WoT Status Update and Next Steps

W3C WoT/ITRF T2TRG Workshop, 15 November 2019, Singapore

Michael McCool and Kunihiko Toumura

Outline

- W3C Web of Things
  - Interest and Working Groups’ charters, status, and goals
  - Recent deliverables
  - Work items and goals of renewed charters
- Discussion Items
  - OneDM SDF/RDF integration
  - Interoperability Profiles
  - Discovery
  - Coordination with T2TRG and CoRE Activity
- Resources for further reading and contact

Outline

- **W3C Web of Things**
  - **Interest and Working Groups’ charters, status, and goals**
  - **Recent deliverables**
  - **Work items and goals of renewed charters**
- Discussion Items
  - OneDM SDF/RDF integration
  - Interoperability Profiles
  - Discovery
  - Coordination with T2TRG and CoRE Activity
- Resources for further reading and contact

W3C Web of Things

Goal: Support IoT Interoperability via Open Standards

- **W3C WoT Interest Group (IG)**
  https://www.w3.org/2016/07/wot-ig-charter.html
  - Started spring 2015
  - ~200 participants
  - Informal work and outreach
  - “PlugFest” validation with running code
  - Exploration of new building blocks
  - “OpenDays” with external speakers
  - Liaisons and collaborations with other organizations and SDOs
  - ***Second Workshop on Web of Things held 3-5 June 2019 in Munich***
  - ***IG charter renewal accepted October 2019***

- **W3C WoT Working Group (WG)**
  https://www.w3.org/2016/12/wot-wg-2016.html
  - Started end of 2016 (effectively Feb 2017)
  - ~100 participants
  - Normative work on specific deliverables
  - W3C Patent Policy for royalty-free standards
  - Only W3C Members and Invited Experts
  - ***Architecture and Thing Description were (re)published as Candidate Recommendations on 6 November 2019***
  - ***Transition to Proposed Recommendations expected by December 2019***
  - ***Notes published on Protocol Bindings, Security, and Scripting API***
  - ***WG charter renewal in progress (draft); final version expected to be submitted Nov 20***

