# Messages

USP contains messages to create, read, update, and delete Objects, perform Object-defined operations, and allow agents to notify controllers of events. This is often referred to as CRUD with the addition of O (operate) and N (notify), or CRUD-ON.

*Note: This version of the specification defines its messages in Protocol Buffers v3 (see* [*encoding*](/specification/encoding/)*). This part of the specification may change to a more generic description (normative and non-normative) if further encodings are specified in future versions.*

These sections describe the types of USP messages and the normative requirements for their flow and operation. USP messages are described in a protocol buffers schema, and the normative requirements for the individual elements of the schema are outlined below:

* [Add](#add)
* [Set](#set)
* [Delete](#delete)
* [Get](#get)
* [GetInstances](#getinstances)
* [GetSupportedDM](#getsupporteddm)
* [Notify](#notify)
* [Operate](#operate)

## Requests, Responses and Errors

The three types of USP messages are Request, Response, and Error.

A request is a message sent from a source USP endpoint to a target USP endpoint that includes elements to be processed and returns a response or error. Unless otherwise specified, all requests have an associated response. Though the majority of requests are made from a Controller to an Agent, the Notify message follows the same format as a request but is sent from an Agent to a Controller.

**R-MSG.0** - The target USP endpoint MUST respond to a request message from the source USP endpoint with either a response message or error message, unless otherwise specified (see Operate and Notify messages).

**R-MSG.1** - The target USP endpoint MUST ignore or send an error message in response to messages it does not understand.

**R-MSG.2** - When the target USP endpoint is not required to send a response, the MTP endpoint that received the message MUST gracefully end the MTP message exchange. How the MTP gracefully ends the MTP message exchange is dependent on the type of MTP.

### Handling Duplicate Messages

Circumstances may arise (such as multiple Message Transfer Protocols) that cause duplicate messages (those with an identical message ID) to arrive at the target USP endpoint.

**R-MSG.3** - If a target USP endpoint receives a message with a duplicate message ID before it has processed and sent a Response or Error to the original message, it MUST gracefully ignore the duplicate message.

For messages that require no response, it is up to the target endpoint implementation when to allow the same message ID to be re-used by the same source USP endpoint.

### Example Message Flows

Successful request/response: In this successful message sequence, a Controller sends an Agent a request. The message header and body are parsed, the message is understood, and the Agent sends a response with the relevant information in the body.

Figure MSG.1 - A successful request/response sequence

Failed request/response: In this failed message sequence, a Controller sends an Agent a request. The message header and body are parsed, but the Agent throws an error. The error arguments are generated and sent in an error message.

Figure MSG.2 - A failed request/response sequence

## Message Structure

A Message consists of a header and body. When using [protocol buffers](https://developers.google.com/protocol-buffers/docs/proto3), the elements of the header and body for different messages are defined in a schema and sent in an encoded format from one USP endpoint to another.

**R-MSG.4** - A Message MUST conform to the schemas defined in [usp-msg.proto](/specification/usp-msg.proto).

*Note: When using protocol buffers for message encoding, default values (when elements are missing) are described in* [*Protocol Buffers v3*](https://developers.google.com/protocol-buffers/docs/proto3#default)*.*

Every USP message contains a header and a body. The header contains basic destination and coordination information, and is separated to allow security and discovery mechanisms to operate. The body contains the message itself and its arguments.

Each of the message types and elements below are described with the element type according to [Protocol Buffers version 3](https://developers.google.com/protocol-buffers/docs/proto3), followed by its name.

### The USP Message

Header header

**R-MSG.5** - A Message MUST contain exactly one header element.

Body body

The Message Body that must be present in every Message. The Body element contains either a Request, Response, or Error element.

**R-MSG.6** - A Message MUST contain exactly one body element.

### Message Header

The message header contains information on source and target of the message, as well as useful coordination information. Its elements include a message ID, the endpoint identifiers for the source and target endpoints, an optional reply-to identifier, and a field indicating the type of message.

The purpose of the message header is to provide basic information necessary for the target endpoint to process the message.

#### Message Header Elements

string msg\_id

A locally unique opaque identifier assigned by the Endpoint that generated this message.

**R-MSG.7** - The msg\_id element MUST be present in every Header.

**R-MSG.8** - The msg\_id element in the Message Header for a Response or Error that is associated with a Request MUST contain the message ID of the associated request. If the msg\_id element in the Response or Error does not contain the message ID of the associated Request, the response or error MUST be ignored.

enum MsgType msg\_type

This element contains an enumeration indicating the type of message contained in the message body. It is an enumeration of:

ERROR (0)  
GET (1)  
GET\_RESP = (2)  
NOTIFY = (3)  
SET = (4)  
SET\_RESP = (5)  
OPERATE = (6)  
OPERATE\_RESP = (7)  
ADD = (8)  
ADD\_RESP = (9)  
DELETE = (10)  
DELETE\_RESP = (11)  
GET\_SUPPORTED\_DM = (12)  
GET\_SUPPORTED\_DM\_RESP = (13)  
GET\_INSTANCES = (14)  
GET\_INSTANCES\_RESP = (15)  
NOTIFY\_RESP = (16)

**R-MSG.9** - The msg\_type element MUST be present in every Header.

string proto\_version

The version of the USP protocol.

**R-MSG.10** - The proto\_version element MUST be present in every Header.

**R-MSG.11** - The proto\_version element MUST be set to a value of "1.0".

string to\_id

The value of this header argument is the Endpoint Identifier of the target Endpoint.

**R-MSG.12** - The to\_id element MUST be present in every Header.

**R-MSG.13** - The target USP endpoint MUST ignore any message that does not contain its own Endpoint Identifier.

string from\_id

The value of this header argument is the Endpoint Identifier of the source Endpoint.

**R-MSG.14** - The from\_id element MUST be present in every Header.

string reply\_to\_id

The value of this header argument is the Endpoint Identifier to which responses associated with this message should be targeted.

**R-MSG.15** - The reply\_to\_id MAY be used to send Response or Error to a USP Endpoint other than the from-to-id in the Request.

*Note: The reply-to endpoint should have prior knowledge of the message and can expect the Response or Error.*

**R-MSG.16** - The source Endpoint MUST ignore a Response or Error message from a target Endpoint when the source Endpoint does not expect the Response or Error.

*Note: The reply-to endpoint should have prior knowledge of the message and can expect the Response.*

**R-MSG.17** - The reply\_to\_id element is optional, and MUST NOT be present in a Message that contains either a Response or Error in the Body element.

**R-MSG.18** - If the reply\_to\_id is omitted from a Message that contains a Request in the Body element, the response or Error MUST be sent to the Endpoint identified in the from\_id element in the request’s Header element.

### Message Body

The message body contains the intended message and the appropriate elements for the message type.

Every message body contains exactly one message and its elements. When an Agent is the target endpoint, these messages can be used to create, read, update, and delete Objects, or execute Object-defined operations. When a Controller is the target endpoint, the message will contain a notification, response, or an error.

#### Message Body Elements

oneof msg\_body

This element contains one of the types given below.

Request request

This element indicates that the Message contains a request of a type given in the Request Message.

Response response

This element indicates that the Message contains a response of a type given in the Response Message.

Error error

This element indicates that the Message contains an Error Message.

#### Request Elements

oneof req\_type

This element contains one of the types given below. Each indicates that the Message contains a Message of the given type.

Get get  
GetObjects get\_Objects  
Set set  
Add add  
Delete delete  
Operate operate  
Notify notify

#### Response Elements

oneof resp\_type

This element contains one of the types given below. Each indicates that the Message contains a Message of the given type.

GetResp get\_resp  
GetObjectsResp get\_objects\_resp  
SetResp set\_resp  
AddResp add\_resp   
DeleteResp delete\_resp   
OperateResp operate\_resp  
NotifyResp notify\_resp

#### Error Elements

fixed32 err\_code

This element contains a [numeric code](#error-codes) indicating the type of error that caused the overall message to fail.

string err\_msg

This element contains additional information about the reason behind the error.

repeated ParamError param\_err\_list

This element is present in an Error Message in response to an Add or Set message when the allow\_partial element is false and detailed error information is available for each Object or parameter that have caused the message to report an Error.

##### ParamError Elements

string param\_path

This element contains a Path Name to the Object or parameter that caused the error.

**R-MSG.19** - Path Names containing Object Instances in the param\_path element of ParamError MUST be addressed using Instance Number Addressing.

fixed32 err\_code

This element contains a [numeric code](#error-codes) indicating the type of error that caused the message to fail.

string err\_msg

This element contains additional information about the reason behind the error.

## Creating, Updating, and Deleting Objects

The [Add](#add), [Set](#set), and [Delete](#delete) requests are used to create, configure and remove Objects that comprise Service Elements.

### Selecting Objects and Parameters for CUD Operations

Each Add, Set, and Delete request operates on one or more paths. For the Add request, these paths are references to Multi-Instance Objects. For all other requests, these paths can contain either addressing based identifiers that match zero or one Object or search based identifiers that matches one or more Objects.

For Add and Set requests, each Object address or search is conveyed in an element that also contains a sub-element listing the parameters to update in the matched Objects.

The Add response contains details about the success or failure of the creation of the Object and the parameters set during its creation. In addition, it also returns those parameters that were set by the Agent upon creation of the Object.

