Outline

• What is WoT?
  • Applying and extending web standards for IoT
  • Descriptive interoperability
  • Current status of deliverables

• Recent Activity
  • Plugfest
  • Commercial usages
  • Discovery/directory implementations
  • Relationship to IETF activity
  • Items under discussion
  • New charters/new deliverables
W3C Web of Things (WoT)

- W3C Working Group goal: Adapting web technologies to IoT
- Already published: Thing Description (TD) metadata format
  - TD describes the available interactions (network API) of a Thing
- New standards work in progress, including Discovery
  - How does a potential user obtain the TDs for a Thing?

![Diagram showing WoT API and protocols]

http
MQTT
JSON
OAUTH2
Modbus
KNX
TLV
WS
BACnet
XML
ECHONET
OPC-UA
Bearer
EXI
AMQP
WoT Descriptive Interoperability

WoT Architecture

• Constraints
  • Things must have a TD
  • Must use hypermedia controls (general WoT)
  • URIs, standard set of methods, media types

• Thing Description Affordances
  • Describes WHAT the possible choices are
  • Describes HOW to interact with the Thing

WoT Thing Description (TD)

```json
{
"@context": [
{"@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": {
...
}}
```
Current Status

New/Updated Normative Documents in Draft Status:

- Architecture 1.1: https://github.com/w3c/wot-architecture
- Thing Description 1.1: https://github.com/w3c/wot-thing-description
- Discovery: https://github.com/w3c/wot-discovery
- Profiles: https://github.com/w3c/wot-profile

New/Updated Informative Documents in Draft Status:

- Binding Templates: https://github.com/w3c/wot-binding-templates
- Scripting API: https://github.com/w3c/wot-scripting-api
- Use Cases and Requirements: https://github.com/w3c/wot-usecases

Marketing Improvements:

- New Web Site, Animation, Resources: https://www.w3.org/WoT/
Recent Activity

• Plugfest
  • Projects: [https://github.com/w3c/wot-testing/labels/Plugfest%202021.09](https://github.com/w3c/wot-testing/labels/Plugfest%202021.09)

• New Commercial Usages
  • Takenaka Construction – Smart Building Information Management systems
  • Netzo – IoT dashboards and device management

• Directory Implementations
  • WoT Hive, LogiLab (SPARQL based), Fraunhofer LinkSmart

• IETF Relationships: JSON Path, CoreRD, COSE/JOSE, ASDF

• Under Discussion (IG Notes expected)
  • Geospatial data, Embedded JSON Signatures

• New Charters/New Deliverables
Contacts

https://www.w3.org/WoT

Dr. Michael McCool
Principal Engineer
Intel
Technology Pathfinding
michael.mccool@intel.com

Dr. Sebastian Kaebisch
Senior Key Expert
Siemens
Technology
sebastian.kaebisch@siemens.com
Backup
WoT Orchestration

Node-RED/node-gen

node-wot/Scripting API

WoTHelpers.fetch( "coap://localhost:5683/counter" ).then(async (td) => {
  // using await for serial execution (note 'async' in then() of fetch())
  try {
    let thing = await WoT.consume(td);
    console.info("=== TD ===");
    console.info(td);
    console.info("==========");

    // read property #1
    let read1 = await thing.readProperty( "count" );
    console.info("count value is", read1);

    // increment property #1 (without step)
    await thing.invokeAction( "increment" );
    let inc1 = await thing.readProperty( "count" );
    console.info("count value after increment #1 is", inc1);

    // increment property #2 (with step)
    await thing.invokeAction( "increment", {step: 3} );
    let inc2 = await thing.readProperty( "count" );
    console.info("count value after increment #2 (with step 3) is", inc2);

    // decrement property
    await thing.invokeAction( "decrement" );
    let dec1 = await thing.readProperty( "count" );
    console.info("count value after decrement is", dec1);
  }
  catch(err) {
    console.error("Script error:", err);
  }
})
.catch((err) => { console.error("Fetch error:", err); });
Current WoT WG Charter Work Items

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 interoperability via a profile mechanism
• Define profiles that allow for finite implementability

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.