W3C Web of Things – Building Blocks

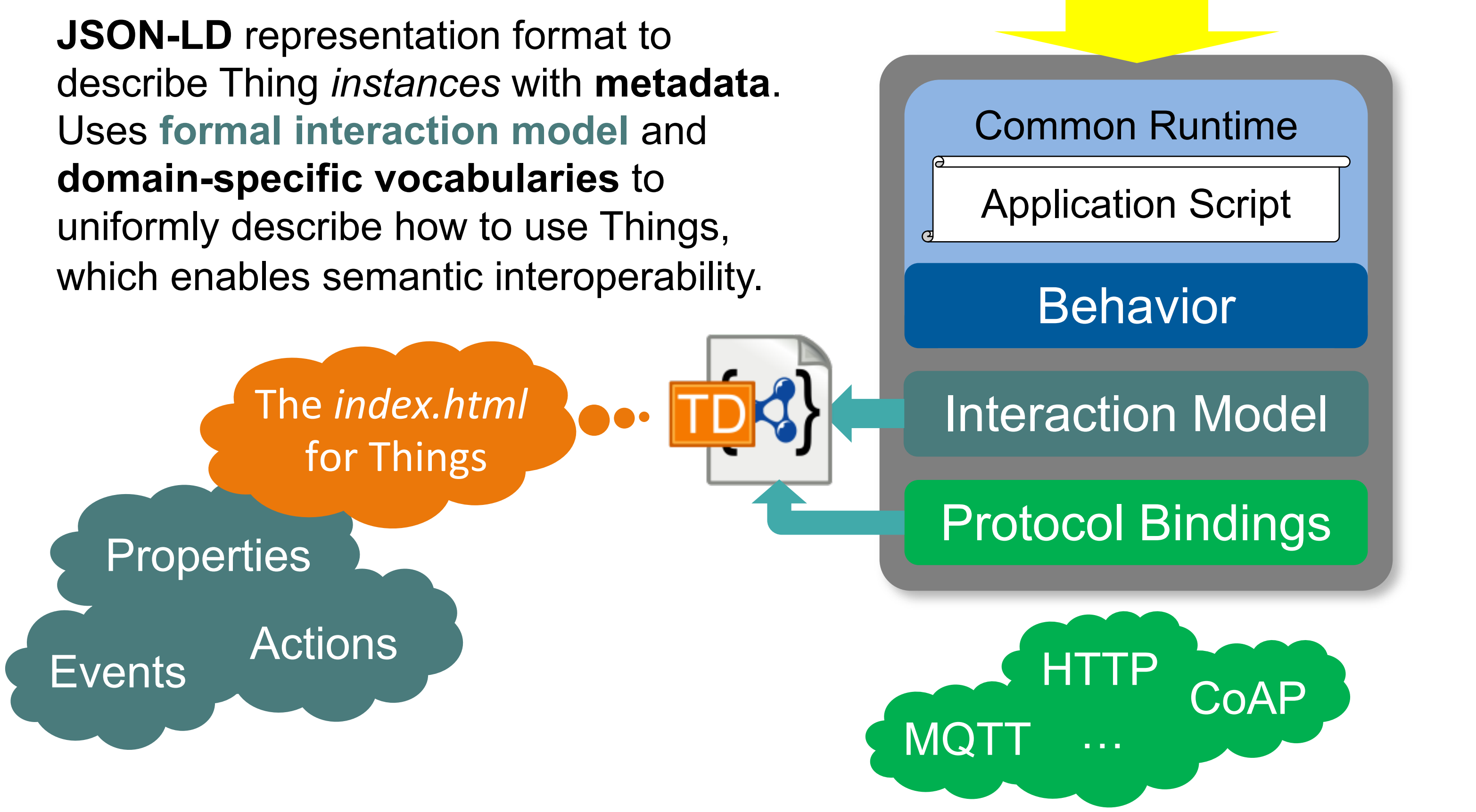
WoT Architecture

Overarching umbrella with architectural constraints and guidance on how to use and combine building blocks.

WoT Thing Description (TD)

JSON-LD representation format to describe Thing *instances* with **metadata**. Uses **formal interaction model** and **domain-specific vocabularies** to uniformly describe how to use Things, which enables semantic interoperability.

Security Guidelines



WoT Scripting API

Standardized **JavaScript** object API for an IoT runtime system **similar to the Web browser**. Provides an interface between applications and Things to simplify IoT application development and enable **portable apps** across vendors, devices, edge, and cloud.

WoT Binding Templates

Capture how the **formal Interaction Model** is mapped to concrete protocol operations (e.g., CoAP) and platform features (e.g., OCF). These templates are re-used by concrete TDs.

W3C Web of Things – Building Blocks

REC Track

WoT Architecture

Overarching umbrella with archite nstraints and guidance on h w to use and combine building blocks.

WoT Thing Description (TD)

REC Track

JSON-LD represent ion format to describe Thin s with **metadata**. Uses **forma** **model** and **domain-s** **ularies** to uniformly d v to use Things, which enable semantic interoperability.

