wildcard values may be used in the CheckResultFilter.

3.5.8.2.18 If an UNKNOWN or INVALID error is being returned it shall be returned in the ACKNOWLEDGE message and the operation shall end.

3.5.8.2.19 The returned list shall contain an entry for each matched check returning the object instance identifier and the latest CheckResult for that CheckLink object.

3.5.8.2.20 The PROGRESS pattern is used to allow the possibly large list of filtered check results to be split into several updates.

3.5.8.2.21 The size of the lists returned in each update and final response is implementation specific.

3.5.8.3 Errors

The operation may return one of the following errors:

    a) ERROR: INVALID:

        1) one of the referenced groups does not contain the correct type of object;

2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: UNKNOWN:

1) one or more of the checks, parameters, or groups specified in the list do not exist;

2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.5.9   OPERATION: GETSUMMARYREPORT

3.5.9.1   Overview

The getSummaryReport operation allows a consumer to obtain the status of a number of checks and the result of any check evaluations linked to them.

Operation Identifier	getSummaryReport	
Interaction Pattern	PROGRESS	
Pattern Sequence	Message	Body Signature
IN	PROGRESS	objInstIds : (List<MAL::Long>)
OUT	ACK	Empty
OUT	UPDATE	updateObjInstIds : (MAL::Long) updateSummaries : (List<CheckResultSummary>)
OUT	RESPONSE	responseObjInstIds : (MAL::Long) responseSummaries : (List<CheckResultSummary>)

3.5.9.2   Structures

3.5.9.2.1   The objInstIds field shall hold one or more CheckIdentity object instance identifiers of which a check report is required.

3.5.9.2.2   A wildcard value of '0' shall report on all checks.

3.5.9.2.3   The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.5.9.2.4   If a requested check is unknown then an UNKNOWN error shall be returned in the ACKNOWLEDGE message and the operation shall end.

3.5.9.2.5   The returned updates and final response shall contain an entry for each requested CheckIdentity.

3.5.9.2.6   The first part of the update shall be the CheckIdentity object instance identifier.

3.5.9.2.7   The second part shall be the list of all CheckLink object instance identifiers and CheckResults associated with that CheckIdentity.

3.5.9.3   Errors

The operation may return the following error: ERROR: UNKNOWN:

    a) one or more of the checks specified in the list do not exist;

    b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.5.10 OPERATION: ENABLESERVICE

3.5.10.1 Overview

The enableService operation allows a consumer to globally control whether evaluation of all checks is performed or not.

It should be noted that no check reports will be generated if the service provider has been disabled via the enableService operation.

Operation Identifier	enableService	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	enableService : (MAL::Boolean)

3.5.10.2 Structures

3.5.10.2.1 If enableService is set to TRUE the service shall be enabled and evaluation and reporting of check will commence.

3.5.10.2.2 If enableService is set to FALSE then all evaluation of checks shall be suspended and no check transitions will be reported.

3.5.10.2.3 If the enableService value matches the current enabled state of the service then no change shall be made and no error reported. Enabling an already enabled service has no effect.

3.5.10.3 Errors

The operation does not return any errors.

3.5.11 OPERATION: GETSERVICESTATUS

3.5.11.1 Overview

The getServiceStatus operation allows a consumer to determine the global check service enabled status.

Operation Identifier	getServiceStatus	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	Empty
OUT	RESPONSE	serviceEnabled : (MAL::Boolean)

3.5.11.2 Structures

The operation shall return TRUE if the service is currently enabled or FALSE if the service is currently disabled.

3.5.11.3 Errors

The operation does not return any errors.

3.5.12 OPERATION: ENABLECHECK

3.5.12.1 Overview

The enableCheck operation allows a consumer to control whether evaluation of a set of checks is performed or not. The operation allows the consumer to select the checks directly or indirectly using groups.

Operation Identifier	enableCheck	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	isGroupIds : (MAL::Boolean) enableInstances : (List<COM::InstanceBooleanPair>)

3.5.12.2 Structures

3.5.12.2.1 If the isGroupIds field is TRUE then the enableInstances field shall contain GroupIdentity object instance identifiers; otherwise the field contains CheckLink object instance identifiers.

3.5.12.2.2 The CheckLink objects referenced, either directly or indirectly via groups, by the enableInstances field shall be the CheckLink objects to match.

3.5.12.2.3 The id of the enableInstances field shall support the wildcard value of '0' and matches all CheckLink objects of the provider.

3.5.12.2.4 The service provider shall check for the wildcard value in the list of object instance identifiers in the enableInstances field first and if found no other checks of supplied object instance identifiers shall be made.

3.5.12.2.5 If the enableInstances field contains a value of TRUE then evaluations of matching CheckLink objects shall be performed, a value of FALSE requests that evaluations will not be performed.

3.5.12.2.6 No error shall be raised if the enableInstances Boolean value supplied is the same as the current checkEnabled field for a CheckLink object; i.e., enabling an already enabled check will not result in an error.

3.5.12.2.7 If a requested CheckLink or GroupIdentity object is unknown then an UNKNOWN error shall be returned.

3.5.12.2.8 If a requested Group, or the Group objects referenced by that Group, does not contain CheckLink objects then an INVALID error shall be returned.

3.5.12.2.9 If an error is raised then no modifications shall be made as a result of this operation call.

3.5.12.2.10 The provider shall create and store a new CheckLinkDefinition object in the COM archive if the checkEnabled field is changed.

3.5.12.2.11 If the check is being enabled, and the check is defined as being periodic in the check link definition, then the provider shall generate a check result immediately and start the checking interval from that check.

3.5.12.3 Errors

The operation may return one of the following errors:

    a) ERROR: UNKNOWN:

        1) one or more of the requested groups or CheckLink objects is unknown;

        2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

    b) ERROR: INVALID:

        1) one of the supplied groups is not a group of either other group objects or CheckLink objects;

        2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

3.5.13 OPERATION: TRIGGERCHECK

3.5.13.1 Overview

The triggerCheck operation allows a consumer to request the immediate evaluation of a number of checks. Any violations will cause appropriate events to be generated.

It should be noted that no check reports will be generated if the service provider has been disabled via the enableService operation.

Operation Identifier	triggerCheck	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	checkObjInstIds : (List<MAL::Long>) linkObjInstIds : (List<MAL::Long>)

3.5.13.2 Structures

3.5.13.2.1 The checkObjInstIds field shall hold a list of CheckIdentity object instance identifiers to trigger the evaluation of all linked checks.

3.5.13.2.2 The linkObjInstIds field shall hold a list of CheckLink object instance identifiers to trigger the evaluation of.

3.5.13.2.3 The wildcard value of '0' shall be permitted in either list.

3.5.13.2.4 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.5.13.2.5 If a requested CheckIdentity or CheckLink object is unknown then an UNKNOWN error shall be returned.

3.5.13.2.6 If an error is raised then no evaluations shall be made as a result of this operation call.

3.5.13.2.7 Either list may be empty in which case filtering on that aspect, check identity or specific check link, shall be ignored.

3.5.13.2.8 The two lists shall be combined using 'OR' logic, where a CheckLink is evaluated if the identity of a check is in the first list or if the link is directly listed in the second list.

3.5.13.2.9 Triggering a check shall ignore the nominalTime, nominalCount, violationTime and violationCount fields and requests an immediate evaluation of the checks.

3.5.13.2.10 Triggering a check during a periodic check shall not influence the periodic check (e.g., it does not reset the checkInterval timer, the successive valid samples that passed/violated the check or the maxReportingInterval timer).

3.5.13.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

a) one or more of the requested CheckIdentity or CheckLink objects is unknown;

b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.5.14 OPERATION: LISTDEFINITION

3.5.14.1 Overview

The listDefinition operation allows a consumer to request the latest object instance identifiers of the CheckIdentity and actual check definition objects for the supported checks of the provider.

Operation Identifier	listDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	names : (List<MAL::Identifier>)
OUT	RESPONSE	objInstIds : (List<CheckTypedInstance>)

3.5.14.2 Structures

3.5.14.2.1 The names field shall hold a list of CheckIdentity names to retrieve the CheckIdentity and actual check definition object instance identifiers for.

3.5.14.2.2 The request may contain the wildcard value of '*' to return all supported check definitions.

3.5.14.2.3 The wildcard value should be checked for first, if found no other checks of supplied identifiers shall be made.

3.5.14.2.4 If a provided identifier does not include a wildcard and does not match an existing CheckIdentity object then this operation shall fail with an UNKNOWN error.

3.5.14.2.5 The response shall contain a list of matching CheckIdentity and actual check definition object instance identifiers and the actual check definition object type.

3.5.14.2.6 The returned list shall maintain the same order as the submitted list unless the wildcard value was included in the request.

3.5.14.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

    a) one of the supplied identifiers is unknown;

    b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.5.15 OPERATION: LISTCHECKLINKS

3.5.15.1 Overview

The listCheckLinks operation allows a consumer to request the object instance identifiers of the CheckLink objects for the checks of the provider.

Operation Identifier	listCheckLinks	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	checkObjInstIds : (List<MAL::Long>)
OUT	RESPONSE	chkLinkObjInstIds : (List<CheckLinkSummary>)

3.5.15.2 Structures

3.5.15.2.1 The checkObjInstIds field shall hold a list of CheckIdentity object instance identifiers to retrieve the CheckLink object instance identifiers for.

3.5.15.2.2 The request may contain the wildcard value of '0' to return all supported check links.

3.5.15.2.3 The wildcard value should be checked for first, if found no other checks of supplied identifiers shall be made.

3.5.15.2.4 If a provided identifier does not include a wildcard and does not match an existing CheckIdentity object then this operation shall fail with an UNKNOWN error.

3.5.15.2.5 The response shall contain a list of CheckLinkSummary that contain the object instance identifiers of the CheckLink, CheckIdentity, and ParameterIdentity for the matched CheckIdentity objects.

3.5.15.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

    a) one of the supplied identifiers is unknown;

    b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.5.16 OPERATION: ADDCHECK

3.5.16.1 Overview

The addCheck operation allows a consumer to define one or more checks that do not currently exist.

The new CheckIdentity and actual check definition objects are expected to be stored in the COM archive by the provider of the check service.

Operation Identifier	addCheck	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	checkNames : (List<MAL::String>) checkDefDetails : (List<CheckDefinitionDetails>)
OUT	RESPONSE	newObjInstIds : (List<ObjectInstancePair>)

3.5.16.2 Structures

3.5.16.2.1 The checkNames field shall hold the names of the checks to be added.

3.5.16.2.2 The checkNames field must not contain NULL, the wildcard '*', or empty value (an INVALID error shall be returned in this case).

3.5.16.2.3 The supplied names must be unique among all CheckIdentity objects for the domain of the provider; otherwise a DUPLICATE error shall be raised.

3.5.16.2.4 The checkDefDetails field shall hold the CheckDefinitionDetails to be added.

3.5.16.2.5 The two lists shall be ordered the same.

3.5.16.2.6 The number of entries in the two lists shall be the same size; otherwise an INVALID error shall be raised.

3.5.16.2.7 Only one of nominalTime and nominalCount is permitted to be zero, an INVALID error shall be returned if this is not the case.

3.5.16.2.8 Only one of violationTime and violationCount is permitted to be zero, an INVALID error shall be returned if this is not the case.

3.5.16.2.9 If an error is raised then no new identities and definitions shall be added as a result of this operation call.

3.5.16.2.10 If the supplied name matches an existing, but removed, CheckIdentity then that CheckIdentity shall be reused; otherwise a new CheckIdentity shall be created.

3.5.16.2.11 The provider shall create a new actual check definition object and store it, and any new CheckIdentity objects, in the COM archive.

3.5.16.2.12 The response shall contain the list of object instance identifiers for the CheckIdentity and new actual definition objects.

3.5.16.2.13 The returned list shall maintain the same order as the submitted definitions.

3.5.16.3 Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

  1) one of the supplied CheckIdentity objects contains an invalid name or the two lists are not the same size or there is an inconsistency in the time and count fields;

  2) if the two lists are not the same length then the extra information field shall contain the first index of the element in the largest list which does not have corresponding element in the other list;

  3) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: DUPLICATE:

  1) one or more of the CheckIdentity objects being added has supplied a check name that is already in use in the domain;

  2) the extra information field contains a list of the indexes of the erroneous values from the originating request list.

Error	Error #	ExtraInfo Type
DUPLICATE	Defined in COM	List<MAL::UInteger>

3.5.17 OPERATION: UPDATEDEFINITION

3.5.17.1 Overview

The updateDefinition operation allows a consumer to update a definition for one or more checks.

This differs from deleting an existing check and adding a new definition with the same check name in the fact that the CheckIdentity object is not changed between the two definitions.

The replacement definition is expected to be stored in the COM archive by the service provider. The operation does not remove the previous object from the COM archive, merely removes the object from the provider.

The operation also cannot be used to update a check definition for a check that is currently being used, i.e., has CheckLink objects linked to it. The CheckLink objects should first be removed using removeParameterCheck before calling this operation.

Operation Identifier	updateDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	checkInstIds : (List<MAL::Long>) checkDefDetails : (List<CheckDefinitionDetails>)
OUT	RESPONSE	newObjInstIds : (List<MAL::Long>)

3.5.17.2 Structures

3.5.17.2.1 The checkInstIds field shall hold the object instance identifiers of the CheckIdentity objects to be updated.

3.5.17.2.2 If the checkInstIds list contains either NULL or '0' an INVALID error shall be raised.

3.5.17.2.3 The supplied object instance identifiers shall match existing identity objects, an UNKNOWN error shall be raised if this is not the case.

3.5.17.2.4 If the check to be updated is currently being used by a CheckLink object, a REFERENCED error shall be raised.

3.5.17.2.5 The checkDefDetails field shall contain the replacement CheckDefinitionDetails.

3.5.17.2.6 The two lists shall be ordered the same.

3.5.17.2.7 The number of entries in the two lists shall be the same size; otherwise an INVALID error shall be raised.

3.5.17.2.8 Only one of nominalTime and nominalCount is permitted to be zero, an INVALID error shall be returned if this is not the case.

3.5.17.2.9 Only one of violationTime and violationCount is permitted to be zero, an INVALID error shall be returned if this is not the case.

3.5.17.2.10 If an error is raised then no definitions shall be updated as a result of this operation call.

3.5.17.2.11 The provider shall create new actual check definition objects and store them in the COM archive.

3.5.17.2.12 The new definition object shall be the current definition used for the specific CheckIdentity.

3.5.17.2.13 The response shall contain the list of object instance identifiers for the new check definition objects.

3.5.17.2.14 The returned list shall maintain the same order as the submitted definitions.

3.5.17.3 Errors

The operation may return one of the following errors:

    a) ERROR: UNKNOWN:

        1) one of the supplied CheckIdentity object instance identifiers is unknown;

        2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

    b) ERROR: INVALID:

        1) the supplied object instance identifiers list contains either a NULL or '0', or the two lists do not contain the same number of entries or there is an inconsistency in the time and count fields;

        2) if the two lists are not the same length then the extra information field shall contain the first index of the element in the largest list which does not have corresponding element in the other list;

        3) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

c) ERROR: REFERENCED:

1) one of the check objects is currently being used by a CheckLink object;

2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
REFERENCED	70021	List<MAL::UInteger>

3.5.18 OPERATION: REMOVECHECK

3.5.18.1 Overview

The removeCheck operation allows a consumer to remove one or more definitions from the list of checks supported by the check provider.

The operation does not remove the CheckIdentity and actual check definition objects from the COM archive, merely removes the objects from the provider. This permits existing CheckLink objects to continue to reference the correct check object in the COM archive.

Operation Identifier	removeCheck	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	objInstIds : (List<MAL::Long>)

3.5.18.2 Structures

3.5.18.2.1 The objInstIds field holds the object instance identifiers of the CheckIdentity objects to be removed from the provider.

3.5.18.2.2 The list may contain the wildcard value of '0'.

3.5.18.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.5.18.2.4 If a provided CheckIdentity instance identifier does not include a wildcard and does not match an existing check then this operation shall fail with an UNKNOWN error.

3.5.18.2.5 If any of the matched CheckIdentity objects are being referenced by a CheckLink object then a REFERENCED error shall be returned.