For example, a Controller wants to create a new Wifi network on an Agent. It could use an Add message with the following elements:

allow\_partial: false  
create\_obj\_list {  
 obj\_path: Device.Wifi.SSID.  
 param\_setting\_list {  
  
 param: LowerLayers  
 value: Device.Wifi.Radio.1.  
 required: True  
  
 param: SSID  
 value: NewSSIDName  
 required: True  
 }  
 }

The Agent’s response would include the successful Object update and the list of parameters that were set, including the default values for the Enable and Status parameters defined in [Device:2](https://github.com/BroadbandForum/usp/tree/master/data-model):

created\_obj\_result\_list {  
 requested\_path: Device.Wifi.SSID.  
 oper\_status {  
 oper\_success {  
 instantiated\_path: Device.Wifi.SSID.2.  
 created\_param\_result\_map:  
  
 key: Enable  
 value: false  
  
 key: Status  
 value: Down  
  
 key: LowerLayers  
 value: : Device.Wifi.Radio.1.  
  
 key: SSID  
 value: NewSSIDName: NewSSIDName   
 }  
}

### Using Allow Partial and Required Parameters

The Add, Set, and Delete requests contain an element called "allow\_partial". This element determines whether or not the message should be treated as one complete configuration change, or a set of individual changes, with regards to the success or failure of that configuration.

For Delete, this is straightforward - if allow\_partial is true, the Agent should return a Response message with affected\_path\_list and unaffected\_path\_err\_list containing the successfully deleted Objects and unsuccessfully deleted objects, respectively. If allow\_partial is false, the Agent should return an Error message if any Objects fail to be deleted.

For the Add and Set messages, parameter updates contain an element called "required". This details whether or not the update or creation of the Object should fail if a required parameter fails.

This creates a hierarchy of error conditions for the Add and Set requests, such as:

Parameter Error -> Object Error -> Message Error

If allow\_partial is true, but one or more required parameters fail to be updated or configured, the creation or update of an individual Object fails. This results in an oper\_failure in the oper\_status element and updated\_obj\_result or created\_obj\_result returned in the Add or Set response.

If allow\_partial is false, the failure of any required parameters will cause the update or creation of the Object to fail, which will cause the entire message to fail. In this case, the Agent returns an error message rather than a response message.

Both the oper\_failure elements and Error messages contain an element called param\_error, which contains elements of type ParamError. This is so that the Controller will receive the details of failed parameter updates regardless of whether or not the Agent returned a response message or error message.

The logic can be described as follows:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| allow\_partial | Required Parameters | Required Parameter Failed | Other Parameter Failed | Response/Error | Oper\_status of Object | Contains param\_error |
| True/False | No | - | No | Response | oper\_success | No |
| True/False | No | - | Yes | Response | oper\_success | Yes |
| True/False | Yes | No | No | Response | oper\_success | No |
| True/False | Yes | No | Yes | Response | oper\_success | Yes |
| True | Yes | Yes | - | Response | oper\_failure | Yes |
| False | Yes | Yes | - | Error | oper\_failure | Yes |

### The Add Message

The Add message is used to create new Instances of Multi-Instance Objects in the Agent's Instantiated Data Model.

#### Add Example

In this example, the Controller requests that the Agent create a new instance in the Device.LocalAgent.Controller table.

Add Request:  
header {  
 msg\_id: "52867"  
 msg\_type: ADD  
 proto\_version: "1.0"  
 to\_id: "oui:112233:agent"  
 from\_id: "oui:112233:controller"  
}  
body {  
 request {  
 add {  
 allow\_partial: true  
 create\_obj\_list {  
 obj\_path: "Device.LocalAgent.Controller."  
 param\_setting\_list {  
 param: "Enable"  
 value: "True"  
  
 param: "EndpointID"  
 value: "controller-temp"  
 }  
 }  
 }  
 }  
}  
  
Add Response:  
header {  
 msg\_id: "55362"  
 msg\_type: ADD\_RESP  
 proto\_version: "1.0"  
 to\_id: “id:oui:112233:controller”  
 from\_id: “id:oui:112233:agent”  
}  
body {  
 response {  
 add\_resp {  
 created\_obj\_result\_list {  
 requested\_path: "Device.LocalAgent.Controller."  
 oper\_status {  
 oper\_success {  
 instantiated\_path: "Device.LocalAgent.Controller.31185."  
 unique\_key\_map {  
 key: "EndpointID"  
 value: "controller-temp"  
 }  
 }  
 }  
 }  
 }  
 }  
}

#### Add Request Elements

bool allow\_partial

This element tells the Agent how to process the message in the event that one or more of the Objects specified in the create\_obj\_list argument fails creation.

**R-ADD.0** - If the allow\_partial element is set to true, and no other exceptions are encountered, the Agent treats each Object matched in obj\_path independently. The Agent MUST complete the creation of valid Objects regardless of the inability to create or update one or more Objects (see [allow partial and required parameters](#allow_partial_and_required_parameters)).

**R-ADD.1** - If the allow\_partial element is set to false, and no other exceptions are encountered, the Agent treats each Object matched in obj\_path holistically. A failure to create any one Object MUST cause the Add message to fail and return an Error Message (see [allow partial and required parameters](#allow_partial_and_required_parameters)).

repeated CreateObject create\_obj\_list

This element contains a repeated set of CreateObject elements.

##### CreateObject Elements

string obj\_path

This element contains an Object Path to a writeable Table in the Agent’s Instantiated Data Model.

**R-ADD.2** - The obj\_path element in the CreateObject message of an Add Request MUST NOT contain Search Paths.

repeated CreateParamSetting param\_setting\_list

This element contains a repeated set of CreateParamSetting elements.

###### CreateParamSetting Elements

string param

This element contains a relative path to a parameter of the Object specified in obj\_path, or a parameter of a single instance sub-object of the Object specified in obj\_path.

string value

This element contains the value of the parameter specified in the param element that the Controller would like to configure as part of the creation of this Object.

bool required

This element specifies whether the Agent should treat the creation of the Object specified in obj\_path as conditional upon the successful configuration of this parameter (see [allow partial and required parameters](#allow_partial_and_required_parameters)).

**R-ADD.3** - If the required element is set to true, a failure to update this parameter MUST result in a failure to create the Object.

#### Add Response Elements

repeated CreatedObjectResult created\_obj\_result\_list

A repeated set of CreatedObjectResult elements for each CreateObject element in the Add message.

##### CreatedObjectResult Elements

string requested\_path

This element returns the value of obj\_path\_list in the CreateObject message associated with this CreatedObjectResult.

OperationStatus oper\_status

The element contains a message of type OperationStatus that specifies the overall status for the creation of the Object specified in requested\_path.

###### OperationStatus Elements

oneof oper\_status

This element contains one of the types given below. Each indicates that the element contains a message of the given type.

OperationFailure oper\_failure

This message is used when the object given in requested\_path failed to be created.

OperationSuccess oper\_success

###### OperationFailure Elements

fixed32 err\_code

This element contains a [numeric code](#error-codes) indicating the type of error that caused the Object creation to fail. A value of 0 indicates the Object was created successfully.

string err\_msg

This element contains additional information about the reason behind the error.

###### Operation Success Elements

string instantiated\_path

This element contains the Object Instance Path (using Instance Number Addressing) of the created Object.

repeated ParameterError param\_err\_list

This element returns a repeated set of ParameterError messages.

**R-ADD.4** - If any of the parameters and values specified in the param\_setting\_list element fail to configure upon creation, this set MUST include one element describing each of the failed parameters and the reason for their failure.

map<string, string> unique\_key\_map

This element contains a map of the local name and value for each supported parameter that is part of any of this Object's unique keys.

**R-ADD.5** - If the Controller did not include some or all of a unique key that the Agent supports in the param\_setting\_list element, the Agent MUST assign values to the unique key(s) and return them in the unique\_key\_map.

**R-ADD.6** - If the Controller does not have Read permission on any of the parameters specified in unique\_key\_map, these parameters MUST NOT be returned in this element.

###### ParameterError Elements

string param

This element contains the Relative Parameter Path to the parameter that failed to be set.

fixed32 err\_code

This element contains the [error code](#error-codes) of the error that caused the parameter set to fail.

string err\_msg

This element contains text related to the error specified by err\_code.

#### Add Message Supported Error Codes

Appropriate error codes for the Add message include 7000-7019 and 7800-7999.

### Set

The Set Message is used to update the Parameters of existing Objects in the Agent's Instantiated Data Model.

#### Set Example

In this example the Controller requests that the Agent change the value of the FriendlyName Parameter in the Device.DeviceInfo. Object.

Set Request:  
header {  
 msg\_id: "19220"  
 msg\_type: SET  
 proto\_version: "1.0"  
 to\_id: “oui:112233:agent”  
 from\_id: “oui:112233:controller”  
}  
body {  
 request {  
 set {  
 allow\_partial: true  
 update\_obj\_list {  
 obj\_path: "Device.DeviceInfo."  
 param\_setting\_list {  
 param: "FriendlyName"  
 value: "MyDevicesFriendlyName"  
 required: true  
 }  
 }  
 }  
 }  
  
Set Response:  
header {  
 msg\_id: "19220"  
 msg\_type: SET\_RESP  
 proto\_version: "1.0"  
 to\_id: “oui:112233:controller”  
 from\_id: “oui:112233:agent”  
}  
body {  
 response {  
 set\_resp {  
 updated\_obj\_result\_list {  
 requested\_path: "Device.DeviceInfo."  
 oper\_status {  
 oper\_success {  
 updated\_inst\_result\_list {  
 affected\_path: "Device.DeviceInfo."  
 updated\_param\_map {  
 key: "FriendlyName"  
 value: "MyDevicesFriendlyName"  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
}

#### Set Request Elements

bool allow\_partial

This element tells the Agent how to process the message in the event that one or more of the Objects matched in the obj\_path fails to update.

**R-SET.0** - If the allow\_partial element is set to true, and no other exceptions are encountered, the Agent treats each UpdateObject message obj\_path independently. The Agent MUST complete the update of valid Objects regardless of the inability to update one or more Objects (see [allow partial and required parameters](#allow_partial_and_required_parameters)).

