Common Resource Model and Transfer Layer for the Web of Things

Michael Koster

May 11, 2016

Problems being solved

- Implementation of WoT Web Interface, Abstract Transfer Layer and Common Resource Model
- Resource design patterns for Events, Actions, and Properties
- Asynchronous communication for RESTful systems

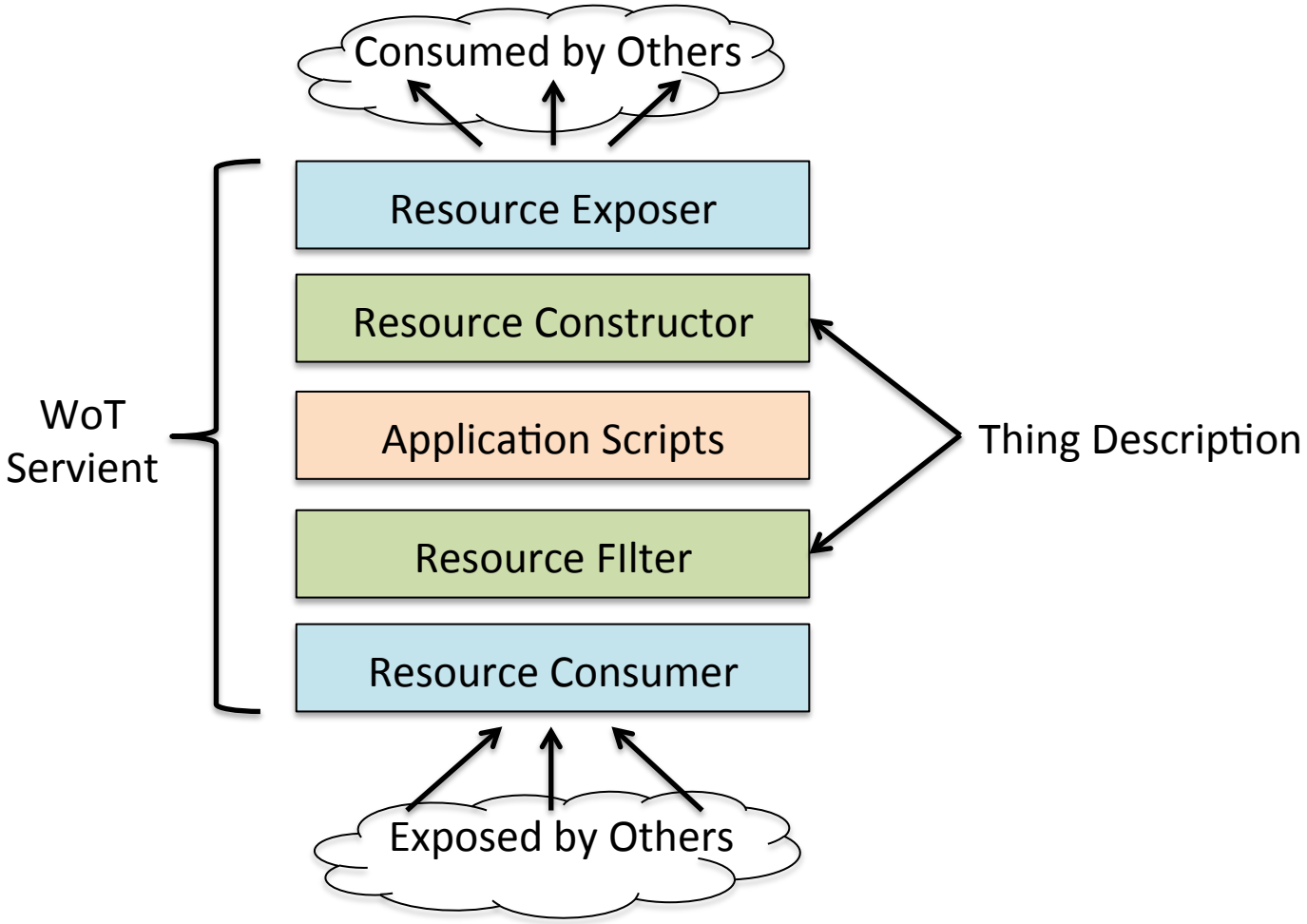
Scope

- Provide a consistent resource interaction model exposed by the "hrefs" of an event, action, or property – Data at rest
- Abstract transfer layer mapping to common protocols CoAP, HTTP, MQTT, Websockets using protocol bindings - Data in motion
- Implement the WoT servient pattern using Thing Description for resource construction, discovery, and application interaction