3.5.18.2.6 Matched CheckIdentity objects shall not be removed from the COM archive only the list of available CheckIdentity objects in the provider.

3.5.18.2.7 If an error is raised then no CheckIdentity objects shall be removed as a result of this operation call.

3.5.18.2.8 If the operation succeeds then the provider shall not allow new CheckLink objects to be created for the matched CheckIdentity anymore, existing CheckLink objects are not affected.

3.5.18.3 Errors

The operation may return one of the following errors:

a) ERROR: UNKNOWN:

    1) one of the supplied CheckIdentity object instance identifiers is unknown;

    2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

b) ERROR: REFERENCED:

    1) one of the check objects is currently being used by a CheckLink object;

    2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
REFERENCED	70021	List<MAL::UInteger>

3.5.19 OPERATION: ADDPARAMETERCHECK

3.5.19.1 Overview

The addParameterCheck operation allows a consumer to request that one or more parameters/check combinations are added to the list of checks that are being evaluated.

The new CheckLink and CheckLinkDefinition objects are expected to be stored in the COM archive by the provider of the check service.

Operation Identifier	addParameterCheck	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	linkDetails : (List<CheckLinkDetails>) linkRefs : (List<COM::ObjectDetails>)
OUT	RESPONSE	newObjInstIds : (List<ObjectInstancePair>)

3.5.19.2 Structures

3.5.19.2.1 The linkDetails field shall contain the new CheckLink details.

3.5.19.2.2 The linkRefs field shall contain the related and source links of the new CheckLink.

3.5.19.2.3 The related field of the ObjectDetails shall reference the object instance identifier of the CheckIdentity being used by the new CheckLink.

3.5.19.2.4 The source field of the ObjectDetails shall reference the ParameterIdentity that the check is being applied to.

3.5.19.2.5 The two lists must be ordered the same so that the correct ObjectDetails for a specific CheckLink can be determined.

3.5.19.2.6 If the requested CheckIdentity and ParameterIdentity do not exist then an UNKNOWN error shall be returned.

3.5.19.2.7 The number of entries in the two lists shall be the same size; otherwise an INVALID error shall be raised.

3.5.19.2.8 If an interval that is not supported by the provider is requested then an INVALID error shall be returned.

3.5.19.2.9 If the checkInterval is not '0' and the checkOnChange Value is TRUE, then an INVALID error shall be returned.

3.5.19.2.10   The provider shall create new CheckLink and CheckLinkDefinition objects for each pair and store them in the COM archive.

3.5.19.2.11   The response shall contain the list of object instance identifiers for the new CheckLink and CheckLinkDefinition objects.

3.5.19.2.12   The returned list shall maintain the same order as the submitted links.

3.5.19.3 Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

1) the supplied lists do not contain the same number of entries, the supplied interval is not supported by the provider, or a period check with changed based checking has been requested;

2) if the two lists are not the same length then the extra information field shall contain the first index of the element in the largest list which does not have corresponding element in the other list;

3) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: UNKNOWN:

1) one of the supplied object instance identifiers is unknown;

2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.5.20 OPERATION: REMOVEPARAMETERCHECK

3.5.20.1 Overview

The removeParameterCheck operation allows a consumer to remove one or more parameters from the list of parameters being checked by the check provider.

The operation does not remove the CheckLink or CheckLinkDefinition objects from the COM archive, merely removes them from the provider. This permits existing CheckTransition events to continue to reference the correct check link/definition objects in the COM archive.

Operation Identifier	removeParameterCheck	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	objInstIds : (List<MAL::Long>)

3.5.20.2 Structures

3.5.20.2.1 The objInstIds field holds the object instance identifiers of the CheckLink objects to be removed from the provider.

3.5.20.2.2 The list may contain the wildcard value of '0'.

3.5.20.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.5.20.2.4 If a provided CheckLink instance identifier does not include a wildcard and does not match an existing link then this operation shall fail with an UNKNOWN error.

3.5.20.2.5 Matched CheckLink objects shall not be removed from the COM archive only the list of available CheckLink objects in the provider.

3.5.20.2.6 If an error is raised then no CheckLink objects shall be removed as a result of this operation call.

3.5.20.2.7 If the operation succeeds then the provider shall not evaluate those parameter/check definition combinations for the deleted CheckLink objects anymore.

3.5.20.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

a) one of the supplied CheckLink object instance identifiers is unknown;

b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.6   SERVICE: STATISTIC

3.6.1   OVERVIEW

The statistic service allows the consumer to monitor and control the statistical evaluation (e.g., average) of parameters.

The statistic service allows the consumer to link the statistic function to a parameter to be evaluated. Figure 3-9 shows the nominal sequence of operations for the statistic service.
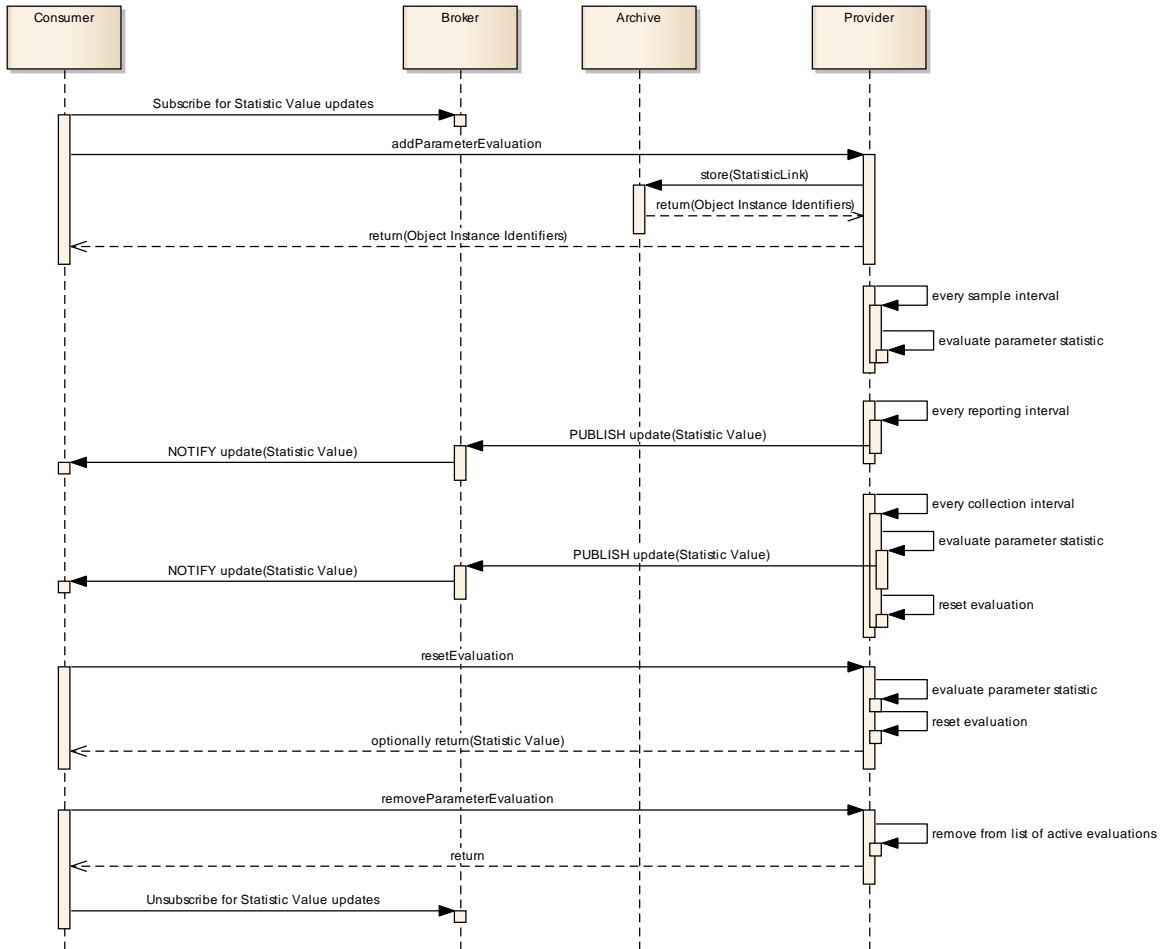
Figure 3-9: Statistic Service Nominal Sequence

For each statistics link the service allows a consumer to specify a sampling interval, a reporting interval, and a collection interval.

The sampling interval defines the interval between samples of the linked parameter, the reporting interval defines the interval between reports of the current statistic evaluation value, and the collection interval defines the period of time which parameter values are collected for the statistic function.

A consumer can create several links to the same parameter with different intervals, for example for the function 'Mean', a consumer might define two links with different collection intervals, one for an hour (D0) and one for four hours (D1). The consumer might also specify the reporting interval of half an hour for the first link and two hours for the second link. This would mean that the service provider would produce reports as shown in figure 3-10.
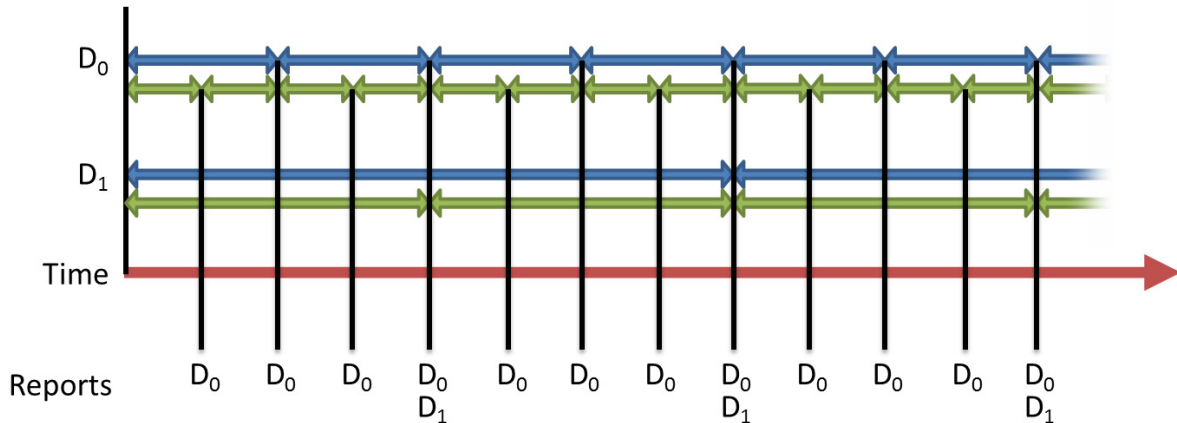
Figure 3-10: Example Statistic Interval Reporting

If a statistic is defined to reset on collection interval expiration the currently calculated value, and any input collection of values, for the parameter being sampled is reset every collection interval, so if an hourly collection interval is defined for the function 'Mean' then every hour the current mean average value is reset (this does not affect the value reported by the Parameter service itself or any other links for the same function and parameter), the set of values used to calculate that mean average is cleared, and a report is generated containing the final value of the function just before the reset.

If a statistic is defined not to reset every collection interval, then the statistic maintains a moving evaluation for the collection interval. For example, if a collection interval of an hour is defined with a reset Boolean of FALSE for the function 'Maximum' then the evaluation will hold the maximum value obtained in the last hour.

The consumer is also able to define the sampling interval of the parameter for a link. This is independent of both the collection interval and the reporting interval. It makes sense for the sampling interval to be smaller than both the collection and reporting interval; however, it is perfectly possible to specify other values, as this is a deployment decision.

For the statistic service, the list of possible statistics functions is deployment-dependent, as any function would have to be implemented in the service provider. There are no operations for the creation, modification, or deletion of the statistic functions. The statistic service defines the evaluation of minimum, maximum, mean, and standard deviation functions; however, the service supports the addition of other statistical functions by implementations.

NOTE –   To ensure deterministic behaviour, implementations should endeavour to respect Nyquist's theorem.

Table 3-12: Statistic Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MC	Statistic	4	5	1
Interaction Pattern	**Operation Identifier**	**Operation Number**	**Support in Replay**	**Capability Set**
REQUEST	getStatistics	1	Yes	1
REQUEST	resetEvaluation	2	No	
PUBLISH-SUBSCRIBE	monitorStatistics	3	Yes	2
SUBMIT	enableService	4	No	3
REQUEST	getServiceStatus	5	No	
SUBMIT	enableReporting	6	No	4
REQUEST	listParameterEvaluations	7	No	5
REQUEST	addParameterEvaluation	8	No	6
REQUEST	updateParameterEvaluation	9	No	
SUBMIT	removeParameterEvaluation	10	No	

3.6.2   HIGH-LEVEL REQUIREMENTS

3.6.2.1   The statistics service may provide the following capabilities:

a) the capability for requesting the current statistics evaluation;

b) the capability for resetting the statistic evaluations;

c) the capability for periodic statistics reporting;

d) the capability to control generation of periodic statistics reporting;

e) the capability for maintaining the list of evaluated parameters.

3.6.2.2   The statistics service may support the evaluation of the following set of statistic functions:

a) the maximum value;

b) the minimum value;

c) the mean value;

d) the standard deviation.

3.6.2.3   Whether the statistics reporting service supports a specific statistic function shall be declared when implementing that service.

3.6.2.4   The statistic service shall maintain a set of links between the statistic function and the parameter that shall be evaluated.

3.6.2.5   Each statistics link shall contain:

a) the identification of the parameter for which statistics are evaluated;

b) the related collection interval;

c) whether the collection resets every collection interval;

d) the related reporting interval;

e) the related sampling interval.

3.6.2.6   The evaluation of a specific statistic shall be reset every collection interval if the reset on collection interval Boolean is TRUE.

3.6.2.7   The parameter referenced by a statistic link shall be sampled every sampling interval.

3.6.2.8   The statistics evaluation value shall be reported every reporting and collection interval if generation is enabled.

3.6.2.9   Each statistics value report shall contain:

a) the start time of the evaluation;

b) the end time of the evaluation;

c) the time of the collected value if applicable;

d) the evaluated value;

e) the number of samples that contributed to the evaluation.

3.6.2.10   When resetting the statistics, the statistics service shall, in sequence:

a) stop the evaluation of the statistics;

b) clear any results accumulated;

c) restart the evaluation process.

3.6.2.11 The statistics service shall provide the capability to enable periodic statistics reporting.

3.6.2.12 The statistics service shall provide the capability to disable the periodic statistics reporting.

3.6.2.13 For each valid instruction to disable periodic statistics reporting, the statistics service shall:

a) set the statistics link status to disabled;

b) continue evaluating the parameter statistics.

3.6.3   FUNCTIONAL REQUIREMENTS

3.6.3.1   The statistic value for a parameter shall be evaluated after each sampling of the parameter and after the collection interval is elapsed.

3.6.3.2   A consumer may request the latest status of the statistical evaluations using the getStatistics operation.

3.6.3.3   If a StatisticLink is required to send periodic reports, then the time between these reports shall be controlled using the reportingInterval duration value.

3.6.3.4   A reportingInterval value of '0' shall mean no periodic reports shall be sent; reports are required to be triggered by another mechanism in this case.

3.6.3.5   The reportingInterval value shall be modified using the updateParameterEvaluation operation.

3.6.3.6   If the reporting interval and the collection intervals align, for example report every minute with a five minute collection interval the fifth minute will align, then only a single report shall be generated in this case.

3.6.3.7   For onboard parameters the link sampling interval should be a multiple of the minimum sampling interval of those parameters.

3.6.3.8   If an interval that is not supported by the provider is requested, then an INVALID error shall be returned, and the change rejected.

3.6.3.9   For the 'Maximum' statistics function, the highest value shall be reported, in the case of multiple samples matching that, the first occurrence shall be report.

3.6.3.10 For the 'Minimum' statistics function, the smallest value shall be reported, in the case of multiple samples matching that, the first occurrence shall be report.

3.6.3.11 For the 'Mean average' statistics function, when used with integer parameters the result shall be reported using a Double type.

3.6.3.12 For the 'Standard deviation' statistics function, when used with integer parameters the result shall be reported using a Double type.

3.6.4   COM USAGE

3.6.4.1   The statistic functions shall be represented as StatisticFunction COM objects.

3.6.4.2   For a 'Maximum' statistics function, the object instance identifier of '1' and name of 'MAX' shall be used.

3.6.4.3   For a 'Minimum' statistics function, the object instance identifier of '2' and name of 'MIN' shall be used.

3.6.4.4   For a 'Mean average' statistics function, the object instance identifier of '3' and name of 'MEAN' shall be used.