*Note: This may cause some counterintuitive behavior if there are no required parameters to be updated. The Set Request can still result in a Set Response (rather than an Error Message) if allow\_partial is set to true.*

**R-SET.1** - If the allow\_partial element is set to false, and no other exceptions are encountered, the Agent treats each UpdateObject message obj\_path holistically. A failure to update any one Object MUST cause the Set message to fail and return an Error message (see [allow partial and required parameters](#allow_partial_and_required_parameters)).

repeated UpdateObject update\_obj\_list

This element contains a repeated set of UpdateObject messages.

##### UpdateObject Elements

string obj\_path

This element contains an Object Path, Instance Path, or Search Path to Objects or Object Instances in the Agent’s Instantiated Data Model.

repeated UpdateParamSetting param\_setting\_list

The element contains a repeated set of UpdatedParamSetting messages.

###### UpdateParamSetting Elements

string param

This element contains the local name of a parameter of the Object specified in obj\_path.

string value

This element contains the value of the parameter specified in the param element that the Controller would like to configure.

bool required

This element specifies whether the Agent should treat the update of the Object specified in obj\_path as conditional upon the successful configuration of this parameter.

**R-SET.2** - If the required element is set to true, a failure to update this parameter MUST result in a failure to update the Object (see [allow partial and required parameters](#allow_partial_and_required_parameters)).

#### Set Response

repeated UpdatedObjectResult updated\_obj\_result\_list

This element contains a repeated set of UpdatedObjectResult messages for each UpdateObject message in the associated Set Request.

##### UpdatedObjectResult Elements

string requested\_path

This element returns the value of updated\_obj\_result\_list in the UpdateObject message associated with this UpdatedObjectResult.

OperationStatus oper\_status

The element contains a message of type OperationStatus that specifies the overall status for the update of the Object specified in requested\_path.

###### OperationStatus Elements

oneof oper\_status

This element contains a message of one of the following types.

OperationFailure oper\_failure

Used when the Object specified in requested\_path failed to be updated.

OperationSuccess oper\_success

###### OperationFailure Elements

fixed32 err\_code

This element contains a [numeric code](#error-codes) indicating the type of error that caused the Object update to fail.

string err\_msg

This element contains additional information about the reason behind the error.

repeated UpdatedInstanceFailure updated\_inst\_failure\_list

This element contains a repeated set of messages of type UpdatedInstanceFailure.

###### UpdatedInstanceFailure Elements

string affected\_path

This element returns the Object Path or Object Instance Path of the Object that failed to update.

repeated ParameterError param\_err\_list

This element contains a repeated set of ParameterError messages.

###### ParameterError Elements

string param

This element contains the Parameter Path (relative to affected\_path) to the parameter that failed to update.

###### OperationSuccess Elements

repeated UpdatedInstanceResult updated\_inst\_result\_list

This element contains a repeated set of UpdatedInstanceResult messages.

###### UpdatedInstanceResult Elements

string affected\_path

This element returns the Object Path or Object Instance Path (using Instance Number Addressing) of the updated Object.

repeated ParameterError param\_err\_list

This element contains a repeated set of ParameterError messages.

map<string, string> updated\_param\_map

This element returns a set of key/value pairs containing a path (relative to the affected\_path) to each of the updated Object’s parameters, their values, plus sub-Objects and their values that were updated by the Set Request.

**R-SET.3** - If the Controller does not have Read permission on any of the parameters specified in updated\_param\_map, these parameters MUST NOT be returned in this element.

**R-SET.4** - Object Instance Paths in the keys of updated\_param\_map MUST use Instance Number Addressing.

*Note: If the Set Request configured a parameter to the same value it already had, this parameter is still returned in the updated\_param\_map.*

###### ParameterError Elements

string param

This element contains the Parameter Path to the parameter that failed to be set.

fixed32 err\_code

This element contains the [error code](#error-codes) of the error that caused the parameter set to fail.

string err\_msg

This element contains text related to the error specified by err\_code.

#### Set Message Supported Error Codes

Appropriate error codes for the Set message include 7000-7016, 7020, 7021, and 7800-7999.

### The Delete Message

The Delete Message is used to remove Instances of Multi-Instance Objects in the Agent's Instantiated Data Model.

#### Delete Example

In this example, the Controller requests that the Agent remove the instance in Device.LocalAgent.Controller table that has the EndpointID value of "controller-temp".

Delete Request:  
header {  
 msg\_id: "24799"  
 msg\_type: DELETE  
 proto\_version: "1.0"  
 to\_id: “oui:112233:agent”  
 from\_id: “oui:112233:controller”  
}  
body {  
 request {  
 delete {  
 obj\_path\_list: "Device.LocalAgent.Controller.[EndpointID==\"controller-temp\"]."  
 }  
 }  
}  
  
Delete Response:  
header {  
 msg\_id: "24799"  
 msg\_type: DELETE\_RESP  
 proto\_version: "1.0"  
 to\_id: “oui:112233:controller”  
 from\_id: “oui:112233:agent”  
}  
body {  
 response {  
 delete\_resp {  
 deleted\_obj\_result\_list {  
 requested\_path: "Device.LocalAgent.Controller.[EndpointID==\"controller-temp\"]."  
 oper\_status {  
 oper\_success {  
 affected\_path\_list: "Device.LocalAgent.Controller.31185."  
 affected\_path\_list: "Device.LocalAgent.Controller.31185.E2ESession."  
 }  
 }  
 }  
 }  
 }  
}

#### Delete Request Elements

bool allow\_partial

This element tells the Agent how to process the message in the event that one or more of the Objects specified in the obj\_path argument fails deletion.

**R-DEL.0** - If the allow\_partial element is set to true, and no other exceptions are encountered, the Agent treats each entry in obj\_path independently. The Agent MUST complete the deletion of valid Objects regardless of the inability to delete one or more Objects (see [allow partial and required parameters](#allow_partial_and_required_parameters)).

**R-DEL.1** - If the allow\_partial element is set to false, and no other exceptions are encountered, the Agent treats each entry in obj\_path holistically. A failure to delete any one Object MUST cause the Delete message to fail and return an Error message (see [allow partial and required parameters](#allow_partial_and_required_parameters)).

repeated string obj\_path\_list

This element contains a repeated set of Object Instance Paths or Search Paths.

#### Delete Response Elements

repeated DeletedObjectResult deleted\_obj\_result\_list

This element contains a repeated set of DeletedObjectResult messages.

##### DeletedObjectResult Elements

string requested\_path

This element returns the value of the entry of obj\_path\_list (in the Delete Request) associated with this DeleteObjectResult.

OperationStatus oper\_status

This element contains a message of type OperationStatus.

###### OperationStatus Elements

oneof oper\_status

This element contains a message of one of the following types.

OperationFailure oper\_failure

Used when the Object specified in requested\_path failed to be deleted.

OperationSuccess oper\_success

###### OperationFailure Elements

*Note: Since the OperationSuccess message of the Delete Response contains an unaffected\_path\_err\_list, the OperationStatus will only contain an OperationFailure message if the requested\_path was did not match any existing Objects (error 7016) or was syntactically incorrect (error 7008).*

fixed32 err\_code

This element contains a [numeric code](#error-codes) indicating the type of error that caused the delete to fail. A value of 0 indicates the Object was deleted successfully.

string err\_msg

This element contains additional information about the reason behind the error.

###### OperationSuccess Elements

repeated string affected\_path\_list

This element returns a repeated set of Path Names to Object Instances.

**R-DEL.2** - If the Controller does not have Read permission on any of the Objects specified in affected\_path\_list, these Objects MUST NOT be returned in this element.

**R-DEL.3** - The Path Names to Object Instances in affected\_path\_list MUST be addressed using Instance Number Addressing.

repeated UnaffectedPathError unaffected\_path\_err\_list

This element contains a repeated set of messages of type UnaffectedPathError.

**R-DEL.4** - If any of the Object Instances specified in the obj\_path\_list element fail to delete, this set MUST include one UnaffectedPathError message for each of the Object Instances that failed to Delete.

**R-DEL.5** - If the Controller does not have Read permission on any of the Objects specified in unaffected\_path\_list, these Objects MUST NOT be returned in this element.

###### UnaffectedPathError Elements

string unaffected\_path

This element returns the Path Name to the Object Instance that failed to be deleted.

**R-DEL.6** - The Path Names to Object Instances in unaffected\_path MUST be addressed using Instance Number Addressing.

fixed32 err\_code

This element contains the error code of the error that caused the deletion of this object to fail.

string err\_msg

This element contains text related to the error specified by err\_code.

#### Delete Message Supported Error Codes

Appropriate error codes for the Delete message include 7000-7008, 7015, 7016, 7018, 7024, and 7800-7999.

## Reading an Agent’s State and Capabilities

An Agent’s current state and capabilities are represented in its data model. The current state is referred to as its Instantiated Data Model, while the data model that represents its set of capabilities is referred to as its Supported Data Model. Messages exist to retrieve data from both the instantiated and Supported Data Models.

### The Get Message

The basic Get message is used to retrieve the values of a set of Object’s parameters in order to learn an Agent’s current state. It takes a set of search paths as an input and returns the complete tree of parameters, plus the parameters of all sub-Objects, of any Object matched by the specified expressions. The search paths specified in a Get request can also target individual parameters within Objects to be returned.

