Example Sequences of interaction patterns (for discussion in Prague)

2018/3/27
Panasonic
Toru Kawaguchi
(with Kazuo Kajimoto)
Motivation

• For interaction patterns (Property, Action and Event), we have rough consensus on which to use for what, how to use them, and so on.

• But there is no clear description of end-to-end sequence for each interaction pattern, so it is not easy to identify
  • what is specified and what is not within spec
  • consistency between TD, Scripting and Binding

• To make clear such aspect, this is a strawman to walk through the end-to-end sequence with diagrams.
1. Property
1.1 readProperty (binding = HTTP)
1. Property
1.2 writeProperty (binding = HTTP)

Client Application

Scripting API

calls thing.writeProperty()

returns Promise

calls back Promise resolving to void

Server (Thing)

WoT Server

Device

sends HTTP PUT Request to href of form (payload: schema data)

sets Property value to corresponding device

sends back HTTP PUT Response
1. Property

1.3a observeProperty

(binding = HTTP Long Polling)

Client

Application

calls thing
.onPropertyChanged()
.subscribe(Observer)

returns Subscription

calls back
Observer.next() with
schema data

calls Subscription
.unsubscribe()

returns void

Scripting API

Server (Thing)

WoT Server

Device

waits until corresponding
device detects change

sends HTTP GET Request to href of form
with rel=observeProperty

sends back HTTP GET Response
(payload: schema data)

sends HTTP GET Request to href of form
with rel=observeProperty again

cancels ongoing HTTP GET Request to href of form

2018/3/27

Panasonic Corporation
1. Property
1.3b observeProperty
   (binding = Simple WebSocket)

Server (Thing)

Client Application  Scripting API

- calls thing
  .onPropertyChange()
  .subscribe(Observer)
- returns Subscription
- calls Subscription
  .unsubscribe()
- returns void

WoT Server

Device

connects WebSocket to href of form with rel=observeProperty

waits until corresponding device detects change

sends schema data to WoT Client Lib through WebSocket connection

detects

disconnects WebSocket connection
2. Action (binding = HTTP)

Note:
- Considering that consumed Thing only have invokeAction() method currently, assumption here is that POST method is synchronous, so it returns after some time when the device completed the action.
- Question: do we need to consider adding “cancelAction()” in this case?
3a. Event (binding = HTTP Long Polling)

Client

- Application
- Scripting API

Server (Thing)

- WoT Server
- Device

calls thing. onEvent().subscribe(Observer)

returns Subscription

calls back Observer.next() with schema data

calls Subscription .unsubscribe()

returns void

sends HTTP GET Request to href of form with rel=subscribeEvent

waits until corresponding device detects change

detect

sends back HTTP GET Response (payload: schema data)

sends HTTP GET Request to href of form with rel=observeProperty again

cancels ongoing HTTP GET Request to href of form
3b. Event
(binding = Simple WebSocket)

Client
Application Scripting API

Server (Thing)
WoT Server Device

- calls thing. onEvent().subscribe(Observer)
- returns Subscription
- calls back Observer.next() with schema data
- calls Subscription.unsubscribe()
- returns void
- disconnects WebSocket connection

- connects WebSocket to href of form with rel=subscribeEvent
- sends schema data to WoT Client Lib through WebSocket connection
- waits until corresponding device detects change
- detect
- returns void
- calls Subscription.unsubscribe()
Further things to consider

• Textual definition and explanation of interaction patterns at one place (e.g. architecture document), prior to this illustration
  • What are the Property, Action, Event is
  • How the Property, Action, Event can be used
    (https://github.com/w3c/wot-architecture/issues/49#issuecomment-375266465)

• How to model more complicated patterns?
  • Asynchronous Action with monitor and cancel features.

• Do we need examples for other protocol bindings?
  • CoAP, MQTT, ...
  • Event with late binding (prior Panasonic proposal)