3.6.4.5   For a 'Standard deviation' statistics function, the object instance identifier of '4' and name of 'SD' shall be used.

3.6.4.6   For non-standard statistics functions, i.e., ones not listed here, the name and object instance identifier shall be communicated through an out-of-band agreement.

3.6.4.7   The link between which parameters are being evaluated for each function shall be represented as StatisticLink COM objects.

3.6.4.8   The details of a statistic link, such as reportingInterval, shall be represented as a StatisticLinkDefinition COM object.

3.6.4.9   Statistic evaluations for a specific parameter and statistic function shall be represented as StatisticValueInstance COM objects.

3.6.4.10   The StatisticLink object shall use the related link to indicate which StatisticFunction object it uses.

3.6.4.11   The StatisticLink object shall use the source link to indicate which ParameterIdentity object it uses.

3.6.4.12   The StatisticLinkDefinition object shall use the related link to indicate which StatisticLink object it uses.

3.6.4.13   The StatisticValueInstance object shall use the related link to indicate which StatisticLinkDefinition object it uses.

3.6.4.14   The StatisticValueInstance object shall use the source link to indicate what object (if any) caused it to be evaluated.

3.6.4.15   If the StatisticValueInstance object was a result of a getStatistics operation call then the source link shall be the COM OperationActivity object of the operation.

Table 3-13: Statistic Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
StatisticFunction	1	StatisticFunctionDetails	Set to NULL	Set to NULL
StatisticLink	2	No body	1	Parameter::1
StatisticLinkDefinition	3	StatisticLinkDetails	2	Set to NULL
StatisticValueInstance	4	StatisticValue	3	The object that caused the statistical evaluation to occur may be NULL if it occurred as a result of evaluation period timeout.

3.6.5   DISCUSSION—COM OBJECT RELATIONSHIPS

Figure 3-11 below shows the COM object relationships for this service.
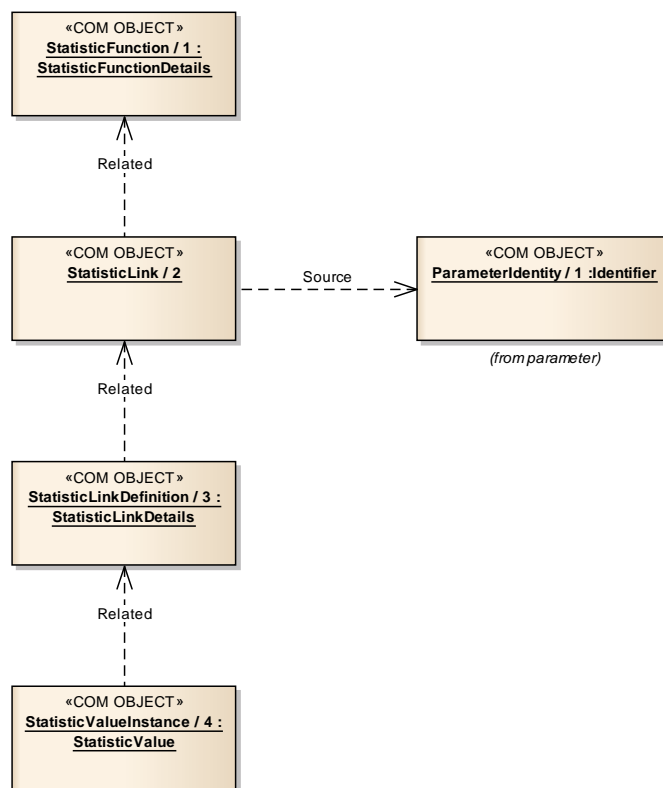
Figure 3-11: Statistic Service COM Object Relationships

3.6.6   COM ARCHIVE SERVICE USAGE

3.6.6.1   StatisticFunction objects should be stored in the COM archive only if they are not one of the standard functions.

3.6.6.2   StatisticLink objects should be stored in the COM archive.

3.6.6.3   StatisticLinkDefinition objects should be stored in the COM archive.

3.6.6.4   When a StatisticValueInstance report is published, the object should be stored in the COM archive.

3.6.7 OPERATION: GETSTATISTICS

3.6.7.1 Overview

The getStatistics operation returns the latest value for a set of existing statistic evaluations.

Operation Identifier	getStatistics	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	funcObjInstIds : (List<MAL::Long>) isGroup : (MAL::Boolean) paramObjInstIds : (List<COM::ObjectKey>)
OUT	RESPONSE	evaluations : (List<StatisticEvaluationReport>)

3.6.7.2 Structures

3.6.7.2.1 The funcObjInstIds field shall include a list of StatisticFunction object instance identifiers to match.

3.6.7.2.2 The funcObjInstIds field shall support the wildcard value of '0' and will match all functions of the provider.

3.6.7.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.6.7.2.4 If the isGroup field is TRUE then the paramObjInstIds field shall contain GroupIdentity object instance identifiers; otherwise the field shall contain ParameterIdentity object instance identifiers.

3.6.7.2.5 If the isGroup field is TRUE, the requested Group, or the Group objects referenced by that Group, must contain ParameterIdentity objects; otherwise an INVALID error shall be returned.

3.6.7.2.6 The ParameterIdentity objects referenced, either directly or indirectly via groups, by the paramObjInstIds field shall be the parameters to match.

3.6.7.2.7 The paramObjInstIds field shall support the wildcard value of '0' and matches all parameters of the provider matched to the functions given in the funcObjInstIds field.

3.6.7.2.8 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.6.7.2.9 If a requested function, group, or parameters are unknown then an UNKNOWN error shall be returned.

3.6.7.2.10 The sets of matched StatisticFunction objects and ParameterIdentity objects shall be matched to the set of existing StatisticLink objects to determine which StatisticLink objects to report on.

3.6.7.2.11 The response shall contain a list of matching statistics evaluation values.

3.6.7.2.12 The operation shall trigger an evaluation of the statistical functions matched and return the new evaluation values.

3.6.7.2.13 If it is not possible to return an evaluation value for a matched evaluation (for example not enough samples available) then no entry for that evaluation shall be included.

3.6.7.2.14 The evaluation shall not trigger a report via the monitorStatistics operation.

3.6.7.2.15 Requesting an evaluation shall ignore the samplingInterval, reportingInterval, and collectionInterval fields and requests an immediate evaluation of the statistic.

3.6.7.2.16 Requesting an evaluation during a periodic evaluation shall not influence the periodic evaluation (e.g., it does not reset the samplingInterval, reportingInterval, and collectionInterval timers or the current periodic collection value).

3.6.7.3 Errors

The operation may return one of the following errors:

a) ERROR: UNKNOWN:

 1) one or more of the requested groups or parameters do not exist in the provider or statistic functions is not supported by the provider;

 2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

b) ERROR: INVALID:

 1) one of the supplied groups is not a group of groups or ParameterIdentity objects;

 2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

3.6.8   OPERATION: RESETEVALUATION

3.6.8.1   Overview

The operation allows a consumer to reset the statistical evaluations so the evaluations restart from the current time (without changing the collection interval), optionally returning the evaluation up to that point. Resetting the evaluation will affect all consumers.

Operation Identifier	resetEvaluation	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	isStatLinkGroup : (MAL::Boolean) objInstIds : (List<MAL::Long>) returnLatestEval : (MAL::Boolean)
OUT	RESPONSE	evaluations : (List<StatisticEvaluationReport>)

3.6.8.2   Structures

3.6.8.2.1   If the isStatLinkGroup field is TRUE then the objInstIds field shall contain GroupIdentity object instance identifiers; otherwise the field shall contain StatisticFunction object instance identifiers.

3.6.8.2.2   If the isStatLinkGroup field is TRUE, the requested Group, or the Group objects referenced by that Group, must contain StatisticLink objects; otherwise an INVALID error shall be returned.

3.6.8.2.3   The StatisticLink objects referenced, either indirectly via statistic functions or indirectly via groups, by the objInstIds field shall be the StatisticLink objects to match.

3.6.8.2.4   The objInstIds field shall support the wildcard value of '0' and matches all StatisticLink objects of the provider.

3.6.8.2.5   The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.6.8.2.6   If a requested function or group is unknown then an UNKNOWN error shall be returned.

3.6.8.2.7   If the returnLatestEval Boolean field is TRUE then the latest evaluation result for each of the matched links shall be returned before resetting; otherwise a NULL is returned.

3.6.8.2.8   If an error is raised then no resetting of evaluations shall be made as a result of this operation call.

3.6.8.3   Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

1) one of the supplied groups is not a group of groups or StatisticLink objects;

2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: UNKNOWN:

1) one or more of the requested groups or functions is unknown;

2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.6.9   OPERATION: MONITORSTATISTICS

3.6.9.1   Overview

The monitorStatistics operation allows a consumer to subscribe for statistical evaluation value reports.

It should be noted that no evaluation reports will be generated if the service provider has been disabled via the enableService operation.

Operation Identifier	monitorStatistics	
Interaction Pattern	PUBLISH-SUBSCRIBE	
Pattern Sequence	Message	Body Signature
OUT	PUBLISH/NOTIFY	relatedId : (MAL::Long) sourceId : (COM::ObjectId) statisticValue : (StatisticValue)

3.6.9.2   Structures

3.6.9.2.1   The MAL EntityKey.firstSubKey shall contain the statistic function name.

3.6.9.2.2   The MAL EntityKey.secondSubKey shall contain the StatisticLink object instance identifier.

3.6.9.2.3   The MAL EntityKey.thirdSubKey shall contain the ParameterIdentity object instance identifier.

3.6.9.2.4   The MAL EntityKey.fourthSubKey shall contain the new StatisticValueInstance object instance identifier.

3.6.9.2.5   The timestamp of the StatisticValueInstance report shall be taken from the publish message.

3.6.9.2.6   The related link of the update shall be held in the relatedId field.

3.6.9.2.7   The source link of the StatisticValueInstance shall be held in the sourceId field.

3.6.9.2.8   If no source link is needed then the sourceId shall be set to NULL.

3.6.9.2.9   The second part of the publish message shall be the StatisticValueInstance object value.

3.6.9.3   Errors

The operation does not return any errors.

3.6.10 OPERATION: ENABLESERVICE

3.6.10.1 Overview

The enableService operation allows a consumer to globally control whether evaluation of all statistics is performed or not.

Operation Identifier	enableService	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	enableService : (MAL::Boolean)

3.6.10.2 Structures

3.6.10.2.1 If enableService is set to TRUE the service shall be enabled and evaluation and reporting of statistics will be reset and commence.

3.6.10.2.2 If enableService is set to FALSE then all evaluation of statistics shall be suspended and no statistics will be reported.

3.6.10.2.3 If the enableService value matches the current enabled state of the service then no change shall be made and no error reported. Enabling an already enabled service has no effect.

3.6.10.3 Errors

The operation does not return any errors.

3.6.11 OPERATION: GETSERVICESTATUS

3.6.11.1 Overview

The getServiceStatus operation allows a consumer to determine the global statistic service enabled status.

Operation Identifier	getServiceStatus	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	Empty
OUT	RESPONSE	serviceEnabled : (MAL::Boolean)

3.6.11.2 Structures

The operation shall return TRUE if the service is currently enabled or FALSE if the service is currently disabled.

3.6.11.3 Errors

The operation does not return any errors.

3.6.12 OPERATION: ENABLEREPORTING

3.6.12.1 Overview

The enableReporting operation allows a consumer to control whether reports for specific statistical functions are generated or not. The operation allows the consumer to select the functions directly or indirectly using groups.

Operation Identifier	enableReporting	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	isGroupIds : (MAL::Boolean) enableInstances : (List<COM::InstanceBooleanPair>)

3.6.12.2 Structures

3.6.12.2.1 If the isGroupIds field is TRUE then the enableInstances field shall contain GroupIdentity object instance identifiers; otherwise the field contains StatisticFunction object instance identifiers.

3.6.12.2.2 If the isGroupIds field is TRUE, the requested Group, or the Group objects referenced by that Group, must contain StatisticLink objects; otherwise an INVALID error shall be returned.

3.6.12.2.3 The StatisticLink objects referenced, either indirectly via StatisticFunction objects or indirectly via groups, by the enableInstances field shall be the StatisticLink objects to match.

3.6.12.2.4 The id of the enableInstances field shall support the wildcard value of '0' and matches all StatisticLink objects of the provider.

3.6.12.2.5 The service provider shall check for the wildcard value in the list of object instance identifiers in the enableInstances field first and if found no other checks of supplied object instance identifiers shall be made.

3.6.12.2.6 If the enableInstances field contains a value of TRUE then reports after the reporting and collection intervals for matching StatisticLink objects shall be generated, a value of FALSE requests that reports will not be generated.

3.6.12.2.7 No error shall be raised if the enableInstances Boolean value supplied is the same as the current reportingEnabled field for a StatisticLink object; i.e., enabling an already enabled link will not result in an error.

3.6.12.2.8 If a requested StatisticFunction or GroupIdentity object is unknown then an UNKNOWN error shall be returned.

3.6.12.2.9 If an error is raised then no modifications shall be made as a result of this operation call.

3.6.12.2.10 The provider should create and store a new StatisticLinkDefinition object in the COM archive if the reportingEnabled field is changed.

3.6.12.2.11 If the generation of reports is being enabled, then the provider shall generate a report immediately and start the report interval from that report.

3.6.12.3 Errors

The operation may return one of the following errors:

a) ERROR: UNKNOWN:

   1) one or more of the requested StatisticFunction or Group objects is unknown;

   2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

b) ERROR: INVALID:

   1) one of the supplied groups is not a group of groups or StatisticLink objects;

   2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

3.6.13 OPERATION: LISTPARAMETEREVALUATIONS

3.6.13.1 Overview

The listParameterEvaluations operation allows a consumer to request the object instance identifiers of the StatisticLink objects for the evaluations of the provider.

Operation Identifier	listParameterEvaluations	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	statObjInstIds : (List<MAL::Long>)
OUT	RESPONSE	statLinkObjInstIds : (List<StatisticLinkSummary>)

3.6.13.2 Structures

3.6.13.2.1 The statObjInstIds field shall hold a list of StatisticFunction object instance identifiers to retrieve the StatisticLink object instance identifiers for.

3.6.13.2.2 The request may contain the wildcard value of '0' to return all supported statistic links.

3.6.13.2.3 The wildcard value should be checked for first, if found no other checks of supplied identifiers shall be made.

3.6.13.2.4 If a provided identifier does not include a wildcard and does not match an existing StatisticFunction object then this operation shall fail with an UNKNOWN error.

3.6.13.2.5 The response shall contain a list of StatisticLinkSummary that contain the object instance identifiers of the StatisticLink, StatisticFunction, and ParameterIdentity for the matched StatisticFunction objects.

3.6.13.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

   a) one of the supplied identifiers is unknown;

   b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.6.14 OPERATION: ADDPARAMETEREVALUATION

3.6.14.1 Overview

The addParameterEvaluation operation allows a consumer to request that one or more parameters/function combinations are added to the list of parameters that are being evaluated.

The new StatisticLink and StatisticLinkDefinition objects are expected to be stored in the COM archive by the provider of the statistic service.

Operation Identifier	addParameterEvaluation	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	newDetails : (List<StatisticCreationRequest>)
OUT	RESPONSE	newObjInstIds : (List<ObjectInstancePair>)

3.6.14.2 Structures

3.6.14.2.1 The newDetails field shall hold a StatisticCreationRequest for each new parameter to be sampled.

3.6.14.2.2 The statFuncInstId field of the StatisticCreationRequest shall reference the object instance identifier of the StatisticFunction to be used.

3.6.14.2.3 If the statFuncInstId field does not match an existing StatisticFunction then an UNKNOWN error shall be raised.

3.6.14.2.4 The parameterId shall reference the ParameterIdentity that the function is being applied to.

3.6.14.2.5 If the parameterId field does not match an existing ParameterIdentity then an UNKNOWN error shall be raised.

3.6.14.2.6 If the type of the matched parameter is not supported by the matched statistical function, for example Mean average of a String parameter, then an INVALID error shall be returned.

3.6.14.2.7 The samplingInterval field shall contain the sampling duration interval for the parameter.

3.6.14.2.8 If the supplied samplingInterval is not supported for the requested parameter then an INVALID error shall be returned.