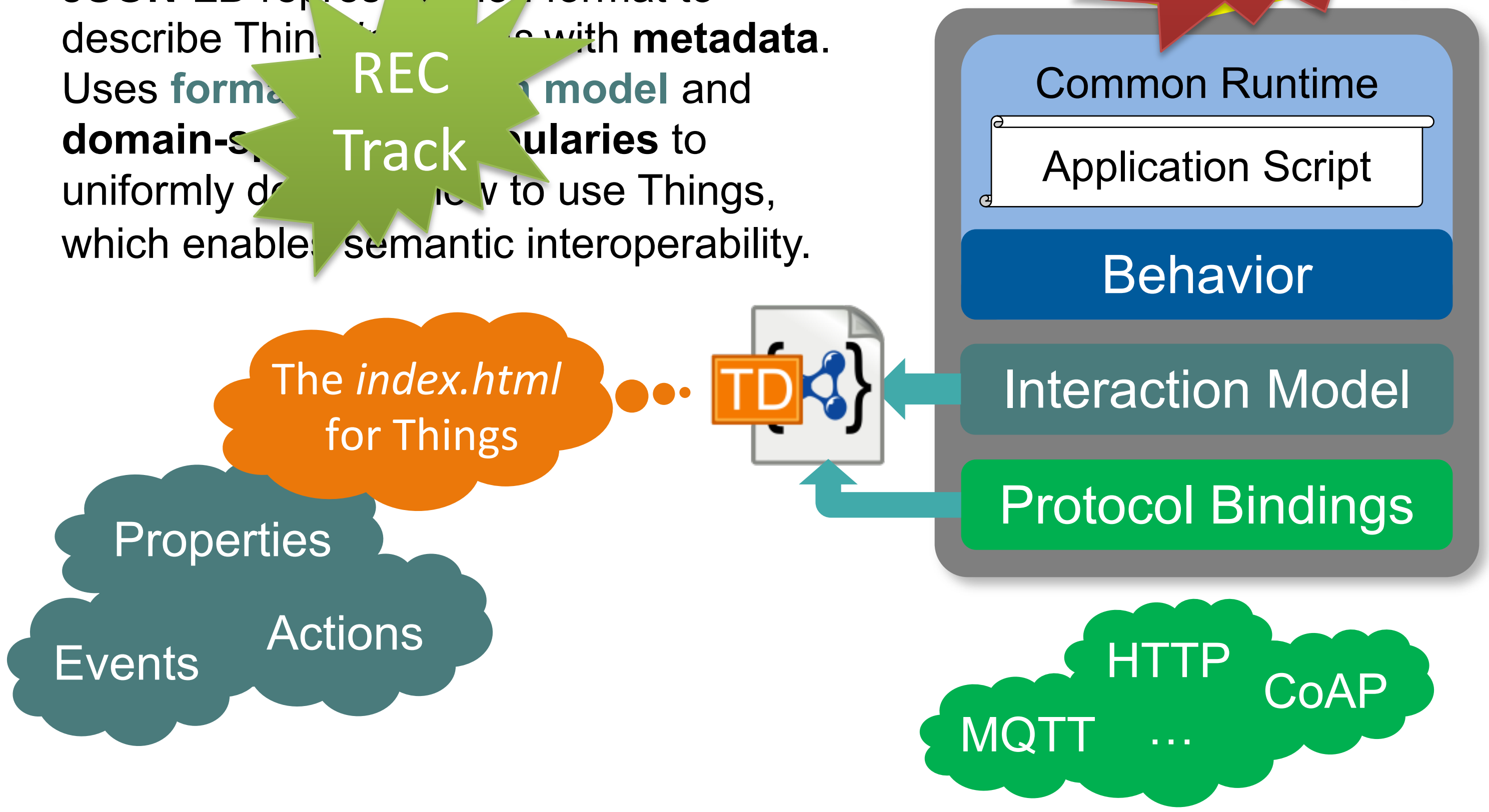
Securi

WG Note

WoT Scripting API

WG Note

Standardized **JavaScri** t object API for an IoT runti sy **ilar to the Web bro** nterface between ings to simplify Io evelopment and enable **rtable apps** across vendors, devices, edge, and cloud.



WoT Binding Templates

WG Note

Capture how the **fo** **nteraction Model** is map tocol operations (latform features (mplates are re-use e TDs.

Published Candidate Recommendations

WoT Architecture

- Constraints that define the difference between IoT and W3C WoT
- Definition of Interaction Affordances
- Definition of Web forms

- Use cases and requirements
- Terminology
- Interplay of W3C WoT building blocks
- Examples

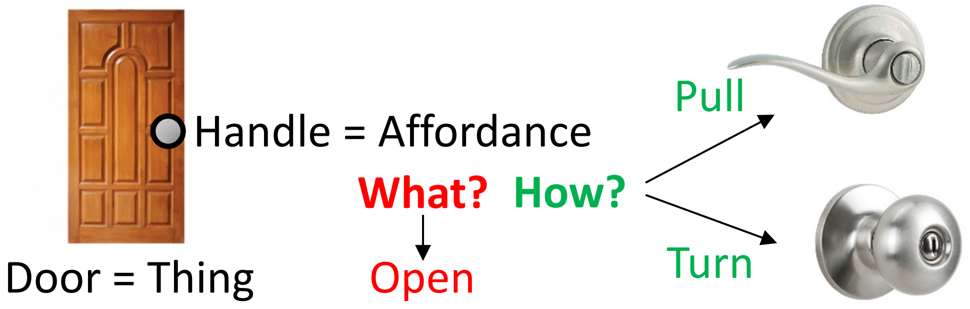
WoT Thing Description (TD)