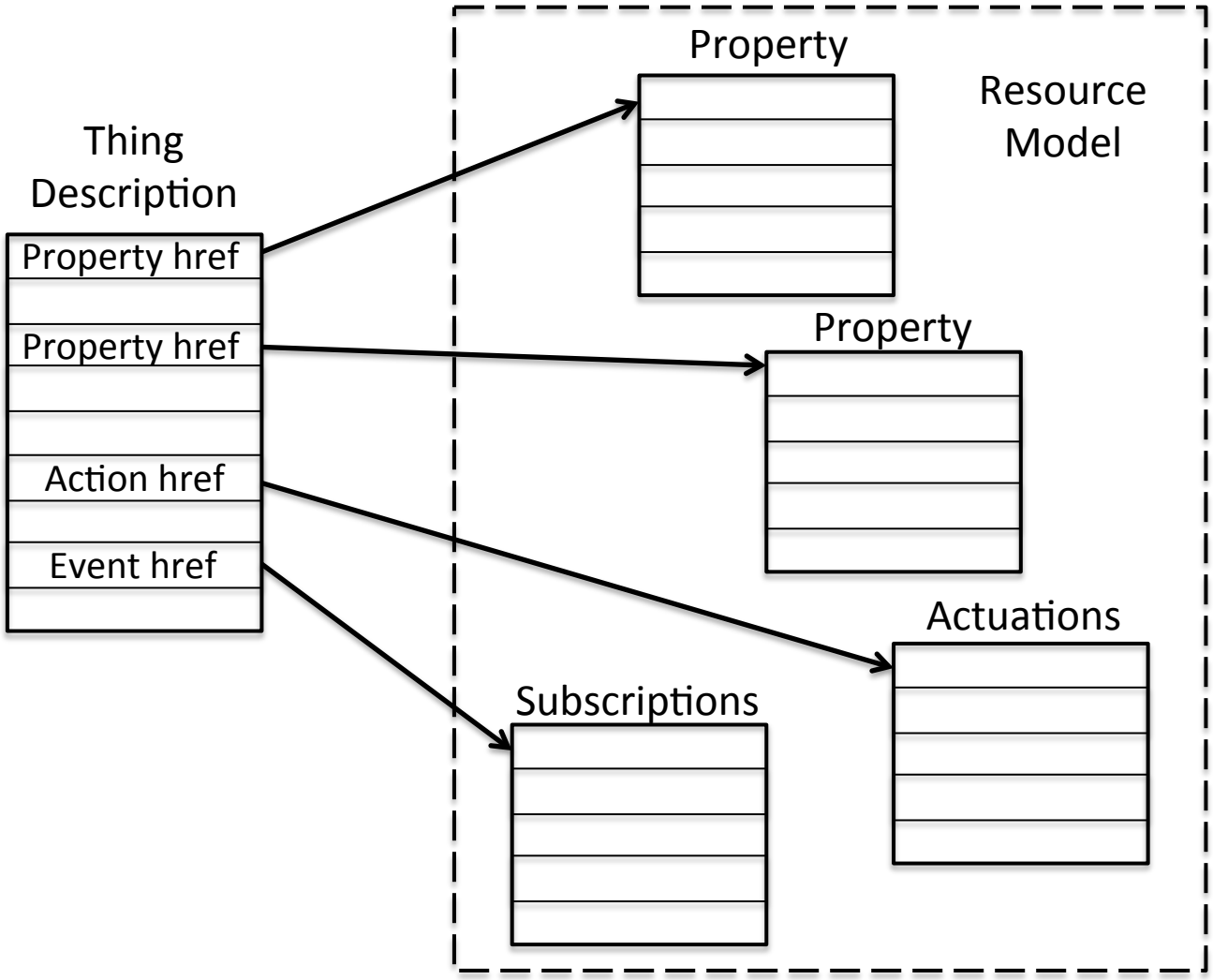
Use Case Requirements

- WoT Servient Pattern
- Exposes and Consumes resources described in Thing Description
  - Multiple Protocols; HTTP, CoAP, MQTT
  - Scripting API
  - Device Bridge
  - Storage Bridge
- Resource Collections
- Asynchronous Notification

WoT Servient Use Case



Resource Model Scope



Transfer Layer

Layer	Description
Application	Scripts that expose and consume resources, execute the "business logic" of things
Things	Thing Description, Stateful Resources