3.6.14.2.9 If an error is raised then no new StatisticLink object shall be created and stored as a result of this operation call.

3.6.14.2.10 If no error is to be raised then StatisticLink and StatisticLinkDefinition objects shall be created for each function/parameter link and stored in the COM archive.

3.6.14.2.11 The referenced parameter shall be sampled immediately and the sampling, reporting, and collection intervals started.

3.6.14.2.12 The response shall contain the list of object instance identifiers for the new StatisticLink and StatisticLinkDefinition objects.

3.6.14.2.13 The object instance identifiers of the StatisticLink and StatisticLinkDefinition objects shall be held in the first and second fields of the ObjectInstancePair structure respectively.

3.6.14.2.14 The returned list shall maintain the same order as the submitted links.

3.6.14.3 Errors

The operation may return one of the following errors:

a) ERROR: INVALID:

    1) one or more of the supplied StatisticLink is either requesting an invalid sampling interval or invalid function for the request parameter;

    2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

b) ERROR: UNKNOWN:

    1) one of the requested StatisticLink objects references either an unknown StatisticFunction object or an unknown ParameterIdentity object;

    2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.6.15 OPERATION: UPDATEPARAMETEREVALUATION

3.6.15.1 Overview

The updateParameterEvaluation operation allows a consumer to modify the intervals, reporting, and reset Booleans for one or more statistical evaluation links.

The replacement StatisticLinkDefinition objects are expected to be stored in the COM archive by the service provider. The operation does not remove the previous object from the COM archive, merely removes the object from the provider.

Operation Identifier	updateParameterEvaluation	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	linkIds : (List<MAL::Long>) newDetails : (List<StatisticLinkDetails>)
OUT	RESPONSE	newLinkDefIds : (List<MAL::Long>)

3.6.15.2 Structures

3.6.15.2.1 The linkIds field shall contain the object instance identifiers of the StatisticLink objects to be updated.

3.6.15.2.2 If the linkIds list contains either NULL or '0' an INVALID error shall be raised.

3.6.15.2.3 The supplied object instance identifiers shall match existing link objects, an UNKNOWN error shall be raised if this is not the case.

3.6.15.2.4 If the supplied samplingInterval is not supported for the requested parameter then an INVALID error shall be returned.

3.6.15.2.5 The newDetails field shall contain the replacement StatisticLinkDetails.

3.6.15.2.6 The two lists shall be ordered the same.

3.6.15.2.7 The number of entries in the two lists shall be the same size; otherwise an INVALID error shall be returned.

3.6.15.2.8 If an error is raised then no links shall be updated as a result of this operation call.

3.6.15.2.9 The provider shall create a new StatisticLinkDefinition object and store it in the COM archive.

3.6.15.2.10 If any of the intervals are updated then the service shall reset the relevant timer and use the new intervals immediately.

3.6.15.2.11 The response shall contain the list of object instance identifiers for the new StatisticLinkDefinition objects.

3.6.15.2.12 The returned list shall maintain the same order as the submitted links.

3.6.15.3 Errors

The operation may return one of the following errors:

a) ERROR: UNKNOWN:

   1) one or more of the supplied StatisticLink object instance identifiers is unknown;

   2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

b) ERROR: INVALID:

   1) one or more of the supplied object instance identifiers list contains either a NULL or '0' or is requesting an invalid sampling interval for the request parameter or the two supplied lists are not the same length;

   2) if the two lists are not the same length then the extra information field shall contain the first index of the element in the largest list which does not have corresponding element in the other list;

   3) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

3.6.16 OPERATION: REMOVEPARAMETEREVALUATION

3.6.16.1 Overview

The removeParameterEvaluation operation allows a consumer to remove one or more parameters from the list of parameters being sampled by the statistic provider.

The operation does not remove the StatisticLink or StatisticLinkDefinition objects from the COM archive, merely removes them from the provider. This permits existing evaluation results to continue to reference the correct StatisticLink and StatisticLinkDefinition objects in the COM archive.

Operation Identifier	removeParameterEvaluation	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	objInstIds : (List<MAL::Long>)

3.6.16.2 Structures

3.6.16.2.1 The objInstIds field holds the object instance identifiers of the StatisticLink objects to be removed from the provider.

3.6.16.2.2 The list may contain the wildcard value of '0'.

3.6.16.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.6.16.2.4 If a provided StatisticLink object instance identifier does not include a wildcard and does not match an existing StatisticLink object then this operation shall fail with an UNKNOWN error.

3.6.16.2.5 Matched StatisticLink objects shall not be removed from the COM archive only the list of evaluated StatisticLink objects in the provider.

3.6.16.2.6 If an error is raised then no StatisticLink objects shall be removed as a result of this operation call.

3.6.16.2.7 If the operation succeeds then the provider shall not evaluate those parameter/function definition combinations for the deleted StatisticLink objects anymore.

3.6.16.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

- a) one or more of the supplied StatisticLink object instance identifiers is unknown;
- b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.7 SERVICE: AGGREGATION

3.7.1 OVERVIEW

The aggregation service allows the user to acquire several parameter values in a single request.

Aggregations are generated either periodically, on an ad-hoc basis, or after a configurable timeout when being filtered. Periodic is where they are generated at a specific generation interval; in this case the periodicity is an aspect of the aggregation rather than the contained parameters. Ad-hoc is when the aggregation generation is triggered by some other deployment-specific mechanism.

Filtering is where an aggregation is only generated when the change in the value of the filtered parameters exceeds a specified threshold or a timeout is passed. Filtering can be applied to both periodic and ad-hoc aggregations.

As there may be a large amount of time between reports of an aggregation when filtering is applied, the filtering concept also has a maximum reporting interval. If a report of the aggregation has not passed the filter in the maximum reporting interval a report is generated regardless. This allows there to be regular reports of an aggregation sent regardless of filtering.

It should be noted that applying a filter with a maximum reporting interval to an ad-hoc aggregation will cause the aggregation to generate reports in a periodic way if the maximum reporting interval is left to expire. This is expected behaviour.

There are a number of intervals defined in the structures of the service; the diagram in figure 3-12 illustrates the use of these.
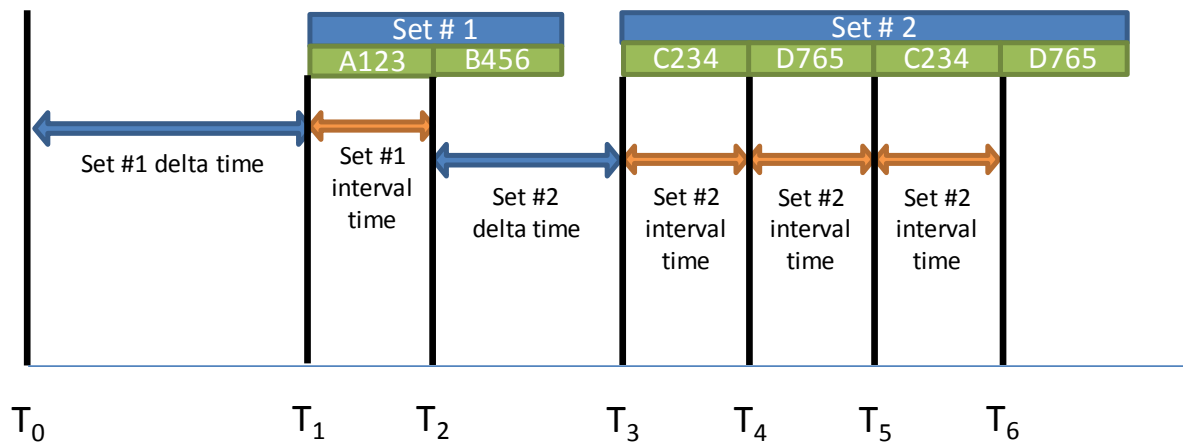
Figure 3-12: Aggregation Delta Time Calculation

The diagram shows a single report of an aggregation being generated which contains two parameter sets. Each set of parameter values in the aggregation report contains two optional durations, delta time and interval time.

– The timestamp of the first value of the first set (T1) is defined as the timestamp of the aggregation report plus the delta time of the first set.

– The timestamp of the second value of the first set (T2) is defined as the timestamp of the first parameter (T1) plus the interval time of the set.

– The timestamp of the first value of the second set (T3) is defined as the timestamp of the last value of the previous set (T2) plus the delta time of the second set.

– The timestamp of the second value of the second set (T4) is defined as the timestamp of the first parameter (T3) plus the interval time of the set.

The COM archive is used to hold the definitions of the aggregations.

Table 3-14: Aggregation Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MC	Aggregation	4	6	1
Interaction Pattern	Operation Identifier	Operation Number	Support in Replay	Capability Set
PUBLISH-SUBSCRIBE	monitorValue	1	Yes	1
REQUEST	getValue	2	Yes	2
SUBMIT	enableGeneration	3	No	3
SUBMIT	enableFilter	4	No	
REQUEST	listDefinition	5	Yes	4
REQUEST	addAggregation	6	No	5
REQUEST	updateDefinition	7	No	
SUBMIT	removeAggregation	8	No	

3.7.2   HIGH-LEVEL REQUIREMENTS

3.7.2.1   The aggregation service shall provide:

a) the capability for periodic reporting of predefined aggregation values;

b) the capability for requesting the current value of a set of aggregations;

c) the capability for controlling reporting of the aggregation values;

d) the capability for maintaining the list of aggregation definitions.

3.7.2.2   Three modes of generating an aggregation value shall exist: periodic, ad-hoc, and filtered:

3.7.2.3   Periodic reports for aggregations shall be generated once for each cycle of a specified interval.

a) Ad-hoc reports for aggregations shall be triggered by some other deployment-specific means.

b) Filtered reports for aggregations shall be generated only when the value of one or more parameters in the aggregation has changed by more than a given threshold since the previous generation or when a timeout expires.

3.7.3   FUNCTIONAL REQUIREMENTS

3.7.3.1   The generation of reports for an aggregation shall be controlled by the generationEnabled Boolean value in the AggregationDefinitionDetails structure.

3.7.3.2   Setting the generationEnabled field to TRUE shall enable generation of reports for a specific aggregation.

3.7.3.3   The generationEnabled values can be set by using the enableGeneration operation.

3.7.3.4   The filtering of reports for an aggregation shall be controlled by the filterEnabled Boolean value in the AggregationDefinitionDetails structure.

3.7.3.5   Setting the filterEnabled field to TRUE shall enable filtering of reports for a specific aggregation.

3.7.3.6   The filterEnabled values can be set by using the enableFilter operation.

3.7.3.7   If an aggregation is required to send periodic reports, then the time between these reports shall be controlled using the reportInterval Duration value in the AggregationDefinitionDetails structure.

3.7.3.8   Periodic reports for aggregations shall set the generationMode field of the AggregationValueInstance to PERIODIC.

3.7.3.9   No periodic reports shall be generated when the reportInterval field is set to '0' and reports will be triggered by another ad-hoc implementation-specific mechanism in this case.

3.7.3.10 Ad-hoc reports for aggregations shall set the generationMode field of the AggregationValueInstance to ADHOC.

3.7.3.11 If an aggregation is filtered, reports shall be generated only when the value of one or more parameters in the aggregation have changed by more than a given threshold since the previous generated aggregation report.

3.7.3.12 The threshold for the filter shall be defined, on a per-parameter basis, either as a percentage or an (absolute) delta change in value.

3.7.3.13 Parameters in an aggregation that do not have a filter specified in the aggregation definition shall be ignored by the filter comparison process.

3.7.3.14 If filtering is enabled and a filtered timeout has been set for that aggregation, if no change has been detected in that time-out period, an aggregation value shall be generated with a generation mode of FILTERED_TIMEOUT. This measure ensures that extensive periods of time do not elapse without reports of an enabled aggregation.

3.7.3.15 The generationEnabled, filterEnabled, filteredTimeout, sendUnchanged, and reportInterval values of an AggregationDefinition object can be set by using the updateDefinition operation.

3.7.3.16 If the filteredTimeout interval, reportInterval or one or more of the sampleIntervals requested are not supported by the provider then an INVALID error shall be returned and the creation or change rejected.

3.7.3.17 If the sendUnchanged field in the AggregationDefinition object is FALSE then parameter values that have not changed since the previous report shall be replaced by a NULL in the values list of the AggregationSetValue composite.

3.7.3.18 The ordering of the values in the AggregationValueInstance object shall match the ordering of parameters in the AggregationDefinitionDetails structure.

3.7.4   COM USAGE

3.7.4.1   Each aggregation of a provider shall be represented by an AggregationIdentity COM object.

3.7.4.2   The body of the AggregationIdentity COM object shall hold the name of the aggregation.

3.7.4.3   The definitions of the aggregations shall be represented as AggregationDefinition COM objects.

3.7.4.4   Aggregation value reports shall be represented as AggregationValueInstance COM objects.

3.7.4.5   The AggregationDefinition object shall use the related link to indicate which AggregationIdentity object it uses.

3.7.4.6   The AggregationValueInstance object shall use the related link to indicate which AggregationDefinition object it uses.

3.7.4.7   The source link of the AggregationIdentity object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.7.4.8   The source link of the AggregationDefinition object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.7.4.9   The source link of the AggregationValueInstance object should be the object that caused the report to be generated.

3.7.4.10 The source link of the AggregationValueInstance object shall be NULL for periodic reports.

Table 3-15: Aggregation Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
AggregationIdentity	1	MAL::Identifier	Set to NULL	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
AggregationDefinition	2	AggregationDefinitionDetails	1	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
AggregationValueInstance	3	AggregationValue	2	The object that caused the value to be generated or NULL for periodic reports.

3.7.5   DISCUSSION—COM OBJECT RELATIONSHIPS

Figure 3-13 below shows the COM object relationships for this service.
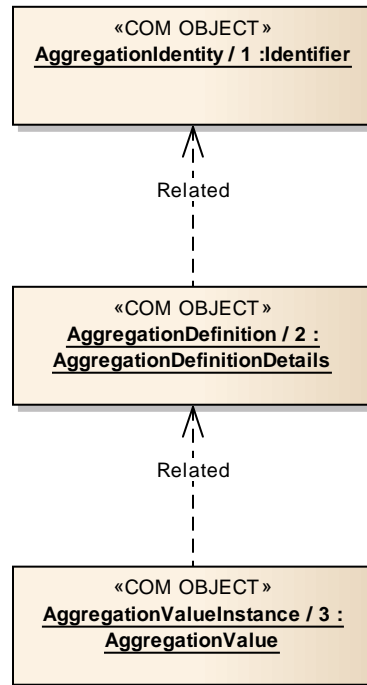
Figure 3-13: Aggregation Service COM Object Relationships

3.7.6   COM ARCHIVE SERVICE USAGE

3.7.6.1   AggregationIdentity and AggregationDefinition objects should be stored in the COM archive.

3.7.6.2   When an aggregation value report is published, the AggregationValueInstance object should be stored in the COM archive.

3.7.7   OPERATION: MONITORVALUE

3.7.7.1   Overview

The monitorValue operation allows a consumer to subscribe for aggregation value reports.

Operation Identifier	monitorValue	
Interaction Pattern	PUBLISH-SUBSCRIBE	
Pattern Sequence	Message	Body Signature
OUT	PUBLISH/NOTIFY	objId : (COM::ObjectId) newValue : (AggregationValue)

3.7.7.2   Structures

3.7.7.2.1   The MAL EntityKey.firstSubKey shall contain the aggregation name.

3.7.7.2.2   The MAL EntityKey.secondSubKey shall contain the AggregationIdentity object instance identifier.

3.7.7.2.3   The MAL EntityKey.thirdSubKey shall contain the AggregationDefinition object instance identifier.

3.7.7.2.4   The MAL EntityKey.fourthSubKey shall contain the new AggregationValueInstance object instance identifier.

3.7.7.2.5   The timestamp of the AggregationValueInstance report shall be taken from the publish message.

3.7.7.2.6   The publish message shall include the ObjectId of the source link of the report.

3.7.7.2.7   If no source link is needed then the ObjectId shall be replaced with a NULL.

3.7.7.2.8   The second part of the publish message shall be the AggregationValue.

3.7.7.3   Errors

The operation does not return any errors.