- Information model & representation format for Thing metadata, generic data model, and hypermedia-based interface desriptions

- Namespace and vocabulary definitions
- Parsing and serialization rules
- Extension points
- Examples

Published Candidate Recommendations

- **WoT Architecture**
  - Constraints
    - Things must have TD (W3C WoT)
    - Must use hypermedia controls (general WoT)
      - URIs
      - Standard set of methods
      - Media Types
  - Interaction Affordances
    - Metadata of a Thing that shows and describes the possible choices (what) to Consumers, thereby suggesting how Consumers may interact with the Thing



- **WoT Thing Description (TD)**

```
{
  "@context": [
    "https://www.w3.org/2019/wot/td/v1",
    { "iot": "http://iotschema.org/" }
  ],
  "id": "urn:dev:org:32473:1234567890",
  "title": "MyLEDThing",
  "description": "RGB LED torchiere",
  "@type": ["Thing", "iot:Light"],
  "securityDefinitions": ["default": {
    "scheme": "bearer"
  }],
  "security": ["default"],
  "properties": {
    "brightness": {
      "@type": ["iot:Brightness"],
      "type": "integer",
      "minimum": 0,
      "maximum": 100,
      "forms": [ ... ]
    }
  },
  "actions": {
    "fadeIn": {
      ...
```

Published WG Notes

- **WoT Security and Privacy Guidelines**
  - Details beyond the security considerations in each specification for a holistic security and privacy configuration of Things
  - Security testing plan
- **WoT Binding Templates**
  - Documentation for how to describe existing IoT ecosystems (e.g., OCF or generic Web) with WoT Thing Description
- **WoT Scripting API**
  - Proposal for a standard API to consume and produce WoT Thing Descriptions
  - Provides interface between applications and network-facing API of IoT devices (cf. Web browser APIs)
  - Documents learnings from the design process

Status and Recent Developments

- Decision to adopt JSON-LD 1.1 proposed features to allow:
  - Default values
  - Object notation (name: value) instead of arrays
  - Alignment with common JSON practices
- Security metadata
  - Focus on HTTPS (Basic Auth, Digest, Tokens, OAuth2)
- Protocol Bindings
  - Focus on HTTP and structured payloads compatible with JSON
  - Support for Events also using subprotocols (e.g., long polling in HTTP)
- Extension Points
  - CoAP(S), MQTT(S), and further security schemes (e.g., ACE)
  - Semantic annotations with custom vocabularies (JSON-LD @context and @type)

Plugfest, Use Cases, and Demos

```
23: [EM] {room:cool}
24: [EM] {room:default}
25: [EM] {demo:init}
26: [EM] {program:started}
27: [EM] {room:bright}
28: [EM] {actorId:1234567890}
29: [EM] {room:quiet}
30: [E
31: [E
32: [E
33: [E
34: [E
35: [E
36: [E
37: [
```

WG Charter Proposal: Work Items

https://cdn.statically.io/gh/w3c/wot/master/charters/wot-wg-charter-draft-2019.html?env=dev

Architectural Requirements, Use Cases, and Vocabulary

- Understand and state requirements for new use cases, architectural patterns, and concepts.

Link Relation Types:

- Definition of specific link relation types for specific relationships.

Observe Defaults:

- For protocols such as HTTP where multiple ways to implement "observe" is possible, define a default.

Implementation View Spec:

- More fully define details of implementations.

Interoperability Profiles:

- Support plug-and-play interoperabilty via a profile mechanism
- Define profiles for specific application domains and use cases.

Thing Description Templates:

- Define how Thing Descriptions can defined in a modular way.

Complex Interactions:

- Document how complex interactions can be supported via hypermedia controls.

Discovery:

- Define how Things are discovered in both local and global contexts and Thing Descriptions are distributed.

Identifier Management:

- Mitigate privacy risks by defining how identifiers are managed and updated.

Security Schemes:

- Vocabulary for new security schemes supporting targeted protocols and use cases.

Thing Description Vocabulary:

- **Extensions to Thing** Description vocabulary definitions.