Transfer	REST, Pub-Sub: HTTP, CoAP, MQTT
Transport	UDP, TCP
Network	Ethernet, WiFi, 6LoWPAN, Thread

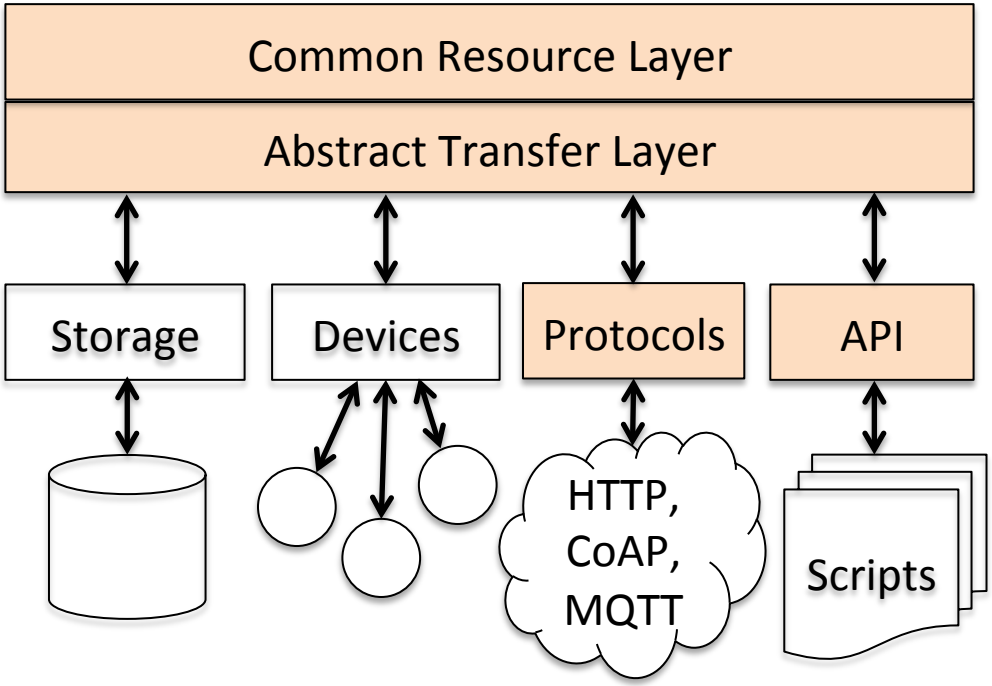
Common Transfer Semantics

- Define REST + Pubsub based transfer semantics for a common resource interaction model exposed by the"hrefs" pointed to in TD
- One model to map to HTTP, CoAP, MQTT using protocol bindings
- Instances of Events, Actions, Properties, and other entity classes in TD point to resources with well defined transfer semantics