3.7.8 OPERATION: GETVALUE

3.7.8.1 Overview

The getValue operation returns the latest received value for a requested aggregation.

Operation Identifier	getValue	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	aggInstIds : (List<MAL::Long>)
OUT	RESPONSE	aggValDetails : (List<AggregationValueDetails>)

3.7.8.2 Structures

3.7.8.2.1 The aggInstIds field shall provide the list of AggregationIdentity object instance identifiers.

3.7.8.2.2 The wildcard value of '0' shall be supported and matches all aggregations of the provider.

3.7.8.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.7.8.2.4 If a requested aggregation is unknown then an UNKNOWN error shall be returned.

3.7.8.2.5 The filter shall not be applied for the getValue operation.

3.7.8.2.6 If an aggregation is being reported periodically, using the operation shall not reset the reportInterval or filteredTimeout timer.

3.7.8.2.7 The response shall contain a list of returned AggregationIdentity and AggregationDefinition object instance identifier pairs and a matching list of AggregationValues.

3.7.8.2.8 The new value shall not be published via the monitorValue operation.

3.7.8.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

    a) one or more of the requested aggregations is unknown;

    b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.7.9   OPERATION: ENABLEGENERATION

3.7.9.1   Overview

The enableGeneration operation allows a consumer to control whether reports for specific aggregations are generated or not. The operation allows the consumer to select the aggregations directly or indirectly using groups. This affects all types of aggregations, periodic, filtered, and ad-hoc.

Operation Identifier	enableGeneration	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	isGroupIds : (MAL::Boolean) enableInstances : (List<COM::InstanceBooleanPair>)

3.7.9.2   Structures

3.7.9.2.1   If the isGroupIds field is TRUE then the enableInstances field shall contain GroupIdentity object instance identifiers; otherwise the field contains AggregationIdentity object instance identifiers.

3.7.9.2.2   The AggregationIdentity objects referenced, either directly or indirectly via groups, by the enableInstances field shall be the AggregationIdentity objects to match.

3.7.9.2.3   The id of the enableInstances field shall support the wildcard value of '0' and matches all AggregationIdentity objects of the provider.

3.7.9.2.4   The service provider shall check for the wildcard value in the list of object instance identifiers in the enableInstances field first and if found no other checks of supplied object instance identifiers shall be made.

3.7.9.2.5   If the enableInstances field contains a value of TRUE then reports of matching AggregationIdentity objects shall be generated, a value of FALSE requests that reports will not be generated.

3.7.9.2.6   No error shall be raised if the enableInstances Boolean value supplied is the same as the current generationEnabled field of the definition for a matched AggregationIdentity object; i.e., enabling an already enabled aggregation will not result in an error.

3.7.9.2.7   If a requested AggregationIdentity or GroupIdentity object is unknown then an UNKNOWN error shall be returned.

3.7.9.2.8   If a requested Group, or the Group objects referenced by that Group, does not contain AggregationIdentity objects then an INVALID error shall be returned.

3.7.9.2.9   If an error is raised then no modifications shall be made as a result of this operation call.

3.7.9.2.10 The provider shall create and store a new AggregationDefinition object in the COM archive if the generationEnabled field is changed.

3.7.9.2.11 If a new AggregationDefinition object is created then that new object shall be the current AggregationDefinition used for the specific AggregationIdentity.

3.7.9.2.12 If the generation of reports is being enabled, and the aggregation is defined as being periodic, then the provider shall generate a report immediately and start the report interval from that report.

3.7.9.3 Errors

The operation may return one of the following errors:

    a) ERROR: UNKNOWN:

        1) one or more of the requested aggregations or groups is unknown;

        2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

    b) ERROR: INVALID:

        1) one of the supplied groups is either not a group of groups or a group of AggregationIdentity objects;

        2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

3.7.10 OPERATION: ENABLEFILTER

3.7.10.1 Overview

The enableFilter operation allows a consumer to control whether reports for specific aggregations are filtered or not. The operation allows the consumer to select the aggregations directly or indirectly using groups. This affects both periodic and ad-hoc aggregations.

Operation Identifier	enableFilter	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	isGroupIds : (MAL::Boolean) enableInstances : (List<COM::InstanceBooleanPair>)

3.7.10.2 Structures

3.7.10.2.1 If the isGroupIds field is TRUE then the enableInstances field shall contain GroupIdentity object instance identifiers; otherwise the field contains AggregationIdentity object instance identifiers.

3.7.10.2.2 The AggregationIdentity objects referenced, either directly or indirectly via groups, by the enableInstances field shall be the AggregationIdentity objects to match.

3.7.10.2.3 The id of the enableInstances field shall support the wildcard value of '0' and matches all AggregationIdentity objects of the provider.

3.7.10.2.4 The service provider shall check for the wildcard value in the list of object instance identifiers in the enableInstances field first and if found no other checks of supplied object instance identifiers shall be made.

3.7.10.2.5 If the enableInstances field contains a value of TRUE then reports of matching AggregationIdentity objects shall be filtered, a value of FALSE requests that reports will not be filtered.

3.7.10.2.6 No error shall be raised if the enableInstances Boolean value supplied is the same as the current filterEnabled field of the definition for a matched AggregationIdentity object; i.e., filtering an already filtered aggregation will not result in an error.

3.7.10.2.7 If a requested AggregationIdentity or GroupIdentity object is unknown then an UNKNOWN error shall be returned.

3.7.10.2.8 If a requested Group, or the Group objects referenced by that Group, does not contain AggregationIdentity objects then an INVALID error shall be returned.

3.7.10.2.9 If an error is raised then no modifications shall be made as a result of this operation call.

3.7.10.2.10 The provider shall create and store a new AggregationDefinition object in the COM archive if the filterEnabled field is changed.

3.7.10.2.11 If a new AggregationDefinition object is created then that new object shall be the current AggregationDefinition used for the specific AggregationIdentity.

3.7.10.3 Errors

The operation may return one of the following errors:

    a) ERROR: INVALID:

        1) one of the supplied groups is either not a group of groups or a group of AggregationIdentity objects;

        2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied;

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

    b) ERROR: UNKNOWN:

        1) one or more of the requested aggregations or groups is unknown;

        2) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.7.11 OPERATION: LISTDEFINITION

3.7.11.1 Overview

The listDefinition operation allows a consumer to request the latest object instance identifiers of the AggregationIdentity and AggregationDefinition objects for the supported aggregations of the provider.

Operation Identifier	listDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	aggNames : (List<MAL::Identifier>)
OUT	RESPONSE	objInstIds : (List<ObjectInstancePair>)

3.7.11.2 Structures

3.7.11.2.1 The aggNames field shall contain a list of aggregation names to retrieve the AggregationIdentity and AggregationDefinition object instance identifiers for.

3.7.11.2.2 The aggNames field may contain the wildcard value of '*' to return all supported AggregationIdentity and AggregationDefinition objects.

3.7.11.2.3 The wildcard value should be checked for first, if found no other checks of supplied identifiers shall be made.

3.7.11.2.4 If a provided identifier does not include a wildcard and does not match an existing AggregationIdentity object then this operation shall fail with an UNKNOWN error.

3.7.11.2.5 The response shall contain a list of matching AggregationIdentity and AggregationDefinition object instance identifiers.

3.7.11.2.6 The returned list shall maintain the same order as the submitted list unless the wildcard value was included in the request.

3.7.11.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

   a) one of the supplied identifiers is unknown;

   b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.7.12 OPERATION: ADDAGGREGATION

3.7.12.1 Overview

The addAggregation operation allows a consumer to define one or more aggregations that do not currently exist.

The new AggregationIdentity and AggregationDefinition objects are expected to be stored in the COM archive by the provider of the aggregation service.

Operation Identifier	addAggregation	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	aggDefDetails : (List<AggregationCreationRequest>)
OUT	RESPONSE	newObjInstIds : (List<ObjectInstancePair>)

3.7.12.2 Structures

3.7.12.2.1 The aggDefDetails field shall hold the name and the AggregationDefinitionDetails to be added.

3.7.12.2.2 The name field must not be the wildcard '*', or empty (an INVALID error shall be returned in this case).

3.7.12.2.3 If the supplied reportInterval or sampleInterval values are not supported by the provider then an INVALID error shall be returned.

3.7.12.2.4 The supplied name must be unique among all AggregationIdentity objects for the domain of the provider; otherwise a DUPLICATE error shall be raised.

3.7.12.2.5 If an error is raised then no new identities and definitions shall be added as a result of this operation call.

3.7.12.2.6 If the supplied name matches an existing, but removed, AggregationIdentity then that AggregationIdentity shall be reused; otherwise a new AggregationIdentity shall be created.

3.7.12.2.7 The provider shall create a new AggregationDefinition object and store it, and any new AggregationIdentity objects, in the COM archive.

3.7.12.2.8 The response shall contain the list of object instance identifiers for the AggregationIdentity and new AggregationDefinition objects.

3.7.12.2.9 The returned list shall maintain the same order as the submitted definitions.

3.7.12.3 Errors

The operation may return one of the following errors:

a) ERROR: DUPLICATE:

   1) one or more of the aggregation objects being added has supplied an aggregation name that is already in use in the domain;

   2) the extra information field contains a list of the indexes of the erroneous values from the originating request list;

Error	Error #	ExtraInfo Type
DUPLICATE	Defined in COM	List<MAL::UInteger>

b) ERROR: INVALID:

   1) one of the supplied aggregation objects contains an invalid name or a supplied interval is not supported by the provider;

   2) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

3.7.13 OPERATION: UPDATEDEFINITION

3.7.13.1 Overview

The updateDefinition operation allows a consumer to update a definition for one or more aggregations.

This differs from deleting an existing aggregation and adding a new definition with the same aggregation name in the fact that the AggregationIdentity object is not changed between the two definitions.

The replacement definition is expected to be stored in the COM archive by the service provider. The operation does not remove the previous object from the COM archive, merely removes the object from the provider.

Operation Identifier	updateDefinition	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Signature
IN	REQUEST	aggInstIds : (List<MAL::Long>) aggDefDetails : (List<AggregationDefinitionDetails>)
OUT	RESPONSE	newObjInstIds : (List<MAL::Long>)

3.7.13.2 Structures

3.7.13.2.1 The aggInstIds field shall contain the object instance identifiers of the AggregationIdentity objects to be updated.

3.7.13.2.2 The supplied object instance identifiers shall match existing identity objects, an UNKNOWN error shall be raised if this is not the case.

3.7.13.2.3 If the aggInstIds list contains either NULL or '0' an INVALID error shall be raised.

3.7.13.2.4 The aggDefDetails field shall contain the replacement AggregationDefinitionDetails.

3.7.13.2.5 The two lists shall be ordered the same.

3.7.13.2.6 The number of entries in the two lists shall be the same size; otherwise an INVALID error shall be returned.

3.7.13.2.7 If the supplied reportInterval or sampleInterval values are not supported by the provider then an INVALID error shall be returned.

3.7.13.2.8 If an error is raised then no definitions shall be updated as a result of this operation call.

3.7.13.2.9 The provider shall create a new AggregationDefinition object and store it in the COM archive.

3.7.13.2.10 The new AggregationDefinition object shall be the current AggregationDefinition used for the specific AggregationIdentity.

3.7.13.2.11 The response shall contain the list of object instance identifiers for the new AggregationDefinition objects.

3.7.13.2.12 The returned list shall maintain the same order as the submitted definitions.

3.7.13.3 Errors

The operation may return one of the following errors:

a) ERROR: UNKNOWN:

   1) one of the supplied AggregationIdentity object instance identifiers is unknown;

   2) a list of the indexes of the error values shall be contained in the extra information field;

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

b) ERROR: INVALID:

   1) the supplied object instance identifiers list contains either a NULL or '0' or the two supplied lists are not the same length or a supplied interval is not supported by the provider;

   2) if the two lists are not the same length then the extra information field shall contain the first index of the element in the largest list which does not have corresponding element in the other list;

   3) the extra information field contains a list of the indexes of the erroneous values from the originating list supplied.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

3.7.14 OPERATION: REMOVEAGGREGATION

3.7.14.1 Overview

The removeAggregation operation allows a consumer to remove one or more aggregations from the list of aggregations supported by the aggregation provider.

The operation does not remove the AggregationIdentity or AggregationDefinition objects from the COM archive, merely removes the objects from the provider. This permits existing AggregationValueInstance objects to continue to reference the correct AggregationDefinition object in the COM archive.

Operation Identifier	removeAggregation	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Signature
IN	SUBMIT	aggInstIds : (List<MAL::Long>)

3.7.14.2 Structures

3.7.14.2.1 The aggInstIds field shall hold the object instance identifiers of the AggregationIdentity objects to be removed from the provider.

3.7.14.2.2 The list may contain the wildcard value of '0'.

3.7.14.2.3 The wildcard value should be checked for first, if found no other checks of supplied object instance identifiers shall be made.

3.7.14.2.4 If a provided AggregationIdentity object instance identifier does not include a wildcard and does not match an existing aggregation then this operation shall fail with an UNKNOWN error.

3.7.14.2.5 Matched AggregationIdentity and AggregationDefinition objects shall not be removed from the COM archive only the list of AggregationIdentity and AggregationDefinition objects in the provider.

3.7.14.2.6 If an error is raised then no aggregations shall be removed as a result of this operation call.

3.7.14.2.7 If the operation succeeds then the provider shall not publish aggregation values for the deleted AggregationIdentity objects anymore.

3.7.14.3 Errors

The operation may return the following error: ERROR: UNKNOWN:

    a) one of the supplied AggregationIdentity object instance identifiers is unknown;

    b) a list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

3.8   SERVICE: CONVERSION

3.8.1   OVERVIEW

The conversion service provides a set of basic conversion definition types that allows the specification of a conversion between two representations. These conversions are used by the other M&C services (such as Action, Alert, and Parameter) to define conversions from raw field representations to some engineering representation.

Conversions are associated with other entities such as parameters or action/alert arguments through the configuration of the relevant service (action/alert/parameter).

The conversion service does not provide any operations directly, but allows consumers to add, remove, and modify conversion definitions via the COM archive.

Table 3-16: Conversion Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MC	Conversion	4	7	1
Interaction Pattern	Operation Identifier	Operation Number	Support in Replay	Capability Set

3.8.2   HIGH-LEVEL REQUIREMENTS

The conversion service shall provide the capability for maintaining the list of conversion definitions.

3.8.3   FUNCTIONAL REQUIREMENTS

3.8.3.1   The conversion service shall use the COM archive operations for maintaining the conversion definitions.

3.8.3.2   Updating a conversion definition shall be performed by adding a new conversion definition and referencing it in the service specific COM objects (such as ParameterDefinition).

3.8.3.3   A Discrete conversion must contain at least one pair of values in the mapping.

3.8.3.4   A Line conversion must contain at least two line points.

3.8.3.5   A Polynomial conversion must contain at least one pair of values in the points list.

3.8.3.6   A Range conversion must contain at least one pair of values in the points list.

3.8.3.7   The service that references a conversion should ensure that the conversion referenced is correct.

3.8.4   COM USAGE

3.8.4.1   Each conversion shall be represented by a ConversionIdentity COM object.

3.8.4.2   The body of the ConversionIdentity COM object shall hold the name of the conversion.

3.8.4.3   The definitions of the conversions shall be represented as either DiscreteConversion, LineConversion, PolyConversion, or RangeConversion COM objects.

3.8.4.4   The conversion definition object shall use the related link to indicate which ConversionIdentity object it uses.

3.8.4.5   The source link of the ConversionIdentity object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.8.4.6   The source link of the conversion definition object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.8.4.7   Other service objects shall reference instances of a conversion using the object instance identifier of the conversion definition.

Table 3-17: Conversion Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
ConversionIdentity	1	MAL::Identifier	Set to NULL	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
DiscreteConversion	2	DiscreteConversionDetails	1	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
LineConversion	3	LineConversionDetails	1	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
PolyConversion	4	PolyConversionDetails	1	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
RangeConversion	5	RangeConversionDetails	1	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.8.5   COM OBJECT RELATIONSHIPS

Figure 3-14 below shows the COM object relationships for this service.