Protocol Vocabulary and Bindings:

- Extensions to protocol vocabulary definitions and protocol bindings.

Outline

- W3C Web of Things
  - Interest and Working Groups' charters, status, and goals
  - Recent deliverables
  - Work items and goals of renewed charters
- **Discussion Items**
  - **OneDM SDF/RDF integration**
  - **Interoperability Profiles**
  - **Discovery**
- Resources for further reading and contact

OneDM SDF/RDF Integration: Status

- WoT Thing Descriptions
  - Can be interpreted as JSON-LD 1.1
  - Can be imported into RDF databases
  - Support use of “extension vocabularies” via @context declarations
  - Do not *need* RDF to be processed;
    can also be interpreted as JSON in many use cases
- Have previously demonstrated
  - Use of iotschema.org vocabularies
  - Keyword-based search using a Thing Directory
  - RDF-based semantic search using a Thing Directory
  - Template-based search in a Thing Directory
- Interaction model is consistent with SDF
  - Properties, events, actions
- Use of similar data schemas based on JSON Schema

OneDM SDF/RDF Integration: Discussion

Discussion:

- TDs describe Thing instances
- OneDM SDF seems to describe Thing "types"
  - TDs support "links" so we could reference SDF URL sources as "types" of Things
- SDF can (probably) be compiled to RDF and used as JSON-LD extension
  - Vocabulary (distinct from types) can could be referenced using @context
- Semantic search etc. can then be supported in WoT TD Discovery
  - However, do not NEED to use RDF, SPARQL, etc
  - Other options possible (template matching, keyword search, etc).
- Need to align data schema definitions, etc.
  - Ideally with JSON Schema alignment
- How to validate use of an extension vocabulary?

Interoperability Profiles: Problem

- TDs are “open” and can describe affordances for any protocol meeting some basic requirements (URL scheme, etc)
- Extension vocabulary also allows additional security schemes, etc.

BUT

- This openness works against interoperability:
  - A consumer does not have a finite set of protocols that it needs to implement to be sure of being able to consume any Thing Description and operate with any Thing
- It also does not encourage best practices:
  - “Bad” Things can be described as well as “Good” Things

Interoperability Profiles: Goals

- Profiles to define a subset of TDs (and Things) that
  - Precisely define the implementation requirements for a consumer
  - Limited (finite) set of protocols and security schemes
  - Finite resource requirements (eg max string lengths, etc)
  - Support best practices (for example, disallow insecure combinations of security schemes and protocols)
- Requirements still under active discussion
  - What protocols
  - What security schemes
  - What best practices
  - How to signal use of a profile?
  - Is more than one profile possible?
  - How to deal with semantic extensions?

Discovery: Requirements

- Capabilities
  - Support local and global discovery
  - Support semantic queries
  - Support directories
  - Support peer-to-peer (self-identifying) discovery
  - Preserve privacy
- Privacy-Preserving Architecture
  - Support device and information lifecycle, trust management, access controls
  - Distribute TDs only to authenticated and authorized users
  - Don’t leak metadata or queries to unauthorized entities
- Alignment with existing standards
  - E.g. IETF CoRE Resource Directories, CoRE Link Format, DID, DNS-SD, DNS-SRV, DHCP, etc…
- Optional:
  - Support for Scripting Discovery API

Discovery: Two-Phase Proposal

1. **Introduction:** First Contact Mechanism
   - Obtain address of directory service - *only*
   - Address should not leak any other metadata, eg type of devices
   - Can have multiple mechanisms for introduction
     - Local: QR code. mDNS, DNS-SD, DHCP, QR code, Bluetooth beacon, etc.
     - Global: search engine
     - Self: Well-known addresses, eg ".well-known/td"
2. **Exploration:** Metadata Retrieval Mechanism
   - Mutual authentication and secure connection required, and then…
   - Queryable Directory service
     - Lightweight: specific query parameters, eg. location, keywords
     - Full: (sub-)SPARQL semantic query AND/OR GraphQL AND/OR by-example (templates)
   - External service: need registration sub-API, timeouts, access control, etc.
   - Self (peer-to-peer): same query API, but no public registration API
   - Mutable IDs → need way to notify registered users of changes

Coordination with T2TRG and CoRE Activity

For new WG charter work items, WG has begun collecting use cases and extracting requirements.