*Note: Those familiar with Broadband Forum* [*TR-069*](https://www.broadband-forum.org/technical/download/TR-069.pdf) *will recognize this behavior as the difference between "partial paths" and "complete paths". This behavior is replicated in USP for the Get message for each path that is matched by the expression(s) supplied in the request.*

*Note: Each search path is intended to be evaluated separately, and the results from a given search path are returned in an element dedicated to that path. As such, it is possible that the same information may be returned from more than one search path. This is intended, and the Agent should treat each search path atomically.*

The response returns an entry for each Path Name resolved by the path given in requested\_path. If a path expression specified in the request does not match any valid parameters or Objects, the response will indicate that this expression was an "invalid path", indicating that the Object or parameter does not currently exist in the Agent’s Instantiated Data Model.

For each resolved Path Name, a ResolvedPathResult message is given in the Response. This ResolvedPathResult contains the resolved\_path, followed by a list of parameters of both the resolved\_path Object and all of its sub-objects, plus their values. These Parameter Paths are Relative Paths to the resolved\_path.

#### Get Examples

For example, a Controller wants to read the data model to learn the settings and stats of a single Wifi SSID, "HomeNetwork" with a BSSID of 00:11:22:33:44:55. It could use a Get request with the following elements:

Get {  
 param\_path\_list {  
 Device.Wifi.SSID.[SSID="Homenetwork", BSSID=00:11:22:33:44:55].  
 }  
}

In response to this request the Agent returns all parameters, plus sub-Objects and their parameters, of the addressed instance. The Agent returns this data in the Get response using an element for each of the requested paths. In this case:

GetResp {  
 req\_path\_result\_list {  
 requested\_path: Device.Wifi.SSID.[SSID="Homenetwork",BSSID=00:11:22:33:44:55].  
 err\_code : 0  
 err\_msg :  
 resolved\_path\_result\_list {  
 resolved\_path : Device.Wifi.SSID.1.  
 result\_parm\_map {   
 key: Enable  
 value: True  
  
 key: Status  
 value: Up  
  
 key: Name  
 value: "Home Network"  
  
 key: LastChange  
 value: 864000  
  
 key: BSSID  
 value: 00:11:22:33:44:55  
  
 key: Stats.BytesSent  
 value: 24901567  
  
 key: Stats.BytesReceived  
 value: 892806908296  
  
 etc.  
 }  
 }  
 }

In another example, the Controller only wants to read the current status of the Wifi network with the SSID "HomeNetwork" with the BSSID of 00:11:22:33:44:55. It could use a Get request with the following elements:

Get {  
 param\_path\_list {  
 Device.Wifi.SSID.[SSID="Homenetwork",BSSID=00:11:22:33:44:55].Status  
 }  
}

In response to this request the Agent returns only the Status parameter and its value.

GetResp {  
 req\_path\_result\_list {  
 requested\_path: Device.Wifi.SSID.[SSID="Homenetwork",BSSID=00:11:22:33:44:55].Status  
 err\_code : 0  
 err\_msg :  
 resolved\_path\_result\_list {  
 resolved\_path : Device.Wifi.SSID.1.  
 result\_parm\_map {  
 key: Status  
 value: Up  
 }  
 }  
 }  
 }

Lastly, using wildcards or another Search Path, the requested path may resolve to more than one resolved path. For example for a Request sent to an Agent with two Wifi.SSID instances:

Get {  
 param\_path\_list {  
 Device.Wifi.SSID.\*.Status  
 }  
 }

The Agent's GetResponse would be:

GetResp {  
 req\_path\_result\_list {  
 requested\_path: Device.Wifi.SSID.\*.  
 err\_code : 0  
 err\_msg :  
 resolved\_path\_result\_list {  
 resolved\_path : Device.Wifi.SSID.1.  
 result\_parm\_map {  
 key: Status  
 value: Up  
 }  
  
 resolved\_path :Device.Wifi.SSID.2.  
 result\_param\_map {  
 key: Status  
 value: Up  
 }  
 }  
 }  
 }

In an example with full USP message header and body, the Controller requests all parameters of the MTP table entry that contains the Alias value "CoAP-MTP1", and the value of the Enable parameter of the Subscription table where the subscription ID is "boot-1" and the Recipient parameter has a value of "Device.LocalAgent.Controller.1":

Get Request:  
  
header {  
 msg\_id: "5721"  
 msg\_type: GET  
 proto\_version: "1.0"  
 to\_id: “oui:112233:agent”  
 from\_id: “oui:112233:controller”  
}  
body {  
 request {  
 get {  
 param\_path\_list: "Device.LocalAgent.MTP.[Alias==\"CoAP-MTP1\"]."  
 param\_path\_list: "Device.LocalAgent.Subscription.[ID==\"boot-1\",Recipient==\"Device.LocalAgent.Controller.1\"].Enable"  
 }  
 }  
}  
  
Get Response:  
header {  
 msg\_id: "5721"  
 msg\_type: GET\_RESP  
 proto\_version: "1.0"  
 to\_id: “oui:112233:controller”  
 from\_id: “oui:112233:agent”  
}  
body {  
 response {  
 get\_resp {  
 req\_path\_result\_list {  
 requested\_path: "Device.LocalAgent.MTP.[Alias==\"CoAP-MTP1\"]."  
 resolved\_path\_result\_list {  
 resolved\_path: "Device.LocalAgent.MTP.5156."  
  
 key: "Alias"  
 value: "CoAP-MTP1"  
  
 key: "Enable"  
 value: "False"  
  
 key: "EnableMDNS"  
 value: "True"  
  
 key: "Protocol"  
 value: "CoAP"  
  
 key: "Status"  
 value: "Inactive"  
 }  
  
 resolved\_path\_result\_list {  
 resolved\_path: "Device.LocalAgent.MTP.5156.XMPP."  
 result\_param\_map {  
 key: "Destination"  
  
 key: "Reference"  
  
 }  
 }  
 resolved\_path\_result\_list {  
 resolved\_path: "Device.LocalAgent.MTP.5156.HTTP."  
 result\_param\_map {  
 key: "CheckPeerID"  
  
 key: "EnableEncryption"  
  
 key: "Host"  
  
 key: "IsEncrypted"  
 value: "False"  
  
 key: "Path"  
  
 key: "Port"  
  
 key: "ValidatePeerCertificate"  
 }  
 }  
 resolved\_path\_result\_list {  
 resolved\_path: "Device.LocalAgent.MTP.5156.WS."  
 result\_param\_map {  
 key: "CheckPeerID"  
  
 key: "EnableEncryption"  
  
 key: "Host"  
  
 key: "IsEncrypted"  
 value: "False"  
  
 key: "Path"  
  
 key: "Port"  
  
 key: "ValidatePeerCertificate"  
 }  
 }  
 resolved\_path\_result\_list {  
 resolved\_path: "Device.LocalAgent.MTP.5156.CoAP."  
 result\_param\_map {  
 key: "CheckPeerID"  
 value: "False"  
  
 key: "EnableEncryption"  
 value: "True"  
  
 key: "Host"  
 value: "127.0.0.1"  
  
 key: "IsEncrypted"  
 value: "False"  
  
 key: "Path"  
 value: "/e/agent"  
  
 key: "Port"  
 value: "5684"  
  
 key: "ValidatePeerCertificate"  
 value: "True"  
 }  
 }  
 resolved\_path\_result\_list {  
 resolved\_path: "Device.LocalAgent.MTP.5156.STOMP."  
 result\_param\_map {  
 key: "Destination"  
  
 key: "Reference"  
 }  
 }  
 }  
 req\_path\_result\_list {  
 requested\_path: "Device.LocalAgent.Subscription.[ID==\"boot-1\",Recipient==\"Device.LocalAgent.Controller.1\"].Enable"  
 resolved\_path\_result\_list {  
 resolved\_path: "Device.LocalAgent.Subscription.6629."  
 result\_param\_map {  
 key: "Enable"  
 value: "True"  
 }  
 }  
 }  
 }  
 }  
}

#### Get Request Elements

repeated string param\_path\_list

This element is a set of Object Paths, Instance Paths, Parameter Paths, or Search Paths to Objects, Object Instances, and Parameters in an Agent’s Instantiated Data Model.

#### Get Response Elements

repeated RequestedPathResult req\_path\_result\_list

A repeated set of RequestedPathResult messages for each of the Path Names given in the associated Get request.

##### RequestedPathResult Element

string requested\_path

This element contains one of the Path Names or Search Paths given in the param\_path element of the associated Get Request.

fixed32 err\_code

This element contains a [numeric code](#error-codes/) indicating the type of error that caused the Get to fail on this path. A value of 0 indicates the path could be read successfully.

**R-GET.0** - If the Controller making the Request does not have Read permission on an Object or Parameter matched through the requested\_path element, the Object or Parameter MUST be treated as if it is not present in the Agent’s instantiated data model.

string err\_msg

This element contains additional information about the reason behind the error.

repeated ResolvedPathResult resolved\_path\_result\_list

This element contains one message of type ResolvedPathResult for each path resolved by the Path Name or Search Path given by requested\_path.

###### ResolvedPathResult Elements

string resolved\_path

This element contains a Path Name to an Object or Object Instance that was resolved from the Path Name or Search Path given in requested\_path.

**R-GET.1** - If the requested\_path included a Path Name to a Parameter, the resolved\_path MUST contain only the Path Name to the parent Object or Object Instance of that parameter.

map<string, string> result\_param\_map

This element contains a set of mapped key/value pairs listing a Parameter Path (relative to the Path Name in resolved\_path) to each of the parameters and their values, plus sub-objects and their values, of the Object given in resolved\_path.

**R-GET.2** - If the requested\_path included a Path Name to a Parameter, result\_param\_map MUST contain only the Parameter included in that path.

**R-GET.3** - If the Controller does not have Read permission on any of the parameters specified in result\_param\_map, these parameters MUST NOT be returned in this element. This MAY result in this element being empty.

**R-GET.4** - Path Names containing Object Instance Paths in the keys of result\_param\_map MUST be addressed using Instance Number Addressing.