Figure 3-14: Conversion Service COM Object Relationships

3.8.6   COM ARCHIVE SERVICE USAGE

3.8.6.1   ConversionIdentity and conversion definition objects should be stored in the COM archive.

3.8.6.2   Each instance of a conversion definition shall be represented by an object instance held in the COM archive.

3.8.6.3   The COM archive service operations shall be used to add, update, and remove conversions.

3.8.6.4   Other services should monitor the archive service for changes to conversions they reference.

3.9 SERVICE: GROUP

3.9.1 OVERVIEW

The group service provides a mechanism for other services to reference sets of their own objects using a single group reference. These groups are used by the other M&C services (such as Action, Alert, Check, Aggregation, and Parameter) to reduce the complexity of operations by allowing consumers to reference groups of objects (such as parameters) in operations rather than having to supply large lists of object references.

Where operations of other service mention the use of groups in their operations, any reference to a group object instance identifier implicitly means a GroupIdentity object.

Groups of other groups are supported; however, all objects within the group of groups should have the same object type as most operations expect a single type.

The creation of cyclic group of groups should also be avoided.

The group service does not provide any operations directly, but allows consumers to add, remove, and modify groups via the COM archive.

Table 3-18: Group Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MC	Group	4	8	1
Interaction Pattern	Operation Identifier	Operation Number	Support in Replay	Capability Set

3.9.2 HIGH-LEVEL REQUIREMENTS

The group service shall provide the capability for maintaining the list of groups.

3.9.3 FUNCTIONAL REQUIREMENTS

3.9.3.1   The group service shall use the COM archive operations for maintaining the group definitions.

3.9.3.2   Updating a group definition shall be performed by adding a new group definition and referencing it in the service specific COM objects.

3.9.4 COM USAGE

3.9.4.1   Each group shall be represented by a GroupIdentity COM object.

3.9.4.2   The body of the GroupIdentity COM object shall hold the name of the group.

3.9.4.3   The definitions of the groups shall be represented as GroupDefinition COM objects.

3.9.4.4   The GroupDefinition object shall use the related link to indicate which GroupIdentity object it uses.

3.9.4.5   The source link of the GroupIdentity object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.9.4.6   The source link of the GroupDefinition object should be the object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.9.4.7   Groups of groups shall reference instances of a group using the object instance identifier of the GroupIdentity.

3.9.4.8   Other service objects shall reference instances of a group using the object instance identifier of the GroupIdentity.

Table 3-19: Group Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
GroupIdentity	1	MAL::Identifier	Set to NULL	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.
GroupDefinition	2	GroupDetails	1	The object that caused it to be created, most likely a COM OperationActivity object or an operator login in the case of off-line editors being used.

3.9.5   DISCUSSION—COM OBJECT RELATIONSHIPS

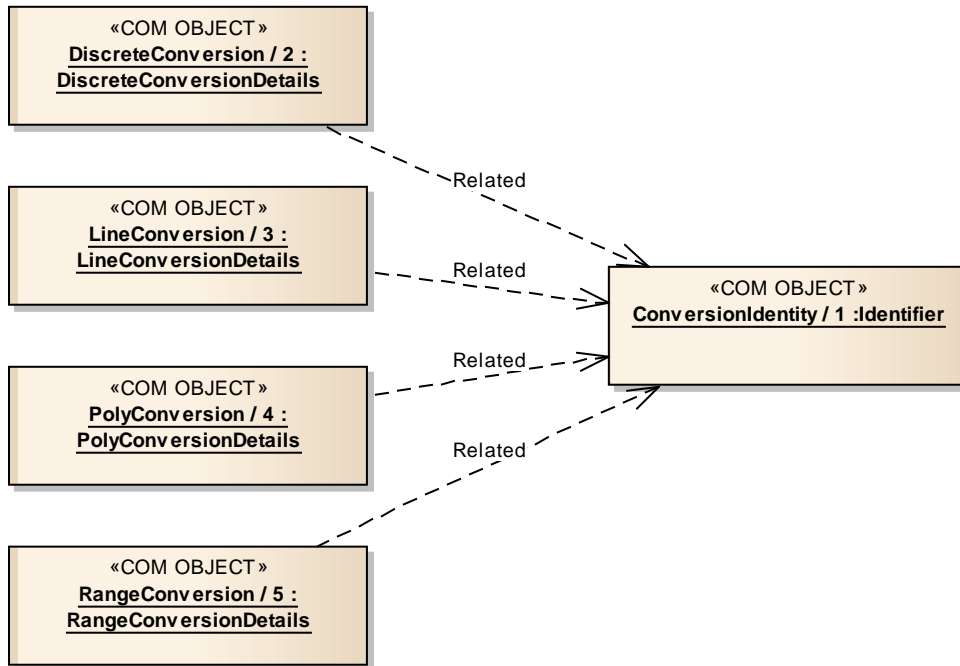Figure 3-15 below shows the COM object relationships for this service.
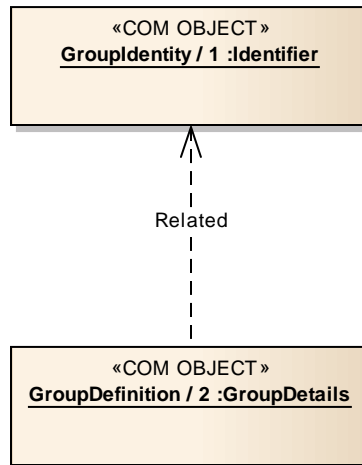
Figure 3-15: Group Service COM Object Relationships

3.9.6   COM ARCHIVE SERVICE USAGE

3.9.6.1   GroupIdentity and GroupDefinition objects should be stored in the COM archive.

3.9.6.2   The COM archive service operations shall be used to add, update, and remove groups.

3.9.6.3   Other services should monitor the archive service for changes to groups they reference.

4 DATA TYPES

4.1 AREA DATA TYPES: MC

4.1.1 ENUMERATION: SEVERITY

The severity enumeration shall be used to hold the possible values for a severity.

NOTE – The numerical value represents the increasing severity, and therefore CRITICAL is more severe than ALARM. Normally, for checks, only the Warning and Critical ranges are used: the colour yellow is associated with Warning, and the colour red is associated with Critical.

Name	Severity	
Short Form Part	6	
Enumeration Value	Numerical Value	Comment
INFORMATIONAL	1	Indicates a nominal situation with no consequences.
WARNING	2	Indicates unexpected behaviour without directly threatening consequences.
ALARM	3	Indicates behaviour of serious concern requiring the attention of an operator, but not necessarily a malfunction.
SEVERE	4	Indicates that the monitored item has malfunctioned. Requires operator attention.
CRITICAL	5	Indicates behaviour with mission threatening consequences. Requires operator attention.

4.1.2 COMPOSITE: ARGUMENTDEFINITIONDETAILS

4.1.2.1 The ArgumentDefinitionDetails structure shall be used to hold the details of an argument definition with a set of associated attributes, such as conversion used.

4.1.2.2 The conditionalConversions shall define the conditions where a referenced conversion is applied.

4.1.2.3 Only the first TRUE conversion should be applied.

Name	ArgumentDefinitionDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
rawType	MAL::Octet	No	Holds the attribute short form part of the raw type of the argument; e.g., for a MAL::String argument it shall hold 15.
rawUnit	MAL::String	Yes	The unit for the raw value.
conditionalConversions	List<ConditionalConversion>	Yes	The conditional conversions to apply to the argument. Only the first TRUE conversion should be applied.
convertedType	MAL::Octet	Yes	Holds the attribute short form part of the converted type of the argument; e.g., for a MAL::String argument it shall hold 15. Must not be NULL if a conversion condition is supplied.
convertedUnit	MAL::String	Yes	The converted argument units.

4.1.3 COMPOSITE: ATTRIBUTEVALUE

The AttributeValue structure shall be used to hold an Attribute value.

NOTE – It allows a list of different Attribute types to be created, whereas List<Attribute> would require the values to be all of the same type.

Name	AttributeValue		
Extends	MAL::Composite		
Short Form Part	2		
Field	Type	Nullable	Comment
value	MAL::Attribute	No	The argument value. Must not be NULL. NULL may be represented by having a NULL in place of the complete AttributeValue composite.

4.1.4   COMPOSITE: CONDITIONALCONVERSION

4.1.4.1   The ConditionalConversion structure shall be used to hold a condition expression to be evaluated to determine if a specific Conversion should be used.

4.1.4.2   In the case that no test is required, i.e., the conversion should always be used, then the condition field should be set to NULL.

Name	ConditionalConversion		
Extends	MAL::Composite		
Short Form Part	3		
Field	Type	Nullable	Comment
condition	ParameterExpression	Yes	The expression indicates which entities are applicable for this check. If NULL, then the condition shall evaluate to TRUE.
conversionId	COM::ObjectKey	No	The object instance identifier of the ConversionIdentity object to be used if the condition evaluates to TRUE or is NULL.

4.1.5   COMPOSITE: PARAMETEREXPRESSION

The ParameterExpression structure shall be used to represent a simple expression between a parameter and a value for that parameter.

Name	ParameterExpression		
Extends	MAL::Composite		
Short Form Part	4		
Field	Type	Nullable	Comment
parameterId	COM::ObjectKey	No	Holds the object instance identifier of the ParameterIdentity object to be used in the expression.
operator	COM::Archive::ExpressionOperator	No	The expression operator.
useConverted	MAL::Boolean	No	If set to TRUE the converted value field of the parameter value should be used; otherwise the raw value field should be used.
value	MAL::Attribute	Yes	The value to be used in the expression.

4.1.6   COMPOSITE: OBJECTINSTANCEPAIR

The ObjectInstancePair structure shall be used to hold the object instance identifier of an Identity object with its associated Definition object.

Name	ObjectInstancePair		
Extends	MAL::Composite		
Short Form Part	7		
Field	Type	Nullable	Comment
objIdentityInstanceId	MAL::Long	No	The object instance identifier of the Identity object.
objDefInstanceId	MAL::Long	No	The object instance identifier of the Definition object.

4.2 SERVICE DATA TYPES: ACTION

4.2.1 ENUMERATION: ACTIONCATEGORY

Contains the default Action category values. It is implementation specific what the meaning of the values are in a particular context.

Name	ActionCategory	
Short Form Part	4	
Enumeration Value	Numerical Value	Comment
DEFAULT	1	Default category
HIPRIORITY	2	Category for high priority actions
CRITICAL	3	Category for critical actions

4.2.2 COMPOSITE: ACTIONDEFINITIONDETAILS

The ActionDefinitionDetails structure holds the definition information of an action.

Name	ActionDefinitionDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
description	MAL::String	No	The description of the action.
category	MAL::UOctet	No	Category of the action. Value taken from ActionCategory enumeration, although the use of a UOctet allows deployment specific extension. Extensions must use values greater than 127.
progressStepCount	MAL::UShort	No	Total number of steps that will be reported if PROGRESS reporting is selected in the sent Action. 0 if PROGRESS reporting is not used.
arguments	List<ArgumentDefinitionDetails>	Yes	The list of argument definitions. If no arguments are defined, then the complete list is replaced with a NULL.
argumentIds	List<MAL::Identifier>	Yes	Optional list of argument definition identifiers. Allows the provider to verify that the correct arguments are being supplied when using the same field in the action instance.

4.2.3 COMPOSITE: ACTIONINSTANCEDETAILS

The ActionInstanceDetails structure holds the information required for an instance of an Action such as the argument values to use.

Name	ActionInstanceDetails		
Extends	MAL::Composite		
Short Form Part	2		
Field	Type	Nullable	Comment
defInstId	MAL::Long	No	The object instance identifier of the ActionDefinition to be used.
stageStartedRequired	MAL::Boolean	No	If TRUE, then an activity event of type Execution is required for the STARTED stage.
stageProgressRequired	MAL::Boolean	No	If TRUE, then activity events of type Execution are required for the PROGRESS stages.
stageCompletedRequired	MAL::Boolean	No	If TRUE, then an activity event of type Execution is required for the COMPLETION stage.
argumentValues	List<AttributeValue>	Yes	List containing the values of the arguments. The ordering of the list matches that of the definition. If a value for a particular entry is not being supplied, then its position is filled with a NULL value. If no arguments are defined, then the complete list is replaced with a NULL.
argumentIds	List<MAL::Identifier>	Yes	Optional list of argument definition identifiers. Allows the provider to verify that the correct arguments are being supplied. The ordering of the list matches that of the argument list of the action definition.
isRawValue	List<MAL::Boolean>	Yes	Optional list of Booleans that determine whether the supplied argument values are raw or converted. If the Boolean for a particular value is TRUE or NULL then that value is assumed to be raw. If the complete isRawValue list is NULL then all arguments are assumed to be raw values. The ordering of the list matches that of the argument list of the action definition.

4.2.4 COMPOSITE: ACTIONCREATIONREQUEST

The ActionCreationRequest contains all the fields required when creating a new action in a provider.

Name	ActionCreationRequest		
Extends	MAL::Composite		
Short Form Part	3		
Field	Type	Nullable	Comment
name	MAL::Identifier	No	The name of the action. Must not be empty or the wildcard value.
actionDefDetails	ActionDefinitionDetails	No	The action definition details.

4.3 SERVICE DATA TYPES: PARAMETER

4.3.1 ENUMERATION: VALIDITYSTATE

The ValidityState enumeration shall be used to hold the validity states and their numeric values.

Name	ValidityState	
Short Form Part	4	
Enumeration Value	Numerical Value	Comment
VALID	0	Valid.
EXPIRED	1	The parameter has a timeout associated which has expired
INVALID_RAW	2	The parameter raw value cannot be obtained, or calculated for synthetic parameters
INVALID_CONVERSION	3	The validity expression either has evaluated to TRUE or there is no validity defined, but the conversion of the parameter value has failed (for example an unexpected value for a discrete conversion)
UNVERIFIED	4	The validity of the validity expression has been evaluated to FALSE and therefore cannot be used to verify the current value
INVALID	5	The validity expression has been evaluated to FALSE

4.3.2   COMPOSITE: PARAMETERDEFINITIONDETAILS

The ParameterDefinitionDetails structure holds a parameter definition. The conversion field defines the conditions where the relevant conversion is applied. For onboard parameters, the report interval should be a multiple of the minimum sampling interval of that parameter.

Name	ParameterDefinitionDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
description	MAL::String	No	The description of the parameter. May be empty.
rawType	MAL::Octet	No	Holds the attribute short form part of the raw type of the parameter; e.g., for a MAL::String parameter it shall hold 15.
rawUnit	MAL::String	Yes	The unit for the raw value. If NULL then raw type has no unit.
generationEnabled	MAL::Boolean	No	Controls whether reports for this parameter are to be generated.
reportInterval	MAL::Duration	No	Periodic report interval. No periodic reports to be generated if this is set to '0'.
validityExpression	ParameterExpression	Yes	Expression that determines this parameter's validity state. Can be NULL if no validity check is required or validity is calculated by implementation-specific mechanisms.
conversion	ParameterConversion	Yes	If present then parameter has a converted type.

4.3.3 COMPOSITE: PARAMETERVALUE

4.3.3.1 The ParameterValue structure shall be used to hold a specific value of the parameter.

4.3.3.2 The type of the value shall match that specified in the parameter definition.

Name	ParameterValue		
Extends	MAL::Composite		
Short Form Part	2		
Field	Type	Nullable	Comment
validityState	MAL::UOctet	No	Holds the validity state for a parameter value. If the parameter is valid then this should be set to '0'.
rawValue	MAL::Attribute	Yes	The parameter raw value. The value of NULL is a valid value and carries no special significance in the parameter service.
convertedValue	MAL::Attribute	Yes	The parameter converted value.

4.3.4 COMPOSITE: PARAMETERCONVERSION

The ParameterConversion structure shall be used to hold information about the conversions to be applied to a parameter.

Name	ParameterConversion		
Extends	MAL::Composite		
Short Form Part	3		
Field	Type	Nullable	Comment
convertedType	MAL::Octet	No	Holds the attribute short form part of the converted type of the parameter; e.g., for a MAL::String parameter it shall hold 15.
convertedUnit	MAL::String	Yes	The converted parameter unit. If NULL then converted type has no unit.
conditionalConversions	List<ConditionalConversion>	No	The conversions to be applied. Only the first TRUE conversion should be applied.

4.3.5   COMPOSITE: PARAMETERCREATIONREQUEST

The ParameterCreationRequest structure shall be used to contain all the fields required when creating a new parameter in a provider.