The workshop/hackathon is a good opportunity to investigate how to interwork with T2TRG/CoRE, as a case study for the work items.

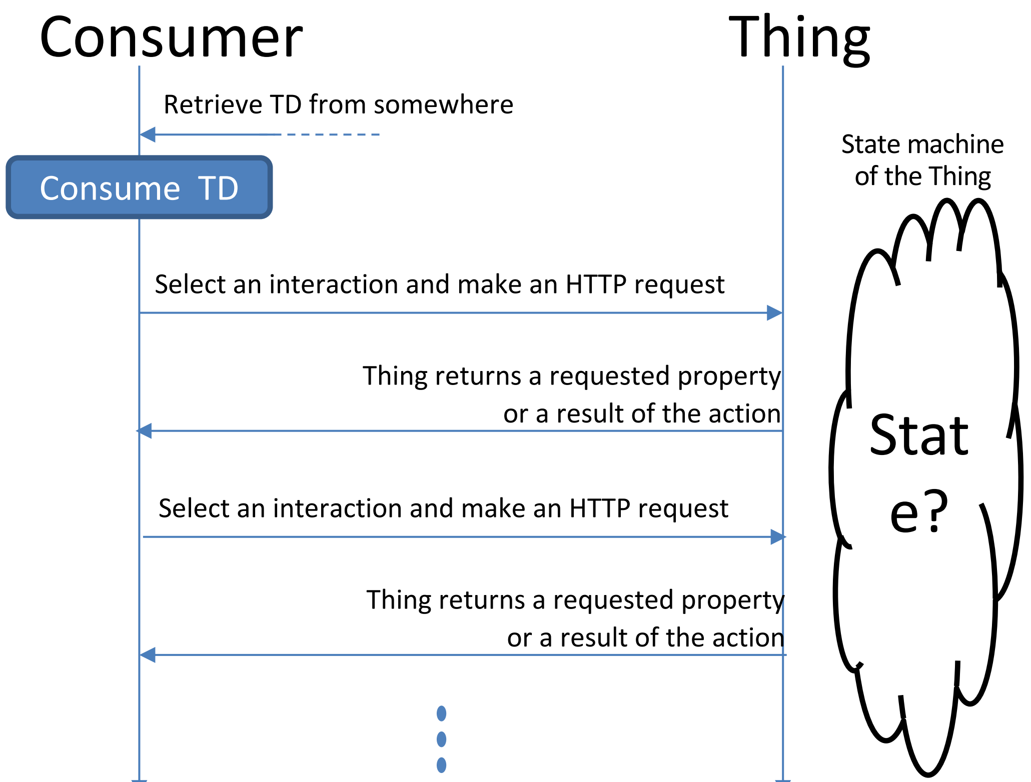
Current Charter Draft:
  - http://w3c.github.io/wot/charters/wot-wg-charter-draft-2019.html

- 2.5 Complex Interactions
  - How to describe a CoRAL-based device?
- 2.6 Discovery, 2.8 Onboarding
  - We can use a CoRE-RD framework for one of discovery introduction mechanisms to retrieve a pointer to a Thing Directory?
  - Can we (and should we) use CoRE-RD to directly retrieve WoT Thing Descriptions?
- 2.10 Security Schemes
  - Use of OSCORE, ACE etc.
- 2.12 Protocol Vocabulary and Bindings, 2.13 Observe Defaults
  - Use of CoAP Observe

