# CoAP Binding

The Constrained Application Protocol (CoAP) MTP transfers USP Messages between USP Endpoints using the CoAP protocol as defined in [RFC 7252](https://tools.ietf.org/html/rfc7252). Messages that are transferred between CoAP clients and servers utilize a request/response messaging interaction based on RESTful architectural principles. The following figure depicts the transfer of the USP messages between USP Endpoints.

Figure COAP.1 - USP Request/Response over the CoAP MTP

USP Requests are encapsulated within a CoAP request message. When a USP Endpoint receives the CoAP request message the USP Endpoint immediately sends a CoAP response message (with no USP Message) to indicate receipt of the message. A USP Response is encapsulated in a new CoAP request message. When a USP Endpoint receives a USP Response, it sends a CoAP response message that indicates receipt of the message. Therefore, all Endpoints supporting CoAP will implement both CoAP client and server.

As noted in the definition of a USP Request, this USP Message either requests the Agent perform some action (create, update, delete, operate, etc.), requests information about an Agent or one or more Service Elements, or acts as a means to deliver Notifications from the Agent to the Controller. Notifications will only cause a USP Response to be generated if specified in the Notification Request. However, the CoAP response will always be sent.

## Mapping USP Endpoints to CoAP URIs

Section 6 of [RFC 7252](https://tools.ietf.org/html/rfc7252) discusses the URI schemes for identifying CoAP resources and provides a means of locating the resource. These resources are organized hierarchically and governed by a CoAP server listening for CoAP requests on a given port. USP Endpoints are one type of CoAP resource that is identified and discovered.

**R-COAP.0** - As the USP Endpoint is a resource governed by a CoAP server, the CoAP server MUST also be identified as defined in section 6 of [RFC 7252](https://tools.ietf.org/html/rfc7252).

**R-COAP.1** - A USP Endpoint MUST be represented as a CoAP resource with the following resource attributes: \* Identifier within the CoAP server (uri-path) \* Resource type (rt): “usp.endpoint” \* Interface (if): "usp.c" for USP Controller or “usp.a” for USP Agent

The identifier within the CoAP server is used to deliver messages to the USP Endpoint. When this identifier is used to deliver messages to the USP Endpoint, this identifier is a uri-path that represents the the USP Endpoint Identifier. For example: “/e/{endpointId}”

## Mapping USP Messages to CoAP Messages

**R-COAP.2** - In order for USP Messages to be transferred between a USP Controller and Agent using CoAP, the USP Message MUST be encapsulated within the CoAP message as defined in [RFC 7252](https://tools.ietf.org/html/rfc7252).

**R-COAP.3** - USP Messages that exceed the CoAP message size MUST be block encapsulated in accordance with [RFC 7959](https://www.rfc-editor.org/rfc/rfc7959.txt).

USP Messages are transferred using the CoAP resource that represents the receiving USP Endpoint using the CoAP POST method as defined in [RFC 7252](https://tools.ietf.org/html/rfc7252).

**R-COAP.4** - The CoAP Content-Format for USP Messages MUST be application/octet-stream (ID=42) for [protobuf encoding](/specification/encoding).

### Handling CoAP Request Success

**R-COAP.5** - Upon successful reception of the CoAP message using POST, the CoAP server MUST respond with a response code of 2.04 (Changed).

### Handling CoAP Request Failures

At times CoAP requests fail to complete due to problems in the underlying transport (e.g., timeout) or a failure response code received from the CoAP server due to problems in the CoAP request sent by the CoAP client (4.xx) or problems with the CoAP server implementation (5.xx).

**R-COAP.6** - CoAP clients and servers MUST implement the required CoAP response codes defined in section 5.9 of [RFC 7252](https://tools.ietf.org/html/rfc7252).

**R-COAP.7** - When a CoAP client receives a failure indication (e.g., timeout) from the underlying transport layer, the CoAP client MUST indicate a timeout to the USP Endpoint.

**R-COAP.8** - When a CoAP client receives a response code of 4.xx or 5.xx, the CoAP client MUST indicate a CoAP failure to the USP Endpoint.

When a CoAP client sends a CoAP request, the CoAP client can provide incorrect or missing information in the CoAP request. For example, a CoAP client can send a CoAP request with an:

* Invalid CoAP method: The CoAP server responds with a 4.05
* Invalid Content-Format options: The CoAP server responds with a 4.15
* Invalid or not understandable payload: The CoAP server responds with a 4.00

**R-COAP.9** - When a CoAP server receives a CoAP request with an invalid CoAP method, the CoAP server MUST respond with a 4.05 response code.

**R-COAP.10** - When a CoAP server receives a CoAP request with an invalid CoAP Content-Format option, the CoAP server MUST respond with a 4.15 response code.

**R-COAP.11** - When a CoAP server receives a CoAP request and the receiving USP Endpoint cannot interpret or decode the USP Message for processing, the CoAP server MUST respond with a 4.00 response code.

## Mapping USP Notification Message to CoAP Message

The USP Notify Request is the only message of the Request type that is sent from an Agent to a Controller. In these cases, the Agent functions as a CoAP client rather than a CoAP server in the POST/Response sequence.

**R-COAP.12** - When a CoAP server receives a USP Notify Request, the CoAP server MUST adhere to the requirements defined in [Mapping USP Messages to CoAP Messages](#mapping_usp_messages_to_coap_messages).

## MTP Message Encryption

CoAP MTP message encryption is provided using DTLS as described in Section 9 of [RFC 7252](https://tools.ietf.org/html/rfc7252).

In section 9 of [RFC 7252](https://tools.ietf.org/html/rfc7252), CoAP messages are secured using one of three modes:

* NoSec: DTLS is disabled
* PreSharedKey: DTLS is enabled and the MTP endpoint uses pre-shared keys that are used to validate the identity of CoAP endpoints involved in the message exchange
* RawPublicKey: DTLS is enabled and the MTP endpoint has an asymmetric key pair without a certificate. The MTP endpoint has an identity calculated from the public key and a list of other MTP endpoints to which it can communicate
* Certificate: DTLS is enabled and the MTP endpoint has an asymmetric key pair with an X.509 certificate.

**R-COAP.13** - CoAP clients and servers MUST implement the NoSec and Certificate modes of CoAP security as defined in RFC 7252.

While section 9 of [RFC 7252](https://tools.ietf.org/html/rfc7252) provides guidance on securing CoAP, further guidance related to DTLS implementations for the Internet of Things is provided by [RFC 7925](https://tools.ietf.org/html/rfc7925).

**R-COAP.14** - CoAP clients and servers MUST implement the mandatory statements of [RFC 7925](https://tools.ietf.org/html/rfc7925) with the exception that:

* Section 4.4.1 USP Controller certificates can contain domain names with wildcard characters per [RFC 6125](https://tools.ietf.org/html/rfc6125) guidance.
* Section 4.4.2 Client certificate identifiers do not use EUI-64 identifier but instead use the identifier defined for Client certificates in this Working Text.
* Section 4.4.5 Client Certificate URLs are not required to be implemented.

[<-- Message Transfer Protocols](/specification/mtp/) [--> Message Encoding](/specification/encoding/)