Name	ParameterCreationRequest		
Extends	MAL::Composite		
Short Form Part	5		
Field	Type	Nullable	Comment
name	MAL::Identifier	No	The name of the parameter. Must not be empty or the wildcard value.
paramDefDetails	ParameterDefinitionDetails	No	The parameter definition details.

4.3.6   COMPOSITE: PARAMETERRAWVALUE

The ParameterRawValue structure shall be used to hold a new raw value for a specific parameter.

Name	ParameterRawValue		
Extends	MAL::Composite		
Short Form Part	6		
Field	Type	Nullable	Comment
paramInstId	MAL::Long	No	The object instance identifier of the parameter identity.
rawValue	MAL::Attribute	Yes	The parameter raw value. The value of NULL is a valid value and carries no special significance in the parameter service.

4.3.7   COMPOSITE: PARAMETERVALUEDETAILS

4.3.7.1   The ParameterValueDetails structure shall be used to hold a specific time stamped value of the parameter.

4.3.7.2   The type of the value shall match that specified in the parameter definition.

Name	ParameterValueDetails		
Extends	MAL::Composite		
Short Form Part	7		
Field	Type	Nullable	Comment
paramId	MAL::Long	No	The ParameterIdentity object instance identifier.
defId	MAL::Long	No	The ParameterDefinition object instance identifier.
timestamp	MAL::Time	No	The timestamp of the value.
value	ParameterValue	No	The parameter value.

4.4   SERVICE DATA TYPES: ALERT

4.4.1   COMPOSITE: ALERTDEFINITIONDETAILS

The AlertDefinitionDetails structure shall be used to provide the definition of an alert including any argument definitions.

Name	AlertDefinitionDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
description	MAL::String	No	The description of the alert.
severity	Severity	No	Severity of the alert.
generationEnabled	MAL::Boolean	No	Controls whether instances of this alert are to be generated.
arguments	List<ArgumentDefinitionDetails>	No	The list of argument definitions.
argumentIds	List<MAL::Identifier>	Yes	Optional list of argument definition identifiers. Allows the provider to verify that the correct arguments are being supplied. If null, the argument definition identifiers are not checked before the emission of an Alert Event.

4.4.2   COMPOSITE: ALERTEVENTDETAILS

The AlertEventDetails structure shall be used to hold the details of an instance of an alert.

Name	AlertEventDetails		
Extends	MAL::Composite		
Short Form Part	2		
Field	Type	Nullable	Comment
argumentValues	List<AttributeValue>	Yes	List containing the values of the arguments. The ordering of the list matches that of the definition. If a value for a particular entry is not being supplied, then its position is filled with a NULL value. If no arguments are defined, then the complete list is replaced with a NULL.
argumentIds	List<MAL::Identifier>	Yes	Optional list of argument definition identifiers. Allows the consumer to verify that the correct arguments are being supplied. The ordering of the list matches that of the argument list of the alert definition.

4.4.3 COMPOSITE: ALERTCREATIONREQUEST

The AlertCreationRequest structure shall be used to contain all the fields required when creating a new alert in a provider.

Name	AlertCreationRequest		
Extends	MAL::Composite		
Short Form Part	3		
Field	Type	Nullable	Comment
name	MAL::Identifier	No	Alert name. Must not be empty or wildcard value.
alertDefDetails	AlertDefinitionDetails	No	The alert definition details.

4.5 SERVICE DATA TYPES: CHECK

4.5.1 ENUMERATION: CHECKSTATE

4.5.1.1 The CheckState enumeration shall be used to hold the possible basic states of a check.

4.5.1.2 The meaning of the NOT_OK value is check specific and shall be detailed in the relevant check type definition.

Name	CheckState	
Short Form Part	6	
Enumeration Value	Numerical Value	Comment
DISABLED	1	The check is disabled.
UNCHECKED	2	The check is enabled but has not passed the selection condition expression.
INVALID	3	Check is enabled, has passed the selection condition, but the entity being checked is not in a valid state and therefore has not been checked.
OK	4	The check is OK.
NOT_OK	5	The check is not OK.

4.5.2 COMPOSITE: CHECKDEFINITIONDETAILS

The CheckDefinitionDetails structure shall be used to hold the definition of a check.

Name	CheckDefinitionDetails		
Extends	MAL::Composite		
Abstract			
Field	Type	Nullable	Comment
description	MAL::String	No	The description of the check. May be empty.
checkSeverity	Severity	No	Indicates the seriousness of the violation based on its possible negative consequences.
maxReportingInterval	MAL::Duration	No	Maximum interval that can elapse between generations of CheckResult reports. If this value expires, then a CheckResult is generated with the same state for the previous and current state. If set to '0', then no maximum reporting interval shall be applied.
nominalCount	MAL::UInteger	No	Number of consecutive valid samples passing the check for the check to be OK.
nominalTime	MAL::Duration	No	If nominalCount is zero then this is duration that a parameter is continuously passing the check for the check to be OK. If nominalCount is not zero, this is the period over which samples will be used in the nominalCount calculation; i.e., samples further in the past than nominalTime are not considered.
violationCount	MAL::UInteger	No	Number of consecutive valid samples violating the check for the check to be in violation.
violationTime	MAL::Duration	No	If violationCount is zero then this is duration that a parameter is continuously violating the check for the check to be in violation. If violationCount not zero, this is the period over which samples will be used in the violationCount calculation; i.e., samples further in the past than violationTime are not considered.

4.5.3 COMPOSITE: CHECKLINKDETAILS

The CheckLinkDetails structure shall be used to represent the link from a check definition to a check result for a specific parameter.

Name	CheckLinkDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
checkEnabled	MAL::Boolean	No	TRUE if the check instance is enabled.
checkOnChange	MAL::Boolean	No	If TRUE then any change to state or value of the parameter, or the check condition will trigger a check evaluation. Ignored for Compound checks.
useConverted	MAL::Boolean	No	If set to TRUE the converted value field of the parameter value should be used; otherwise the raw value field should be used. Ignored for Compound checks.
checkInterval	MAL::Duration	No	The interval that a check should be applied. Only applicable if checkOnChange is FALSE. If '0', then no periodic checking shall be performed, and a check will be triggered by another mechanism. Ignored for Compound checks.
condition	ParameterExpression	Yes	Should this check be applied; if NULL then always applied.

4.5.4 COMPOSITE: CHECKRESULT

4.5.4.1 The CheckResult structure shall be used to hold basic information about the check state and the value of the parameter at the time of the check.

4.5.4.2 The timestamp of the event shall be the transition time of the check.

Name	CheckResult		
Extends	MAL::Composite		
Short Form Part	2		
Field	Type	Nullable	Comment
previousCheckState	CheckState	No	The previous evaluation state of the check. Initially UNCHECKED for the first transition of a check. For check evaluations that do not detect a check transition, this value will be the same as the currentCheckState.
currentCheckState	CheckState	No	The current evaluation state of the check
paramDefInstId	MAL::Long	Yes	The object instance identifier of the ParameterDefinition used for the check evaluation. NULL if compound check.
checkedValue	MAL::Attribute	Yes	This is the value of the parameter or for a compound check the number of checks in violation at the time of a check state transition; or, if it is a report resulting from expiration of the CheckDefinitionDetails maxReportingInterval, it is the value or the number when the interval expired.

4.5.5 COMPOSITE: CHECKLINKSUMMARY

The CheckLinkSummary structure shall be used to hold the IDs of a specific check link and the check and parameter it links to.

Name	CheckLinkSummary		
Extends	MAL::Composite		
Short Form Part	3		
Field	Type	Nullable	Comment
checkId	MAL::Long	No	The object instance identifier of the CheckIdentity object.
linkId	MAL::Long	No	The object instance identifier of the CheckLink object.
linkDefinitionId	MAL::Long	No	Contains the object instance identifier of the CheckLinkDefinition object.
checkEnabled	MAL::Boolean	No	TRUE if the check instance is enabled.
parameterId	COM::ObjectKey	Yes	The object instance identifier of the ParameterIdentity object for the check link. NULL for Compound checks.

4.5.6 COMPOSITE: CHECKRESULTSUMMARY

The CheckResultSummary structure shall be used to hold details about a specific check link and its evaluated result.

Name	CheckResultSummary		
Extends	MAL::Composite		
Short Form Part	4		
Field	Type	Nullable	Comment
linkId	MAL::Long	No	The object instance identifier of the check link.
checkEnabled	MAL::Boolean	No	The current enabled state of the check link.
parameterId	COM::ObjectKey	Yes	The object instance key of the ParameterIdentity being checked. NULL only for Compound checks.
evaluationTime	MAL::Time	No	The timestamp of the check result. If as a result of max reporting interval expiring then it shall contain the expiration timestamp.
result	CheckResult	No	The check result value.

4.5.7   COMPOSITE: CHECKRESULTFILTER

The CheckResultFilter structure shall be used to hold a filter for the current check result transition information.

Name	CheckResultFilter		
Extends	MAL::Composite		
Short Form Part	5		
Field	Type	Nullable	Comment
checkFilterViaGroups	MAL::Boolean	No	If TRUE then the checkFilter field contains GroupIdentity object instance identifiers that link to CheckIdentity objects; otherwise it contains CheckIdentity object instance identifiers directly.
checkFilter	List<MAL::Long>	No	The list of GroupIdentity object instance identifiers if checkFilterViaGroups is TRUE; otherwise the CheckIdentity object instance identifiers to filter on. A value of '0' means match all.
parameterFilterViaGroups	MAL::Boolean	No	If TRUE then the parameterFilter field contains GroupIdentity object instance identifiers that link to ParameterIdentity objects; otherwise it contains ParameterIdentity object instance identifiers directly.
parameterFilter	List<MAL::Long>	No	The list of GroupIdentity object instance identifiers if parameterFilterViaGroups is TRUE; otherwise the ParameterIdentity object instance identifiers to filter on. A value of '0' means match all.
stateFilter	List<CheckState>	No	The list of required check states to filter on. Empty list means match all.

4.5.8   COMPOSITE: REFERENCEVALUE

The ReferenceValue structure shall be used to define a value to compare against.

NOTE – A validCount of '1' and deltaTime of '0' would compare against the previous sample value.

Name	ReferenceValue		
Extends	MAL::Composite		
Short Form Part	7		
Field	Type	Nullable	Comment
validCount	MAL::UShort	No	Number of valid samples that should be collected to update the reference value.
deltaTime	MAL::Duration	No	Delta time from now into the past from which the reference value should be sampled.
parameterId	COM::ObjectKey	Yes	The ParameterIdentity object to compare against. If NULL, then checked parameter should be compared against itself.

4.5.9   COMPOSITE: CONSTANTCHECKDEFINITION

The ConstantCheckDefinition structure shall be used to hold the constant values to compare against for a consistency check.

Name	ConstantCheckDefinition		
Extends	CheckDefinitionDetails		
Short Form Part	8		
Field	Type	Nullable	Comment
operator	COM::Archive::ExpressionOperator	No	The operator to be used to perform the check.
values	List<AttributeValue>	No	The set of constant values to be checked against. An empty list means that any value change triggers the check.

4.5.10 COMPOSITE: REFERENCECHECKDEFINITION

The ReferenceCheckDefinition structure shall be used to hold the key to another entity to compare against for a consistency check.

Name	ReferenceCheckDefinition		
Extends	CheckDefinitionDetails		
Short Form Part	9		
Field	Type	Nullable	Comment
operator	COM::Archive::ExpressionOperator	No	The operator to be used to perform the check.
checkReference	ReferenceValue	No	The value to check against.

4.5.11 COMPOSITE: DELTACHECKDEFINITION

The DeltaCheckDefinition shall be used to define a delta transition check.

Name	DeltaCheckDefinition		
Extends	CheckDefinitionDetails		
Short Form Part	10		
Field	Type	Nullable	Comment
checkReference	ReferenceValue	No	The value to compare the current value against.
violateInRange	MAL::Boolean	No	If TRUE, then the safe (non violating) values lie outside the specified threshold range.
valueDelta	MAL::Boolean	No	If TRUE, then the thresholds contain value deltas. If FALSE, they contain percentage deltas.
lowerThreshold	MAL::Attribute	Yes	The lower threshold of the delta value. Must be of the correct type for the entity being checked. Must be a Float if percentage threshold in the range (-1.0 to 1.0 representing +-100%).
upperThreshold	MAL::Attribute	Yes	The upper threshold of the delta value. Must be of the correct type for the entity being checked. Must be a Float if percentage threshold in the range (-1.0 to 1.0 representing +-100%).

4.5.12 COMPOSITE: LIMITCHECKDEFINITION

4.5.12.1 The LimitCheckDefinition shall be used to define a high and low limit check.

4.5.12.2 If only one limit is supplied, the other limit shall be assumed to be the relevant maximum supported by the type being checked in this case.

Name	LimitCheckDefinition		
Extends	CheckDefinitionDetails		
Short Form Part	11		
Field	Type	Nullable	Comment
violateInRange	MAL::Boolean	No	If TRUE, then the safe (non violating) values lie outside the specified limits range.
lowerLimit	MAL::Attribute	Yes	The lower limit of the value. Must be of the correct type for the entity being checked.
upperLimit	MAL::Attribute	Yes	The upper limit of the value. Must be of the correct type for the entity being checked.

4.5.13 COMPOSITE: COMPOUNDCHECKDEFINITION

The CompoundCheckDefinition structure shall be used to hold the object instance identifiers of one or more check link objects to monitor for a compound check.

Name	CompoundCheckDefinition		
Extends	CheckDefinitionDetails		
Short Form Part	12		
Field	Type	Nullable	Comment
minimumChecksInViolation	MAL::UInteger	No	The number of referenced checks that must be in violation for this check to be considered in violation. If set to '0' then all referenced checks must be in violation.
checkLinkIds	List<MAL::Long>	No	The set of CheckLink objects that form the compound check.

4.5.14 COMPOSITE: CHECKTYPEDINSTANCE

The CheckTypedInstance structure shall be used to hold the two COM object instance identifiers that form the identity and the body of the check definition in combination with the COM object type of the check body definition.

Name	CheckTypedInstance		
Extends	MAL::Composite		
Short Form Part	13		
Field	Type	Nullable	Comment
objDefCheckType	COM::ObjectType	No	The COM object type of the check body.
objInstIds	ObjectInstancePair	Yes	The object instance identifiers.

4.6 SERVICE DATA TYPES: STATISTIC

4.6.1 COMPOSITE: STATISTICFUNCTIONDETAILS

The StatisticFunctionDetails structure shall be used to hold the details of the function.

Name	StatisticFunctionDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
name	MAL::Identifier	No	The name of the statistical function.
description	MAL::String	No	The description of the statistical function.

4.6.2 COMPOSITE: STATISTICLINKDETAILS

The StatisticLinkDetails structure shall be used to hold the sampling, reporting, and collection intervals for one parameter statistic function link.

Name	StatisticLinkDetails		
Extends	MAL::Composite		
Short Form Part	2		
Field	Type	Nullable	Comment
samplingInterval	MAL::Duration	No	The interval between samples of the parameter.
reportingInterval	MAL::Duration	No	The interval between periodic reports being generated. If set to '0', then no periodic reports shall be sent.
collectionInterval	MAL::Duration	No	The collection and reset interval of the statistical evaluation for the linked parameter. If set to '0', then no periodic reset of the evaluation shall be performed.
resetEveryCollection	MAL::Boolean	No	If TRUE the evaluation will reset its value every collection interval. If FALSE it will maintain a moving evaluation of the function for the collection interval.
reportingEnabled	MAL::Boolean	No	TRUE if reporting of the evaluation instance is enabled.
useConverted	MAL::Boolean	No	If TRUE then use the converted value of the Parameter, else use the raw value

4.6.3 COMPOSITE: STATISTICVALUE

The StatisticValue structure shall be used to hold the statistical result for a parameter.