###### Get Message Supported Error Codes

Appropriate error codes for the Get message include 7000-7006, 7008, 7010, and 7800-7999.

### The GetInstances Message

The GetInstances message takes a Path Name to an Object and requests that the Agent return the Instances of that Object that exist and *possibly* any Multi-Instance sub-Objects that exist as well as their Instances. This is used for getting a quick map of the Multi-Instance Objects (i.e., tables) the Agent currently represents, and their unique keys, so that they can be addressed and manipulated later.

GetInstances takes one or more Path Names to Multi-Instance Objects in a Request to an Agent. In addition, both GetInstances and GetSupportedDM (below) make use of a flag called first\_level\_only, which determines whether or not the Response should include all of the sub-Objects that are children of the Object specified in obj\_path. A value of true means that the Response should return data *only* for the Object specified. A value of false means that all sub-Objects should be resolved and returned.

#### GetInstances Examples

For example, if a Controller wanted to know *only* the current instances of Wifi SSID Objects that exist on an Agent (that has 3 SSIDs), it would send a GetInstances Request as:

GetInstances {  
 obj\_path\_list : Device.Wifi.SSID.  
 bool first\_level\_only : true  
 }

The Agent's Response would contain:

GetInstancesResp {  
 req\_path\_result\_list {  
 requested\_path : Device.Wifi.SSID.  
 err\_code : 0  
 err\_msg :  
 curr\_inst\_list {  
 instantiated\_obj\_path : Device.Wifi.SSID.1.  
 unique\_key\_map :  
  
 key : Alias  
 value : UserWifi1  
  
 key : Name  
 value : UserWifi1  
  
 key : SSID  
 value : SecureProviderWifi  
  
 key : BSSID  
 value : 00:11:22:33:44:55  
  
 instantiated\_obj\_path : Device.Wifi.SSID.2.  
 unique\_key\_map :  
  
 key : Alias  
 value : UserWifi2  
  
 key : Name  
 value : UserWifi2  
  
 key : SSID  
 value : GuestProviderWifi  
  
 key : BSSID  
 value : 00:11:22:33:44:55  
  
 }  
 }  
 }

In another example, the Controller wants to get all of the Instances of the Device.Wifi.AccessPoint table, plus all of the instances of the AssociatedDevice Object and AC Object (sub-Objects of AccessPoint). It would issue a GetInstances Request with the following:

GetInstances {  
 obj\_path\_list : Device.Wifi.AccessPoint.  
 bool first\_level\_only : false  
 }

The Agent's Response will contain an entry in curr\_inst\_list for all of the Instances of the Device.Wifi.AccessPoint table, plus the Instances of the Multi-Instance sub-Objects .AssociatedDevice. and .AC.:

GetInstancesResp {  
 req\_path\_result\_list {  
 requested\_path : Device.Wifi.AccessPoint.  
 err\_code : 0  
 err\_msg :  
 curr\_inst\_list {  
 instantiated\_obj\_path : Device.Wifi.AccessPoint.1.  
 unique\_key\_map :  
  
 key : Alias  
 value : SomeAlias  
  
 key : SSIDReference  
 value : Device.Wifi.SSID.1  
  
 instantiated\_obj\_path : Device.Wifi.AccessPoint.2.  
 unique\_key\_map :  
  
 key : Alias  
 value : SomeAlias  
  
 key : SSIDReference  
 value : Device.Wifi.SSID.2  
  
 instantiated\_obj\_path : Device.Wifi.AccessPoint.1.AssociatedDevice.1.  
 unique\_key\_map :  
  
 key : MACAddress  
 value : 11:22:33:44:55:66  
  
 instantiated\_obj\_path : Device.Wifi.AccessPoint.1.AC.1.  
 unique\_key\_map :  
  
 key : AccessCategory  
 value : BE  
  
 instantiated\_obj\_path : Device.Wifi.AccessPoint.2.AssociatedDevice.1.  
 unique\_key\_map :  
  
 key : MACAddress  
 value : 11:22:33:44:55:66  
  
 instantiated\_obj\_path : Device.Wifi.AccessPoint.2.AC.1.  
 unique\_key\_map :  
  
 key : AccessCategory  
 value : BE  
 }  
 }  
 }

Or more, if more Object Instances exist.

#### GetInstances Request Elements

repeated string obj\_path\_list

This element contains a repeated set of Path Names or Search Paths to Multi-Instance Objects in the Agent's Instantiated Data Model.

bool first\_level\_only

This element, if true, indicates that the Agent should return only those instances in the Object(s) matched by the Path Name or Search Path in obj\_path, and not return any child objects.

#### GetInstances Response Elements

repeated RequestedPathResult req\_path\_result\_list

This element contains a RequestedPathResult message for each Path Name or Search

string requested\_path

This element contains one of the Path Names or Search Paths given in obj\_path of the associated GetInstances Request.

fixed32 err\_code

This element contains a [numeric code](#error-codes) indicating the type of error that caused the Get to fail on this path. A value of 0 indicates the path could be read successfully.

**R-GIN.0** - If the Controller making the Request does not have Read permission on an Object or Parameter matched through the requested\_path element, the Object or Parameter MUST be treated as if it is not present in the Agent’s instantiated data model.

string err\_msg

This element contains additional information about the reason behind the error.

repeated CurrInstance curr\_inst\_list

This element contains a message of type CurrInstance for each Instance of *all* of the Objects matched by requested\_path that exists in the Agent's Instantiated Data Model.

###### CurrInstance Elements

string instantiated\_obj\_path

This element contains the Instance Path Name of the Object Instance.

map<string, string> unique\_key\_map

This element contains a map of key/value pairs for all supported parameters that are part of any of this Object's unique keys.

**R-GIN.1** - If the Controller does not have Read permission on any of the parameters specified in unique\_key\_map, these parameters MUST NOT be returned in this element.

#### GetInstances Error Codes

Appropriate error codes for the GetInstances message include 7000-7006, 7008, 7016, 7018 and 7800-7999.

### The GetSupportedDM Message

GetSupportedDM is used to retrieve the Objects, Parameters, Events, and Commands in the Agent's Supported Data Model. This allows a Controller to learn what an Agent understands, rather than its current state.

The GetSupportedDM is different from other USP messages in that it deals exclusively with the Agent's Supported Data Model. This means that Path Names to Multi-Instance Objects only address the Object itself, rather than Instances of the Object, and those Path Names that contain Multi-Instance objects in the Path use the {i} identifier to indicate their place in the Path Name.

For example, a Path Name to the AssociatedDevice Object (a child of the .Wifi.AccessPoint Object) would be addressed in the Supported Data Model as:

Device.Wifi.AccessPoint.{i}.AssociatedDevice. or Device.Wifi.AccessPoint.{i}.AssociatedDevice.{i}.

Both of these syntaxes are supported and equivalent. The Agent's Response returns the Path Name to the Object in the associated Device Type document as specified in [TR-106](https://www.broadband-forum.org/technical/download/TR-106_Amendment-8.pdf).

#### GetSupportedDM Examples

For example, the Controller wishes to learn the Wifi capabilities the Agent represents. It could issue a GetSupportedDM Request as:

GetSupportedDM {  
 first\_level\_only : false  
 return\_commands : true  
 return\_events : true  
 return\_params : true  
 discover\_obj\_list {  
 obj\_path : Device.Wifi.  
 }  
 }

The Agent's Response would be:

GetSupportedDMResp {  
 req\_obj\_result\_list {  
 req\_obj\_path : Device.Wifi.  
 err\_code : 0  
 err\_msg :  
 data\_model\_inst\_uri : urn:broadband-forum-org:tr-181-2-12-0  
 supported\_obj\_list {  
 supported\_obj\_path : Device.Wifi.  
 is\_multi\_instance : false  
 supported\_param\_list {  
 param\_name : RadioNumberOfEntries   
  
 param\_name : SSIDNumberOfEntries   
  
 param\_name : AccessPointNumberOfEntries   
  
 param\_name : EndPointNumberOfEntries   
 }  
 supported\_command\_list {  
 command\_name : SomeCommand()  
 input\_arg\_name\_list {  
 SomeArgument1  
 SomeArgument2  
 }  
 output\_arg\_name\_list {  
 SomeArgument1  
 SomeArgument2  
 }  
 }  
 supported\_event\_list {  
 event\_name : SomeEvent!  
 arg\_name\_list {  
 SomeArgumentA  
 SomeArgumentB  
 }  
 }  
 supported\_obj\_path : Device.Wifi.SSID.{i}.  
 access : ADD\_DELETE (1)  
 is\_multi\_instance : true  
 supported\_param\_list {  
 param\_name : Enable  
 access : PARAM\_READ\_WRITE (1)  
  
 param\_name: Status   
  
 param\_name : Alias  
 access : PARAM\_READ\_WRITE (1)  
  
 param\_name : Name  
  
 param\_name: LastChange   
  
 param\_name : LowerLayers  
 access : PARAM\_READ\_WRITE (1)  
  
 param\_name : BSSID   
  
 param\_name : MACAddress   
  
 param\_name : SSID  
 access : PARAM\_READ\_WRITE (1)  
 }  
 supported\_command\_list {  
 command\_name : SomeCommand()  
 input\_arg\_name\_list {  
 SomeArgument1  
 SomeArgument2  
 }  
 output\_arg\_name\_list {  
 SomeArgument1  
 SomeArgument2  
 }  
 }  
 supported\_event\_list {  
 event\_name : SomeEvent!  
 arg\_name\_list {  
 SomeArgumentA  
 SomeArgumentB  
 }  
 }   
  
 // And continued, for Objects such as Device.Wifi.SSID.{i}.Stats., Device.Wifi.Radio.{i}, Device.Wifi.AccessPoint.{i}, Device.Wifi.AccessPoint.{i}.AssociatedDevice.{i}, etc.  
 }  
 }

#### GetSupportedDM Request Elements

bool first\_level\_only

This element, if true, indicates that the Agent should return only those objects matched by the Path Name or Search Path in obj\_path and its immediate (i.e., next level) child objects.

bool return\_commands

This element, if true, indicates that, in the supported\_obj\_list, the Agent should include a supported\_command\_list element containing Commands supported by the reported Object(s).

bool return\_events

This element, if true, indicates that, in the supported\_obj\_list, the Agent should include a supported\_event\_list element containing Events supported by the reported Object(s).

bool return\_params

This element, if true, indicates that, in the supported\_obj\_list, the Agent should include a supported\_param\_list element containing Parameters supported by the reported Object(s).

repeated DiscoverObject discover\_obj\_list

This element contains a repeated set of messages of type DiscoverObject.

