WoT-AP Proposal for Interaction Model mapping to an Abstract Transfer Layer

Scope

The scope of this proposal is to define an abstract transfer layer interface and default mapping of the event, action, and property endpoints identified by the "hrefs" elements in a Thing Description resource, as defined in the W3C Web of Things IG Current Practice document.

This document is intended to provide guidance for the creation of interoperable network interfaces between WoT Servient instances and other WoT clients and servers, using standard transfer layer protocols HTTP, CoAP, and MQTT.

Abstract transfer layer

The examples use an abstract transfer layer, with the following mapping to REST transfer layer operations and pubsub communication patterns

Abstract Transfer	Pubsub	CoAP	HTTP
Create	(Publish)	POST	POST
Retrieve	Subscribe (with retain)	GET	GET
Update	Publish	PUT	PUT
Delete	N/A	DELETE	DELETE
Observe	Subscribe	GET with OBS option	GET text/stream, TE=chunked (SSE)
Notify	Notify Client (onMessage)	Response with OBS option	SSE chunk Response

TD example

Events, actions, and properties defined in a Thing Description contain one or more "hrefs" resource references which identify transfer layer endpoints using URIs or other transfer layer specific identifiers. For the purpose of this proposal, URIs are assumed.

```
{
       ...
  "properties": [
    {
      "@type": "sensor:Temperature",
      "name": "temperature",
      "sensor:unit": "sensor:Celsius",
      "valueType": "xsd:float",
      "writable": false,
      "hrefs": "temp"
    }
  ]
       ...
}
```

The hrefs key value pair contains a URI pointing to the resource that exposes the described property, in this example the URI is "temp", which according to RFC3986 should be resolved as a relative URI from the current URI context (the URI of the thing)

Events and Actions also contain "hrefs" definitions for exposed transfer layer endpoints which are used in a similar fashion.

WoT Interaction Model mapping to the abstract transfer layer

These sections define the mapping of Event, Action, and Property elements of the W3C WoT Interaction Model to operations exposed by the abstract transfer layer. Additional resource classes ActionInstance and Subscription are defined to expose explicit representations of long running state transition relationships.

Property

The Property resource exposes a value of a property, and has the following methods:

Retrieve (getProperty)

Retrive implements the getProperty operation, which returns a current representation of the property according to the type definition for the resource.

Update (setProperty)

Update implements the setProperty operation, which accepts a representation of the desired state of the resource, according to the type definition, and replaces the resource representation with the supplied representation

Observe

Observe invokes the transfer layer asynchronous notification mechanism, and registers a retrieve request with the resource server such that a new response will be sent each time the resource is updated. This can use the familiar "RESThook" pattern for REST resources ans is analogous to MQTT SUBSCRIBE.

Create (createSubscription)

See "Subscription" section.

Event

The Event resource emits instances of events. The Event resource allows the following methods:

Retrieve (getEvents)

The Event resource may keep a list of recent events, which may be retrieved using the retrieve method on the Event resource. This resource should at least supply a representation of the most recently emitted event.

Observe

Observe of an event resource registers a retrieve request with the resource server, as it does with property resources. Each time an event is emitted by the event source, a representation of the event will be sent as a notification response to the Observe operation.

Create (createSubscription)

See "Subscription" section.

Action

The Action resource accepts commands to invoke Actions with a set of defined parameters. The Action resource allows the following methods:

Create (invokeAction)

Performing a Create method on an action resource invokes the action and creates an ActionInstance resource which represents the pending, executing, or final state of the action.

The return information from this method includes an identifier, location or handle, for the created ActionInstance.

Retrieve (getActionInstances)

Performing a Retrieve method operation on an Action resource will return a list (array) of zero or more pending, executing, or finished ActionInstance representations.

ActionInstance

ActionInstances are created when actions are invoked, and allow tracking and managing the progress of running actions. When an Action is invoked, a handle to an ActionInstance is returned. The ActionInstance allows the following methods:

Observe

Register a retrieve request with the resource server which transmits each update on the ActionInstance as a progress update, consisting of at least one update to indicate the final disposition of the physical action, success or failure.

Update

The paramaters of a pending or executing ActionInstance may be modified, potentially affecting the physical actuation pending or in progress. Updates of an ActionInstance will result in notification responses transmitted to any registered Observers.

Delete

Am ActionInstance may be deleted, potentially resulting in the cancellation of a pending action or aborting of an action in progress.

Create (createSubscription)

See "Subscription" section.

Subscription

Subscriptions are created on Properties, Events, and ActionInstances to link the state changes of the associated property, the updates to the action instance, or emitted events to a transfer layer operation that can update the state of another resource or collection, or publish state updates to a broker. Subscription resources allow the following methods:

Create (createSubscription)

If no target URI is specified, the system will create the target of the subscription at the location returned in the handle for the created subscription. In this case, the resource identified by the handle may be observed or have another subscription created on it.

If a target URI is specified, the subscription will use information supplied in the constructor of the subscription to send update or create messages to the target resource.

Subscriptions may be used to publish updates to a message broker, for example using the MQTT protocol. For example, if the target resource of the subscription specifies the "mqtt" scheme, the subscription will publish updates to the MQTT broker and topic specified in the target URL.

Observe

The target resource of a Subscription should be observable.

A Subscription created without providing a target URL will result in the creation of an observable resource at the location returned as the handle for the created object.

Update

A subscription resource, once created, may be updated to change it's parameter values. Some values of a subscription may be immutable once the subscription is created, for example the target URL.

Delete

A subscription resource may be deleted, resulting in the cessation of transmission of any messages. A final message indicating termination of the subscription may be sent, for example CoAP response with obs=1, HTTP 400 response, MQTT LWT message.

Payload formats

TBD - formats and keywords for ActionInstances and Subscriptions