Name	StatisticValue		
Extends	MAL::Composite		
Short Form Part	3		
Field	Type	Nullable	Comment
paramDefInstId	MAL::Long	No	The object instance identifier of the ParameterDefinition object used for the parameter.
startTime	MAL::Time	Yes	Time the statistic calculations started. This value can be NULL if the start time can be derived by other means, e.g., other start times in a set of StatisticValue structures.
endTime	MAL::Time	Yes	Time the statistic calculations ended. This value can be NULL if the time can be derived by other means, e.g., other times in a set of StatisticValue structures.
valueTime	MAL::Time	Yes	Time the statistic value was reached. The time is only applicable for particular statistic values such as min or max. Shall be NULL if not applicable for cases such as 'mean average'.
value	MAL::Attribute	Yes	Value of the statistic.
sampleCount	MAL::UInteger	No	Holds the number of samples that contributed to the statistic value. For calculated values such as 'mean average' this holds the number of samples that were used to calculate the value, for non-calculated values such as 'min' then it is the number of samples that were in the set evaluated.

4.6.4 COMPOSITE: STATISTICCREATIONREQUEST

The StatisticCreationRequest structure shall be used to hold the link details for a specific parameter and function association.

Name	StatisticCreationRequest		
Extends	MAL::Composite		
Short Form Part	4		
Field	Type	Nullable	Comment
statFuncInstId	MAL::Long	No	The object instance identifier of the statistical function to be used.
parameterId	COM::ObjectKey	No	The object key of the ParameterIdentity object being referenced.
linkDetails	StatisticLinkDetails	No	The collection, reporting, and sampling intervals.

4.6.5 COMPOSITE: STATISTICLINKSUMMARY

The StatisticLinkSummary structure shall be used to hold the IDs of a specific statistic link and the function and parameter it links to.

Name	StatisticLinkSummary		
Extends	MAL::Composite		
Short Form Part	5		
Field	Type	Nullable	Comment
funcId	MAL::Long	No	The object instance identifier of the StatisticFunction object.
linkId	MAL::Long	No	The object instance identifier of the StatisticLink object.
linkDefId	MAL::Long	No	The object instance identifier of the StatisticLinkDefinition object.
reportingEnabled	MAL::Boolean	No	TRUE if reporting of the evaluation instance is enabled.
parameterId	COM::ObjectKey	No	The object instance identifier of the ParameterIdentity object for the statistic link.

4.6.6 COMPOSITE: STATISTICEVALUATIONREPORT

The StatisticEvaluationReport structure shall be used to hold the set of statistical results.

Name	StatisticEvaluationReport		
Extends	MAL::Composite		
Short Form Part	6		
Field	Type	Nullable	Comment
linkId	MAL::Long	No	The statistic link object instance identifier.
value	StatisticValue	No	The statistical evaluation value.

4.7 SERVICE DATA TYPES: AGGREGATION

4.7.1 ENUMERATION: AGGREGATIONCATEGORY

AggregationCategory is an enumeration definition that shall be used to hold the categories of aggregations.

Name	AggregationCategory	
Short Form Part	7	
Enumeration Value	Numerical Value	Comment
GENERAL	1	General aggregation.
DIAGNOSTIC	2	Diagnostic aggregation.

4.7.2 ENUMERATION: THRESHOLDTYPE

ThresholdType is an enumeration definition that shall be used to hold the types of filtering thresholds.

Name	ThresholdType	
Short Form Part	8	
Enumeration Value	Numerical Value	Comment
PERCENTAGE	1	Threshold value is a percentage.
DELTA	2	Threshold value is a delta.

4.7.3 ENUMERATION: GENERATIONMODE

GenerationMode is an enumeration definition that shall be used to hold the reasons for the aggregation to be generated.

Name	GenerationMode	
Short Form Part	9	
Enumeration Value	Numerical Value	Comment
ADHOC	1	The aggregation value was generated because of an ad-hoc implementation dependent reason.
PERIODIC	2	The aggregation value was generated because of a periodic report.
FILTERED_TIMEOUT	3	The item is filtered but it exceeded its timeout value.

4.7.4   COMPOSITE: AGGREGATIONDEFINITIONDETAILS

The AggregationDefinitionDetails structure shall be used to hold definition details of an aggregation.

Name	AggregationDefinitionDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
description	MAL::String	No	The description of the parameter. May be empty.
category	MAL::UOctet	No	Category of the aggregation. Value taken from AggregationCategory enumeration, although the use of a UOctet allows deployment specific extension. Extensions must use values greater than 127.
reportInterval	MAL::Duration	No	The interval between periodic reports on this aggregation. If this aggregation is not periodic, this field must be '0'.
sendUnchanged	MAL::Boolean	No	If TRUE reports will include all values regardless of whether changed, if FALSE values unchanged from previous report are replaced with a NULL.
sendDefinitions	MAL::Boolean	No	If TRUE reports will include the ParameterDefinition object instance identifier in the AggregationParameterValue, if FALSE it will be set to NULL.
filterEnabled	MAL::Boolean	No	Controls whether reports for this aggregation are to be filtered.
filteredTimeout	MAL::Duration	No	The maximum duration between filtered reports. If this value is exceeded, then a report is sent regardless of filtered thresholds. Ignored if not filtered.
generationEnabled	MAL::Boolean	No	Controls whether reports for this aggregation are to be generated.
parameterSets	List<AggregationParameterSet>	No	List containing the parameter sets which define the aggregation.

4.7.5   COMPOSITE: AGGREGATIONPARAMETERSET

The AggregationParameterSet structure shall be used to hold the identifier and optional filter for a parameter, or set of parameters, in an aggregation.

Name	AggregationParameterSet		
Extends	MAL::Composite		
Short Form Part	2		
Field	Type	Nullable	Comment
domain	List<MAL::Identifier>	Yes	The domain of the parameters being referenced in this set of parameters, NULL if the same domain as the aggregation.
parameters	List<MAL::Long>	No	The list of object instance identifiers of the ParameterIdentity objects being included in the aggregation.
sampleInterval	MAL::Duration	No	The interval between samples of the parameters in the set. If '0' then just a single sample of the parameters is required per aggregation report.
reportFilter	ThresholdFilter	Yes	If the AggregationParameterSet contains a single parameter then this field contains the filter to apply for filtered reports when filters are applied. NULL if no filter required or this set contains more than one parameter.

4.7.6   COMPOSITE: AGGREGATIONVALUE

4.7.6.1   The AggregationValue structure shall be used to hold the values for one or more sets of parameter values.

4.7.6.2   The value sets must be held in the same order as that defined in the matching AggregationDefinitionDetails.

Name	AggregationValue		
Extends	MAL::Composite		
Short Form Part	3		
Field	Type	Nullable	Comment
generationMode	GenerationMode	No	Reason for the aggregation being generated.
filtered	MAL::Boolean	No	If a filter is enabled when the aggregation value is generated then this value shall be set to TRUE, else FALSE.
parameterSetValues	List<AggregationSetValue>	No	The parameterSetValues list holds the sets of values of the aggregation. The sets must be held in the same order as that defined in the aggregation definition.

4.7.7   COMPOSITE: AGGREGATIONSETVALUE

4.7.7.1   The AggregationSetValue structure shall be used to hold the values for one set of parameter values.

4.7.7.2   If the definition sendUnchanged field is set to FALSE, parameter values that are unchanged since the previous report shall be replaced by a NULL in this list.

4.7.7.3   The parameter values must be held in the same order as that defined in the matching AggregationDefinitionDetails.

Name	AggregationSetValue		
Extends	MAL::Composite		
Short Form Part	4		
Field	Type	Nullable	Comment
deltaTime	MAL::Duration	Yes	Optional delta time, from the timestamp of the aggregation for the first parameter set of the aggregation or the last value of the previous parameter set; otherwise, for the first parameter sample of this set. If NULL, then the first sample time is the same as the aggregation timestamp for the first parameter set of the aggregation, or the last value of the previous parameter set otherwise.
intervalTime	MAL::Duration	Yes	Optional delta time between samples in this set. If NULL, then all samples in this set are given the same time. This is usually driven by the sampleInterval in the aggregation set definition.
values	List<AggregationParameterValue>	No	List containing values of the parameters which are part of the aggregation. The ordering of the list entries shall match that of the definition of the aggregation. If there are more values than contained in the definition then it is assumed that the parameters cycle as a complete parameter set.

4.7.8 COMPOSITE: AGGREGATIONPARAMETERVALUE

The AggregationParameterValue structure shall be used to hold a single parameter value with its definition instance identifier.

Name	AggregationParameterValue		
Extends	MAL::Composite		
Short Form Part	5		
Field	Type	Nullable	Comment
value	Parameter::ParameterValue	No	The parameter value.
paramDefInstId	MAL::Long	Yes	The object instance identifier of the ParameterDefinition. NULL if sendDefinitions in the AggregationDefinitionDetails is FALSE.

4.7.9 COMPOSITE: THRESHOLDFILTER

The ThresholdFilter structure shall be used to hold the filter for a parameter.

Name	ThresholdFilter		
Extends	MAL::Composite		
Short Form Part	6		
Field	Type	Nullable	Comment
thresholdType	ThresholdType	No	The type of filter to apply for filtered periodic reports when filters are applied.
thresholdValue	MAL::Attribute	No	Threshold value to apply.
useConverted	MAL::Boolean	No	If true, and the relevant Parameter has a conversion, then the converted value for the threshold comparison is to be used; otherwise the raw value is to be used.

4.7.10 COMPOSITE: AGGREGATIONCREATIONREQUEST

The AggregationCreationRequest structure shall be used to contain all the fields required when creating a new aggregation in a provider.

Name	AggregationCreationRequest		
Extends	MAL::Composite		
Short Form Part	10		
Field	Type	Nullable	Comment
name	MAL::Identifier	No	The name of the aggregation. Must not be empty or the wildcard value.
aggDefDetails	AggregationDefinitionDetails	No	The aggregation definition details.

4.7.11 COMPOSITE: AGGREGATIONVALUEDETAILS

The AggregationValueDetails structure shall be used to hold a specific time stamped value of the aggregation.

Name	AggregationValueDetails		
Extends	MAL::Composite		
Short Form Part	11		
Field	Type	Nullable	Comment
aggId	MAL::Long	No	The AggregationIdentity object instance identifier.
defId	MAL::Long	No	The AggregationDefinition object instance identifier.
timestamp	MAL::Time	No	The timestamp of the value. Use for the calculation of the individual parameter value timestamps.
value	AggregationValue	No	The aggregation value.

4.8 SERVICE DATA TYPES: CONVERSION

4.8.1 COMPOSITE: DISCRETECONVERSIONDETAILS

4.8.1.1 The DiscreteConversionDetails structure shall be used to hold a bidirectional conversion between raw and converted values.

4.8.1.2 The first element of the pair shall be the raw value, and the second shall be the converted value.

4.8.1.3 Both sets of values must be unique.

Name	DiscreteConversionDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
mapping	List<MAL::Pair>	No	Defines a mapping between raw and converted values as a discrete set of points. The first entry in the pair is the raw value, and the second entry is the converted value.

4.8.2 COMPOSITE: LINECONVERSIONDETAILS

The LineConversionDetails structure shall be used for a bi-directional conversion between raw and converted values.

NOTE – It is defined by a series of points between which values are to be interpolated. The extrapolate attribute indicates if values can also be linearly extrapolated beyond the initial and final points.

Name	LineConversionDetails		
Extends	MAL::Composite		
Short Form Part	2		
Field	Type	Nullable	Comment
extrapolate	MAL::Boolean	No	Indicates whether or not values can be extrapolated beyond the start and the end of the points.
points	List<MAL::Pair>	No	Defines the bi-directional conversion. The first attribute of the point is a raw value, and the second attribute is the converted value.

4.8.3 COMPOSITE: POLYCONVERSIONDETAILS

The PolyConversionDetails structure shall be used to hold only forward (raw to converted) polynomial conversions.

NOTE – They are defined by a series of points for the polynomial coefficients.

Name	PolyConversionDetails		
Extends	MAL::Composite		
Short Form Part	3		
Field	Type	Nullable	Comment
points	List<MAL::Pair>	No	The first attribute of a point is a MAL::Integer, being the degree of the polynomial; the second attribute is either a MAL::Float or a MAL::Double, being the coefficient of the term.

4.8.4   COMPOSITE: RANGECONVERSIONDETAILS

4.8.4.1   The RangeConversionDetails structure shall be used to hold a range for a one-way conversion to convert between a continuous range to a discrete value.

4.8.4.2   A range shall be from this point up to, but not including, the next point.

Name	RangeConversionDetails		
Extends	MAL::Composite		
Short Form Part	4		
Field	Type	Nullable	Comment
points	List<MAL::Pair>	No	The first attribute in each pair is the raw range, and the second attribute is the converted value.

4.9   SERVICE DATA TYPES: GROUP—COMPOSITE: GROUPDETAILS

The GroupDetails structure shall be used to hold the object type, domain, and set of object instance identifiers for a set of objects from another service.

Name	GroupDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
description	MAL::String	No	Description of the group.
objectType	COM::ObjectType	No	The object type of the objects referenced by this group.
domain	List<MAL::Identifier>	No	The domain of the objects being referenced by this group.
instanceIds	List<MAL::Long>	No	The list of object instance identifiers of the objects being referenced by this group.

5 ERROR CODES

Errors codes defined in table 5-1 shall apply for this specification.

Table 5-1: MC Error Codes

Error	Error #	Comment
READONLY	70020	Operation specific
REFERENCED	70021	Operation specific

6   SERVICE SPECIFICATION XML

The use of XML for service specification provides a machine-readable format rather than the text-based document format. The service specification in XML notation as specified in reference [2] may be accessed at the URLs below.

The published specifications and XML schemas are held in an online SANA registry, located at the following URL:

<div align="center">

http://sanaregistry.org/r/moschemas/

</div>

The normative XML for this specification, validated against the XML schemas, is located at the following URL:

<div align="center">

http://sanaregistry.org/r/moschemas/ServiceDefMC-v.v.xml

</div>

where the 'v.v' part is replaced with the issue number of the corresponding document.

The latest version of any specification may always be directly addressed by removing the '-v.v' part from the URL; for example:

<div align="center">

http://sanaregistry.org/r/moschemas/ServiceDefMC.xml

</div>

ANNEX A

PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT PROFORMA

(NORMATIVE)

A1   INTRODUCTION

A1.1   OVERVIEW

This annex provides the Protocol Implementation Conformance Statement (PICS) Requirements List (PRL) for an implementation of the Mission Operations M&C Services standard.   The PICS for an implementation is generated by completing the PRL in accordance with the instructions below. An implementation claiming conformance must satisfy the mandatory requirements referenced in the PRL.

An implementation's completed PRL 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 PRL 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 PRL

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

A3 MO M&C SERVICES PICS

Item	Protocol Feature	Reference	Status	Support
1-1	Action service	3.2 and 4.2	O	
1-2	Parameter Service	3.3 and 4.3	O	
1-3	Alert Service	3.4 and 4.4	O	
1-4	Check Service	3.5 and 4.5	O	
1-5	Statistics Service	3.6 and 4.6	O	
1-6	Aggregation Service	3.7 and 4.7	O	
1-7	Conversion Service	3.8 and 4.8	O	
1-8	Group Service	3.9 and 4.9	O	

ANNEX B

SECURITY, SANA, AND PATENT CONSIDERATIONS

(INFORMATIVE)

B1   SECURITY CONSIDERATIONS

The security considerations of this specification are the same as those of reference [2]. Specifically, authentication and authorisation of a participating consumer or provider is provided by the MAL access control concept and is covered in subsections 3.6, 5.2, and 5.3 of the Reference Model (reference [1]).

Security of a communications link is delegated to the transport layer.

B2   SANA CONSIDERATIONS

The recommendations of this document request SANA populate the registry specified in reference [2] with the schema and XML detailed in section 6 of this document.

As stated in reference [2], the registration rule for change to this registry requires an engineering review by a designated expert. The expert shall be assigned by the WG Chair, or in absence, Area Director.

B3   PATENT CONSIDERATIONS

The recommendations of this document have no patent issues.

ANNEX C

DEFINITION OF ACRONYMS

(INFORMATIVE)

API	Application Program Interface
AMS	CCSDS Asynchronous Messaging System
CCS	Central Checkout System
CCSDS	Consultative Committee for Space Data Systems
COM	Common Object Model
MAL	Message Abstract Layer
M&C	Monitor and Control
MCS	Mission Control System
MO	Mission Operations
SDU	Service Data Units
SPP	[CCSDS] Space Packet Protocol
UML	Unified Modeling Language
XML	eXtensible Markup Language

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.

NOTE   –   Normative references are contained in 1.9.