##### DiscoverObject Elements

string obj\_path

This element contains a Path Name to an Object (not an Object Instance) in the Agent's Supported Data Model.

#### GetSupportedDMResp Elements

repeated RequestedObjectResult req\_obj\_result\_list

This element contains a repeated set of messages of type RequestedObjectResult.

##### RequestedObjectResult Elements

string req\_obj\_path

This element contains one of the Path Names given in obj\_path of the associated GetSupportedDM Request.

fixed32 err\_code

This element contains a [numeric code](#error-codes) indicating the type of error that caused the Get to fail on this path. A value of 0 indicates the path could be read successfully.

**R-GSP.0** - If the Controller making the Request does not have Read permission on an Object or Parameter matched through the requested\_path element, the Object or Parameter MUST be treated as if it is not present in the Agent’s instantiated data model.

string err\_msg

This element contains additional information about the reason behind the error.

string data\_model\_inst\_uri

This element contains a Uniform Resource Identifier (URI) to the Data Model associated with the Object specified in obj\_path.

repeated SupportedObjectResult supported\_obj\_list

The element contains a message of type SupportedObjectResult for each reported Object.

###### SupportedObjectResult Elements

string supported\_obj\_path

This element contains the Path Name of the reported Object.

ObjAccessType access

The element contains an enumeration of type ObjAccessType specifying the access permissions that are specified for this Object in the Agent's Supported Data Model. This usually only applies to Multi-Instance Objects. This may be further restricted to the Controller based on rules defined in the Agent's Access Control List. It is an enumeration of:

OBJ\_READ\_ONLY (0)  
OBJ\_ADD\_DELETE (1)  
OBJ\_ADD\_ONLY (2)  
OBJ\_DELETE\_ONLY (3)

bool is\_multi\_instance

This element, if true, indicates that the reported Object is a Multi-Instance Object.

repeated SupportedParamResult supported\_param\_list

The element contains a message of type SupportedParamResult for each Parameter supported by the reported Object. If there are no Parameters in the Object, this should be an empty list.

repeated SupportedCommandResult supported\_command\_list

The element contains a message of type SupportedCommandResult for each Command supported by the reported Object. If there are no Parameters in the Object, this should be an empty list.

repeated SupportedEventResult supported\_event\_list

The element contains a message of type SupportedEventResult for each Event supported by the reported Object. If there are no Parameters in the Object, this should be an empty list.

###### SupportedParamResult Elements

string param\_name

This element contains the local name of the Parameter.

ParamAccessType access

The element contains an enumeration of type ParamAccessType specifying the access permissions that are specified for this Parameter in the Agent's Supported Data Model. This may be further restricted to the Controller based on rules defined in the Agent's Access Control List. It is an enumeration of:

PARAM\_READ\_ONLY (0)  
PARAM\_READ\_WRITE (1)  
PARAM\_WRITE\_ONLY (2)

###### SupportedCommandResult Elements

string command\_name

This element contains the local name of the Command.

repeated string input\_arg\_name\_list

This element contains a repeated set of local names for the input arguments of the Command.

**R-GSP.1** - If any input arguments are multi-instance, the Agent MUST report them using Instance Number Addressing.

repeated string output\_arg\_name\_list

This element contains a repeated set of local names for the output arguments of the Command.

**R-GSP.2** - If any output arguments are multi-instance, the Agent MUST report them using Instance Number Addressing.

###### SupportedEventResult

string event\_name

This element contains the local name of the Event.

repeated string arg\_name\_list

This element contains a repeated set of local names for the arguments of the Event.

**R-GPS.3** - If any arguments are multi-instance, the Agent MUST report them using Instance Number Addressing.

#### GetSupportedDM Error Codes

Appropriate error codes for the GetSupportedDM message include 7000-7006, 7008, 7016, and 7800-7999.

*Note - when using error 7016 (Object Does Not Exist), it is important to note that in the context of GetSupportedDM this applies to the Agent's Supported Data Model.*

## Notifications and Subscription Mechanism

A Controller can use the Subscription mechanism to subscribe to certain events that occur on the Agent, such as a parameter change, Object removal, wake-up, etc. When such event conditions are met, the Agent sends a [Notify message](#notify) to the Controller.

### The Notify Message

#### Using Subscription Objects

Subscriptions are maintained in instances of the Multi-Instance Subscription Object in the USP data model. The normative requirements for these Objects are described in the data model parameter descriptions for Device.LocalAgent.Subscription.{i}. in [Device:2](https://github.com/BroadbandForum/usp/tree/master/data-model).

**R-NOT.0** - The Agent and Controller MUST follow the normative requirements defined in the Device.LocalAgent.Subscription.{i}. Object specified in [Device:2](https://github.com/BroadbandForum/usp/tree/master/data-model).

*Note: Those familiar with Broadband Forum* [*TR-069*](https://www.broadband-forum.org/technical/download/TR-069.pdf) *will recall that a notification for a value change caused by an Auto-Configuration Server (ACS - the CWMP equivalent of a Controller) are not sent to the ACS. Since there is only a single ACS notifying the ACS of value changes it requested is unnecessary. This is not the case in USP: an Agent should follow the behavior specified by a subscription, regardless of the originator of that subscription.*

###### ReferenceList Parameter

All subscriptions apply to one or more Objects or parameters in the Agent’s Instantiated Data Model. These are specified as Path Names or Search Paths in the ReferenceList parameter. The ReferenceList parameter may have different meaning depending on the nature of the notification subscribed to.

For example, a Controller wants to be notified when a new Wifi station joins the Wifi network. It uses the Add message to create a subscription Object instance with Device.WiFi.AccessPoint.1.AssociatedDevice. specified in the ReferenceList parameter and ObjectCreation as the NotificationType.

In another example, a Controller wants to be notified whenever an outside source changes the SSID of a Wifi network. It uses the Add message to create a subscription Object instance with Device.Wifi.SSID.1.SSID specified in the ReferenceList and ValueChange as the NotificationType.

#### Responses to Notifications and Notification Retry

The Notify request contains a flag, send\_resp, that specifies whether or not the Controller should send a response message after receiving a Notify request. This is used in tandem with the NotifRetry parameter in the subscription Object - if NotifRetry is true, then the Agent sends its Notify requests with send\_resp : true, and the Agent considers the notification delivered when it receives a response from the Controller. If NotifRetry is false, the Agent does not need to use the send\_resp flag and should ignore the delivery state of the notification.

If NotifRetry is true, and the Agent does not receive a response from the Controller, it begins retrying using the retry algorithm below. The subscription Object also uses a NotifExpiration parameter to specify when this retry should end if no success is ever achieved.

**R-NOT.1** - When retrying notifications, the Agent MUST use the following retry algorithm to manage the retransmission of the Notify request.

The retry interval range is controlled by two Parameters, the minimum wait interval and the interval multiplier, each of which corresponds to a data model Parameter, and which are described in the table below. The factory default values of these Parameters MUST be the default values listed in the Default column. They MAY be changed by a Controller with the appropriate permissions at any time.

|  |  |  |  |
| --- | --- | --- | --- |
| Descriptive Name | Symbol | Default | Data Model Parameter Name |
| Minimum wait interval | m | 5 seconds | Device.Controller.{i}.USPRetryMinimumWaitInterval |
| Interval multiplier | k | 2000 | Device.Controller.{i}.USPRetryIntervalMultiplier |

|  |  |  |
| --- | --- | --- |
| Retry Count | Default Wait Interval Range (min-max seconds) | Actual Wait Interval Range (min-max seconds) |
| #1 | 5-10 | m - m.(k/1000) |
| #2 | 10-20 | m.(k/1000) - m.(k/1000)2 |
| #3 | 20-40 | m.(k/1000)2 - m.(k/1000)3 |
| #4 | 40-80 | m.(k/1000)3 - m.(k/1000)4 |
| #5 | 80-160 | m.(k/1000)4 - m.(k/1000)5 |
| #6 | 160-320 | m.(k/1000)5 - m.(k/1000)6 |
| #7 | 320-640 | m.(k/1000)6 - m.(k/1000)7 |
| #8 | 640-1280 | m.(k/1000)7 - m.(k/1000)8 |
| #9 | 1280-2560 | m.(k/1000)8 - m.(k/1000)9 |
| #10 and subsequent | 2560-5120 | m.(k/1000)9 - m.(k/1000)10 |

**R-NOT.2** - Beginning with the tenth retry attempt, the Agent MUST choose from the fixed maximum range. The Agent will continue to retry a failed notification until it is successfully delivered or until the NotifExpiration time is reached.

**R-NOT.3** - Once a notification is successfully delivered, the Agent MUST reset the retry count to zero for the next notification message.

**R-NOT.4** - If a reboot of the Agent occurs, the Agent MUST reset the retry count to zero for the next notification message.