Example Abstract Transfer Layer

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

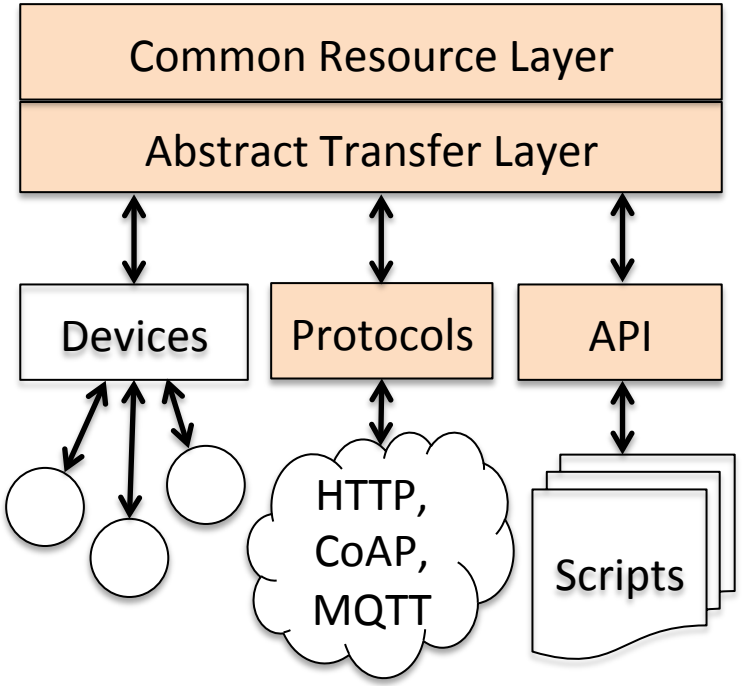
Architecture



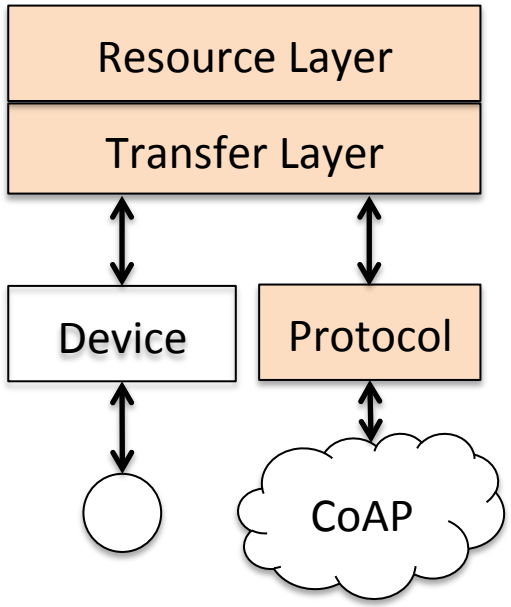
Servient Shell

- Resource Exposer
  - Constructor
  - Handler Interface
- Resource Consumer
  - Discovery
  - Caching Client
- TD Interface

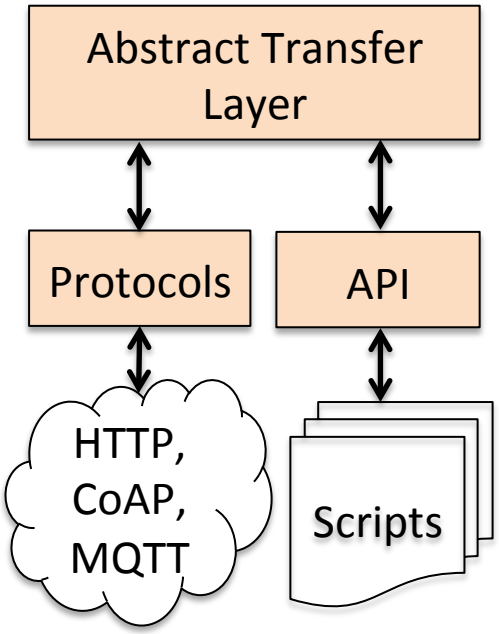
Device Server



Simple Device



Scripting Client



Milestones

- Define resource layer format – T2TRG draft
- Define transfer layer format – T2TRG draft
- Construct and expose WoT Interface resources using a TD template
- Discover and consume WoT Interface resources using a TD template
- Demonstration of control and orchestration of connected things through WoT Servients

Working Documents

https://github.com/hyperstate/hyperstate-docs