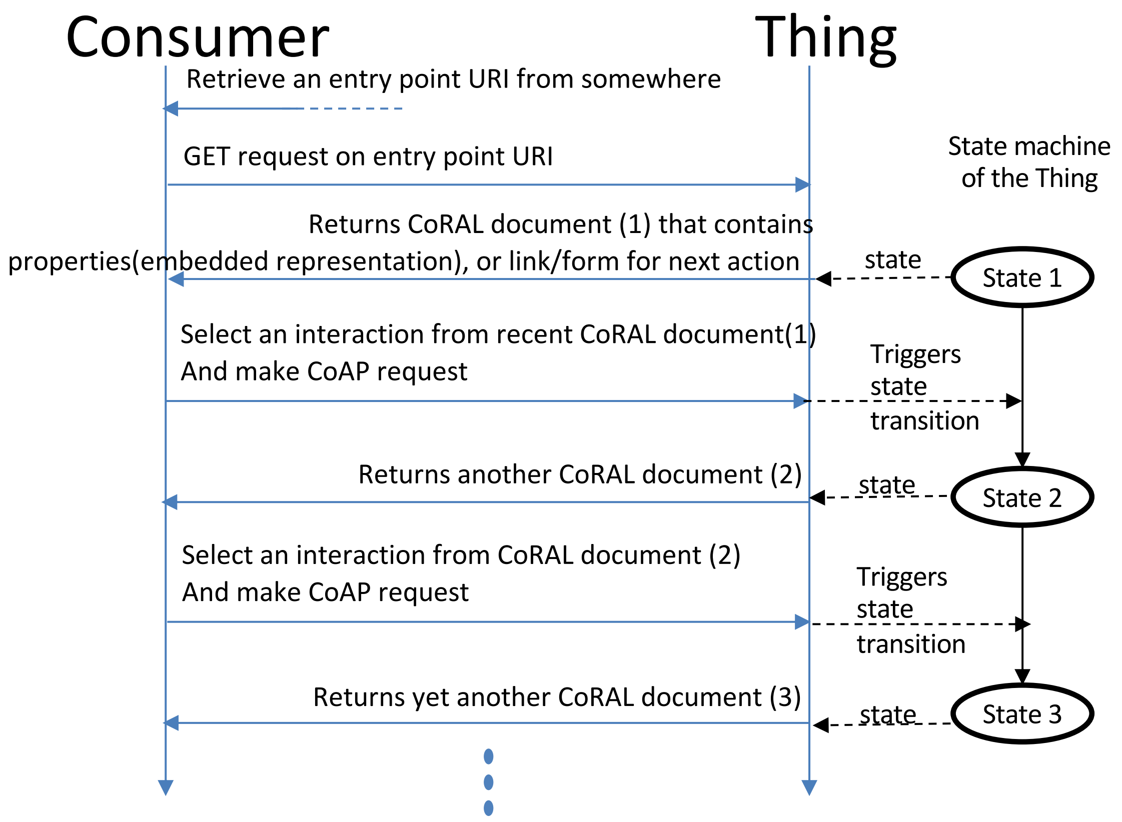
Complex Interactions (1/2): Differences

- (Current) WoT Thing Description contains all interactions that a thing provides, regardless of current state of the Thing.

- CoRE/CoRAL takes pure HATEOAS approach, i.e. an CoRAL document may contain interactions that are meaningful in current state.



Consumer
Thing
Retrieve TD from somewhere
Consume TD
State machine of the Thing
Select an interaction and make an HTTP request
Thing returns a requested property or a result of the action
Stat e?
Select an interaction and make an HTTP request
Thing returns a requested property or a result of the action



Consumer
Thing
Retrieve an entry point URI from somewhere
GET request on entry point URI
State machine of the Thing
Returns CoRAL document (1) that contains properties(embedded representation), or link/form for next action
state
State 1
Select an interaction from recent CoRAL document(1) And make CoAP request
Triggers state transition
Returns another CoRAL document (2)
state
State 2
Select an interaction from CoRAL document (2) And make CoAP request
Triggers state transition
Returns yet another CoRAL document (3)
state
State 3

Complex Interactions (2/2): Approaches

- How to interact with a pure RESTful-style Thing from a WoT Consumer?
  1. Static TD
     - Introduce a concept of a “State Machine” to Thing Description
     - TD describes all possible interactions of the Things.
     - Designate feasible interactions of each state using an explicit state machine.
     - Interaction with a Thing triggers a state transition in Consumer’s state machine.
  2. Dynamic TD:
     - Consumer updates its internal Thing Description dynamically, based on a response from the Thing
     - Based on recent Thing Description, an application select a feasible interaction for the current state.
  3. Hypermedia controls
     - TD describes only initial static entry points to API
     - Initial interactions with entry points return additional hypermedia controls
       - Thing Descriptions (or Action Descriptions, or Event Descriptions) that provide additional interactions