#### Notification Types

There are several types events that can cause a Notify request. These include those that deal with changes to the Agent’s Instantiated Data Model (ValueChange, ObjectCreation, ObjectDeletion), the completion of an asynchronous Object-defined operation (OperationComplete), a policy-defined OnBoardRequest, and a generic Event for use with Object-defined events.

##### ValueChange

The ValueChange notification is subscribed to by a Controller when it wants to know that the value of a single or set of parameters has changed from the state it was in at the time of the subscription or to a state as described in an expression, and then each time it transitions from then on for the life of the subscription. It is triggered when the defined change occurs, even if it is caused by the originating Controller.

##### ObjectCreation and ObjectDeletion

These notifications are used for when an instance of the subscribed to Multi-Instance Objects is added or removed from the Agent’s Instantiated Data Model. Like ValueChange, this notification is triggered even if the subscribing Controller is the originator of the creation or deletion.

The ObjectCreation notification also includes the Object’s unique keys and their values as data in the notification.

##### OperationComplete

The OperationComplete notification is used to indicate that an asynchronous Object-defined operation finished (either successfully or unsuccessfully). These operations may also trigger other Events defined in the data model (see below).

##### OnBoardRequest

An OnBoardRequest notification is used by the Agent when it is triggered by an external source to initiate the request in order to communicate with a Controller that can provide on-boarding procedures and communicate with that Controller (likely for the first time).

**R-NOT.5** - An Agent MUST send an OnBoardRequest notify request in the following circumstances:

1. When the SendOnBoardRequest() command is executed. This sends the notification request to the Controller that is the subject of that operation. The SendOnBoardRequest() operation is defined in the [Device:2 Data Model for TR-069 Devices and USP Agents](https://github.com/BroadbandForum/usp/tree/master/data-model).
2. When instructed to do so by internal application policy (for example, when using DHCP discovery defined above).

*Note: as defined in the Subscription table, OnBoardRequest is not included as one of the enumerated types of a Subscription, i.e., it is not intended to be the subject of a Subscription.*

**R-NOT.6** a response is required, the OnBoardRequest MUST follow the Retry logic defined above.

##### Event

The Event notification is used to indicate that an Object-defined event was triggered on the Agent. These events are defined in the data model and include what parameters, if any, are returned as part of the notification.

#### Notify Examples

In this example, a Controller has subscribed to be notified of changes in value to the Device.DeviceInfo.FriendlyName parameter. When it is changed, the Agent sends a Notify Request to inform the Controller of the change.

Noify Request:  
header {  
 msg\_id: "33936"  
 msg\_type: NOTIFY  
 proto\_version: "1.0"  
 to\_id: “oui:112233:controller”  
 from\_id: “oui:112233:agent”  
}  
body {  
 request {  
 notify {  
 subscription\_id: "vc-1"  
 send\_resp: true  
 value\_change {  
 param\_path: "Device.DeviceInfo.FriendlyName"  
 param\_value: "MyDevicesFriendlyName"  
 }  
 }  
 }  
}  
  
Notify Response:  
header {  
 msg\_id: "33936"  
 msg\_type: NOTIFY\_RESP  
 proto\_version: "1.0"  
 to\_id: “oui:112233:agent”  
 from\_id: “oui:112233:controller”  
}  
body {  
 response {  
 notify\_resp {  
 subscription\_id: "vc-1"  
 }  
 }  
}

In another example, the event "Boot!", defined in the Device.LocalAgent. object, is triggered. The Agent sends a Notify Request to the Controller(s) subscribed to that event.

Notify Request  
header {  
 msg\_id: "26732"  
 msg\_type: NOTIFY  
 proto\_version: "1.0"  
 to\_id: “oui:112233:controller”  
 from\_id: “oui:112233:agent”  
}  
body {  
 request {  
 notify {  
 subscription\_id: "boot-1"  
 send\_resp: true  
 event {  
 obj\_path: "Device.LocalAgent."  
 event\_name: "Boot!"  
 param\_map {  
 key: "Cause"  
 value: "LocalReboot"  
  
 key: "CommandKey"  
  
 key: "Parameter.1.Path"  
 value: "Device.LocalAgent.Controller.1.Enable"  
  
 key: "Parameter.1.Value"  
 value: "True"  
 }  
 }  
 }  
 }  
}  
  
Notify Response:  
header {  
 msg\_id: "26732"  
 msg\_type: NOTIFY\_RESP  
 proto\_version: "1.0"  
 to\_id: “oui:112233:agent”  
 from\_id: “oui:112233:controller”  
}  
body {  
 response {  
 notify\_resp {  
 subscription\_id: "boot-1"  
 }  
 }  
}

#### Notify Request Elements

string subscription\_id

This element contains the locally unique opaque identifier that was set by the Controller when it created the Subscription on the Agent.

**R-NOT.7** - The subscription\_id element MUST contain the Subscription ID of the Subscription Object that triggered this notification.

bool send\_resp

This element lets the Agent indicate to the Controller whether or not it expects a response in association with the Notify request.

**R-NOT.8** - When send\_response is set to false, the Controller SHOULD NOT send a response or error to the Agent. If a response is still sent, the responding Controller MUST expect that any such response will be ignored.

oneof notification

Contains one of the following Notification messages:

Event event  
ValueChange value\_change  
ObjectCreation obj\_creation  
ObjectDeletion obj\_deletion  
OperationComplete oper\_complete  
OnBoardRequest on\_board\_req

##### Event Elements

string obj\_path

This element contains the Object or Object Instance Path of the Object that caused this event (for example, Device.LocalAgent.).

**R-NOT.9** - Path Names containing Object Instances in the obj\_path element of the Event notification MUST be addressed using Instance Number Addressing.

string event\_name

This element contains the name of the Object defined event that caused this notification (for example, Boot!).

map<string, string> parameter\_map

This element contains a set of key/value pairs of parameters associated with this event.

**R-NOT.10** - Any values in parameter\_map whose keys contain Object Paths to Multi-Instance Objects MUST be addressed by Instance Number.

##### ValueChange Elements

string param\_path

This element contains the Path Name of the changed parameter.

**R-NOT.11** - Path Names containing Object Instances in the param\_path element of the ValueChange notification MUST be addressed using Instance Number Addressing.

string param\_value

This element contains the value of the parameter specified in param\_path.

##### ObjectCreation Elements

string obj\_path

This element contains the Path Name of the created Object instance.

**R-NOT.12** - Path Names containing Object Instances in the obj\_path element of the ObjectCreation notification MUST be addressed using Instance Number Addressing.

map<string, string> unique\_key\_map

This element contains a map of key/value pairs for all supported parameters that are part of any of this Object's unique keys.

##### ObjectDeletion Elements

string obj\_path

This element contains the Path Name of the deleted Object instance.

**R-NOT.13** - Path Names containing Object Instances in the obj\_path element of the ObjectDeletion notification MUST be addressed using Instance Number Addressing.

##### OperationComplete Elements

string command\_name

This element contains the local name l of the Object defined command that caused this notification (i.e., Download()).

string obj\_path

This element contains the Object or Object Instance Path to the Object that contains this operation.

**R-NOT.14** - Path Names containing Object Instances in the obj\_path element of the OperationComplete notification MUST be addressed using Instance Number Addressing.

string command\_key

This element contains the command key set during an Object defined Operation that caused this notification.

oneof operation\_resp

Contains one of the following messages:

OutputArgs req\_output\_args  
CommandFailure cmd\_failure

###### OutputArgs Elements

map<string, string> output\_arg\_map

This element contains a map of key/value pairs indicating the output arguments (relative to the command specified in the command\_name element) returned by the method invoked in the Operate message.

**R-NOT.15** - Any key in the output\_arg\_map that contains multi-instance arguments MUST use Instance Number Addressing.

###### CommandFailure Elements

fixed32 err\_code

This element contains the [error code](#error-codes) of the error that caused the operation to fail. Appropriate error codes for CommandFailure include 7002-7008, 7016, 7022, 7023, and 7800-7999.

string err\_msg

This element contains additional (human readable) information about the reason behind the error.

##### OnBoardRequest Elements

string obj\_path

This element contains the Path Name of the Object associated with this notification.

**R-NOT.16** - Path Names containing Object Instances in the obj\_path element of the OnBoardRequest notification MUST be addressed using Instance Number Addressing.

#### Notify Response Elements

string subscription\_id

This element contains the locally unique opaque identifier that was set by the Controller when it created the Subscription on the Agent.

**R-NOT.17** - The subscription\_id element MUST contain the Subscription ID of the Subscription Object that triggered this notification. If the subscription\_id element does not contain the Subcription ID of the Subscription Object that triggered this notification, this Response MUST be ignored and not considered valid for the purpose of calculating notification retries.

#### Notify Error Codes

Appropriate error codes for the Notify message include 7000-7006, and 7800-7999.

## Defined Operations Mechanism

Additional methods (operations) are and can be defined in the USP data model. Operations are generally defined on an Object, using the "command" attribute, as defined in [TR-106](https://www.broadband-forum.org/technical/download/TR-106_Amendment-8.pdf). The mechanism is controlled using the [Operate message](#operate) in conjunction with the Multi-Instance Request Object.

### Synchronous Operations

A synchronous operation is intended to complete immediately following its processing. When complete, the output arguments are sent in the Operate response. If the send\_resp flag is false, the Controller doesn’t need the returned information (if any), and the Agent does not send an Operate Response.

Figure OPR.1 - Operate Message Flow for Synchronous Operations

### Asynchronous Operations

An asynchronous operation expects to take some processing on the system the Agent represents and will return results at a later time. When complete, the output arguments are sent in a Notify (OperationComplete) request to any Controllers that have an active subscription to the operation and Object(s) to which it applies.

When a Controller using the Operate request specifies an operation that is defined as asynchronous, the Agent creates an instance of the Request Object in its data model, and includes a reference to the created Object in the Operate response. If the send\_resp flag is false, the Controller doesn’t need the Request details, and intends to ignore it.

The lifetime of a Request Object expires when the operation is complete (either by success or failure). An expired Request Object is removed from the Instantiated Data Model.

**R-OPR.0** - When an Agent receives an Operate Request that addresses an asynchronous operation, it MUST create a Request Object in the Request table of its Instantiated Data Model (see [Device:2](https://github.com/BroadbandForum/usp/tree/master/data-model)). When the Operation is complete (either success or failure), it MUST remove this Object from the Request table.

If any Controller wants a notification that an operation has completed, it creates a Subscription Object with the NotificationType set to OperationComplete and with the ReferenceList parameter including a path to the specified command. The Agent processes this Subscription when the operation completes and sends a Notify message, including the command\_key value that the Controller assigned when making the Operate request.

Figure OPR.2 - Operate Message Flow for Asynchronous Operations

#### Persistance of Asynchronous Operations

Synchronous Operations do not persist across a reboot or restart of the Agent or its underlying system. It is expected that Asynchronous Operations do not persist, and a command that is in process when the Agent is rebooted can be expected to be removed from the Request table, and is considered to have failed. If a command is allowed or expected to be retained across a reboot, it will be noted in the command description.

### Operate Requests on Multiple Objects

Since the Operate request can take a path expression as a value for the command element, it is possible to invoke the same operation on multiple valid Objects as part of a single Operate request. Responses to requests to Operate on more than one Object are handled using the OperationResult element type, which is returned as a repeated set in the Operate Response. The success or failure of the operation on each Object is handled separately and returned in a different OperationResult entry. For this reason, operation failures are never conveyed in an Error message - in reply to an Operate request, Error is only used when the message itself fails for one or more reasons, rather than the operation invoked.

**R-OPR.1** - When processing Operate Requests on multiple Objects, an Agent MUST NOT send an Error message due to a failed operation. It MUST instead include the failure in the cmd\_failure element of the Operate response.

**R-OPR.2** - For asynchronous operations the Agent MUST create a separate Request Object for each Object and associated operation matched in the command element.

### Event Notifications for Operations

When an operation triggers an Event notification, the Agent sends the Event notification for all subscribed recipients as described [above](#notifications_and_subscrptions)

### Concurrent Operations

If an asynchronous operation is triggered multiple times by one or more Controllers, the Agent has the following options:

1. Deny the new operation (with, for example, 7005 Resources Exceeded )
2. The operations are performed in parallel and independently.
3. The operations are queued and completed in order.

**R-OPR.3** - When handling concurrently invoked operations, an Agent MUST NOT cancel an operation already in progress unless explicitly told to do so by a Controller with permission to do so.

### Operate Examples

In this example, the Controller requests that the Agent initiate the SendOnBoardRequest() operation defined in the Device.LocalAgent.Controller. object.

Operate Request:  
header {  
 msg\_id: "42314"  
 msg\_type: OPERATE  
 proto\_version: "1.0"  
 to\_id: “oui:112233:agent”  
 from\_id: “oui:112233:controller”  
}  
body {  
 request {  
 operate {  
 command: "Device.LocalAgent.Controller.[EndpointID==\"controller\"].SendOnBoardRequest()"  
 command\_key: "onboard\_command\_key"  
 send\_resp: true  
 }  
 }  
}  
  
  
Response:  
header {  
 msg\_id: "42314"  
 msg\_type: OPERATE\_RESP  
 proto\_version: "1.0"  
 to\_id: “oui:112233:controller”  
 from\_id: “oui:112233:agent”  
}  
body {  
 response {  
 operate\_resp {  
 operation\_result\_list {  
 executed\_command: "Device.LocalAgent.Controller.1.SendOnBoardRequest()"  
 }  
 }  
 }  
}

### The Operate Message

#### Operate Request Elements

string command

This element contains a Command Path or Search Path to an Object defined Operation in one or more Objects.

string command\_key

This element contains a string used as a reference by the Controller to match the operation with notifications.

bool send\_resp

This element lets the Controller indicate to Agent whether or not it expects a response in association with the operation request.

**R-OPR.4** - When send\_resp is set to false, the target Endpoint SHOULD NOT send a response or error to the source Endpoint. If a response is still sent, the responding Endpoint MUST expect that any such response will be ignored.

map<string, string> input\_arg\_map

This element contains a map of key/value pairs indicating the input arguments (relative to the command path in the command element) to be passed to the method indicated in the command element.

**R-OPR.5** - Any key in the input\_arg\_map that contains multi-instance arguments MUST use Instance Number Addressing. This element contains a map of name/value pairs indicating the input arguments (relative to the Object that is the subject of this command) to be passed to the method invoked indicated in the command element.

#### Operate Response Elements

repeated OperationResult operation\_result\_list

This element contains a repeated set of OperationResult messages.

##### OperationResult Elements

string executed\_command

This element contains a Command Path to the Object defined Operation that is the subject of this OperateResp message.

oneof operate\_resp

This element contains a message of one of the following types.

string req\_object\_path

This element contains an Object Instance Path to the Request Object created as a result of this asynchronous operation.

**R-OPR.6** - Path Names in the req\_object\_path MUST use Instance Number Addressing.

OutputArgs req\_output\_args  
CommandFailure cmd\_failure

This element contains one message of type CommandFailure. It is used when at synchronous operation is not successful.

###### OutputArgs Elements

map<string, string> output\_arg\_map

This element contains a map of key/value pairs indicating the output arguments (relative to the command specified in the command element) returned by the method invoked in the Operate message.

**R-OPR.7** - Any key in the output\_arg\_map that contains multi-instance arguments MUST use Instance Number Addressing.

###### CommandFailure elements

fixed32 err\_code

This element contains the [error code](#error-codes) of the error that caused the operation to fail.

string err\_msg

This element contains additional (human readable) information about the reason behind the error.

#### Operate Message Error Codes

Appropriate error codes for the Operate message include 7000-7008, 7012 7015, 7016, 7022, and 7800-7999.

## Error Codes

USP uses error codes with a range 7000-7999 for both Controller and Agent errors. The errors appropriate for each message (and how they must be implemented) are defined in the message descriptions below.

|  |  |  |
| --- | --- | --- |
| Code | Name | Description |
| 7000 | Message failed | This error indicates a general failure that is described in an err\_msg element. |
| 7001 | Message not supported | This error indicates that the attempted message was not understood by the target endpoint. |
| 7002 | Request denied (no reason specified) | This error indicates that the target endpoint cannot or will not process the message. |
| 7003 | Internal error | This error indicates that the message failed due to internal hardware or software reasons. |
| 7004 | Invalid arguments | This error indicates that the message failed due to invalid values in the Request elements and/or the failure to update one or more parameters during an Add or Set message. |
| 7005 | Resources exceeded | This error indicates that the message failed due to memory or processing limitations on the target endpoint. |
| 7006 | Permission denied | This error indicates that the source endpoint does not have the authorization for this action. |
| 7007 | Invalid configuration | This error indicates that the message failed because processing the message would put the target endpoint in an invalid or unrecoverable state. |
| 7008 | Invalid path syntax | This error indicates that the Path Name used was not understood by the target endpoint. |
| 7009 | Parameter action failed | This error indicates that the parameter failed to update for a general reason described in an err\_msg element. |
| 7010 | Unsupported parameter | This error indicates that the requested Path Name associated with this ParamError did not match any instantiated parameters. |
| 7011 | Invalid type | This error indicates that the requested value was not of the correct data type for the parameter. |
| 7012 | Invalid value | This error indicates that the requested value was not within the acceptable values for the parameter. |
| 7013 | Attempt to update non-writeable parameter. | This error indicates that the source endpoint attempted to update a parameter that is not defined as a writeable parameter. |
| 7014 | Value conflict | This error indicates that the requested value would result in an invalid configuration based on other parameter values. |
| 7015 | Operation error | This error indicates a general failure in the creation, update, or deletion of an Object that is described in an err\_msg element. |
| 7016 | Object does not exist | This error indicates that the requested Path Name associated with this OperationStatus did not match any instantiated Objects. |
| 7017 | Object could not be created | This error indicates that the operation failed to create an instance of the specified Object. |
| 7018 | Object is not a table | This error indicates that the requested Path Name associated with this OperationStatus is not a Multi-Instance Object. |
| 7019 | Attempt to create non-creatable Object | This error indicates that the source endpoint attempted to create an Object that is not defined as able to be created. |
| 7020 | Object could not be updated | This error indicates that the requested Object in a Set request failed to update. |
| 7021 | Required parameter failed | This error indicates that the request failed on this Object because one or more required parameters failed to update. Details on the failed parameters are included in an associated ParamError message. |
| 7022 | Command failure | This error indicates that an command initiated in an Operate Request failed to complete for one or more reasons explained in the err\_msg element. |
| 7023 | Command canceled | This error indicates that an asynchronous command initiated in an Operate Request failed to complete because it was cancelled using the Cancel() operation. |
| 7024 | Delete failure | This error indicates that this Object Instance failed to be deleted. |
| 7025 | Object exists with duplicate key | This error indicates that an Object tried to be created with a unique keys that already exist, or the unique keys were configured to those that already exist. |
| 7800-7999 | Vendor defined error codes | These errors are [vendor defined](#vendor_defined_error_codes). |

### Vendor Defined Error Codes

Implementations of USP MAY specify their own error codes for use with Errors and Responses. These codes use the 7800-7999 series. There are no requirements on the content of these errors.