Discovery with CoRE

- CoRE proposes several discovery and directory mechanisms.
  - Discover Things by CoAP multicast on well-known URI (/.well-known/core)
    - Discovery of Resource Directory is similar approach (i.e. use "/.well-known/core/?rt=core.rd*")
  - Moreover, IPv6 Authoritative Border Router Option, DHCP, DNS-SRV that can also be used for discovery of Things and Directory Services

- Can we (and should we) use these mechanisms to retrieve Thing Descriptions directly (combined introduction and exploration)?
- Can we (and should we) just use these mechanisms to retrieve links to Thing Description Directory services (introductions only)?
- If we just provide links, when should we augment them with metadata?
- What kind of privacy and security protections are provided by these mechanisms?
- What kind of query parameters are needed (location, keyword, filtering, etc)?

Ideas for IETF Hackathon

- Implementing WoT TD discovery based on CoRE-RD.
  - Discover Resource Directory based on multicast
    - coap://(*"all CoRE resource directories" address)*/.well-known/core?rt=core.rd*
  - Combined Introduction and Exploration
    - Register TD to CoRE-RD
      - POST coap://rd.example.com/rd?ep=(Thing-ID)
      - Content-Format: application/td+json (Content-format ID is t.b.d., in the 256-9999 range)
    - Retrieve TD from CoRE-RD
      - GET coap://rd.example.com/rd-lookup?rt=MyLampThing
      - Accept: application/td+json
  - Introduction Only
    - Register TD Directory service link with CoRE-RD
      - With what link relation type?
    - Retrieve TD Directory service link from CoRE-RD
    - Authenticate with TD Directory service
    - Execute query to search for and retrieve TDs

    OR
    - Bypass CoRE-RD and use .well-known approach to discover TD Directory service directly?
  - Using retrieved TD, interact with the Thing (e.g. generate Node-RED node, etc.)
    - Interaction protocol: HTTP(S), CoAP(S)

Outline

- W3C Web of Things
  - Interest and Working Groups' charters, status, and goals
  - Recent deliverables
  - Work items and goals of renewed charters
- Discussion Items
  - OneDM SDF/RDF integration
  - Interoperability Profiles
  - Discovery
  - Coordination with T2TRG and CoRE Activity
- **Resources for further reading and contact**

W3C WoT Resources

- W3C WoT Wiki
  - https://www.w3.org/WoT/IG/wiki
    (IG/WG organizational information)

- W3C WoT Interest Group
  - https://www.w3.org/2016/07/wot-ig-charter.html
    (old charter)
  - https://www.w3.org/2019/10/wot-ig-2019.html
    (new charter)
  - https://lists.w3.org/Archives/Public/public-wot-ig/
    (mailing list)
  - https://github.com/w3c/wot
    (technical proposals)

- W3C WoT Working Group
  - https://www.w3.org/2016/12/wot-wg-2016.html
    (old charter)
  - https://cdn.statically.io/gh/w3c/wot/master/charters/wot-wg-charter-draft-2019.html?env=dev
    (new charter draft)
  - https://www.w3.org/WoT/WG/
    (dashboard)

- W3C WoT Candidate Recommendations
  - https://www.w3.org/TR/wot-architecture/
  - https://www.w3.org/TR/wot-thing-description/

- W3C WoT Working Drafts / Group Notes
  - https://www.w3.org/TR/wot-binding-templates/
  - https://www.w3.org/TR/wot-scripting-api/
  - https://www.w3.org/TR/wot-security/

- W3C WoT Editors' Drafts and Issue Tracker
  - https://github.com/w3c/wot-architecture/
  - https://github.com/w3c/wot-thing-description/
  - https://github.com/w3c/wot-binding-templates/
  - https://github.com/w3c/wot-scripting-api/
  - https://github.com/w3c/wot-security/
  - https://github.com/w3c/wot-security-best-practices/
  - https://github.com/w3c/wot-profile/

- Reference Implementations and Tools: node-wot
  - node-wot: https://github.com/eclipse/thingweb.node-wot
  - TD playground: https://github.com/thingweb/thingweb-playground

Contacts

https://www.w3.org/WoT/WG/

Dr. Michael McCool

Principal Engineer

Intel

Technology Pathfinding

michael.mccool@intel.com

Dr. Sebastian Kaebisch

Research Scientist

Siemens

Corporate Technology

sebastian.kaebisch@siemens.com