IoT and Semantic Interoperability

Michael McCool, Principal Engineer, Intel

IETF100, November 14, 2017

Agenda

What is semantic interoperability?

Why do we care, in the context of IoT?

How we implement it?

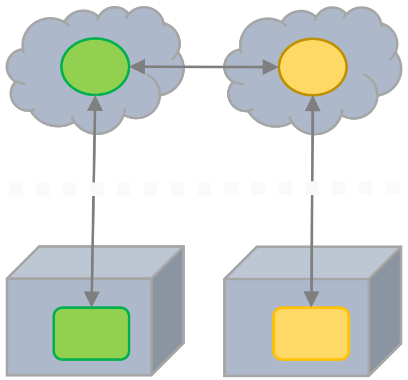
Now some recent activity…

- Web of Things – Thing Description and Thing Directory (semantic discovery)
- iotschema.org – IoT ontology
- OCF metadata bridge – translate OCF metadata to TDs
- AVS integration (in progress) – translate AVS intents to IoT ontology
- Fog service integration (planned) – support services as discoverable “things”

Evolution of the IoT Ecosystem

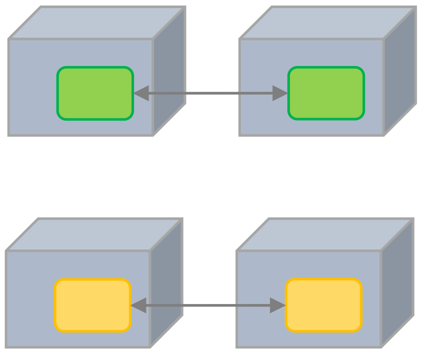
Cloud Supported IoT Devices

IoT devices connect to mostly independent cloud services



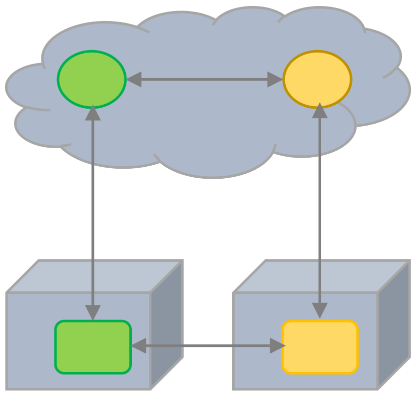
Locally Networked IoT Devices

IoT devices connect to others in isolated ecosystems



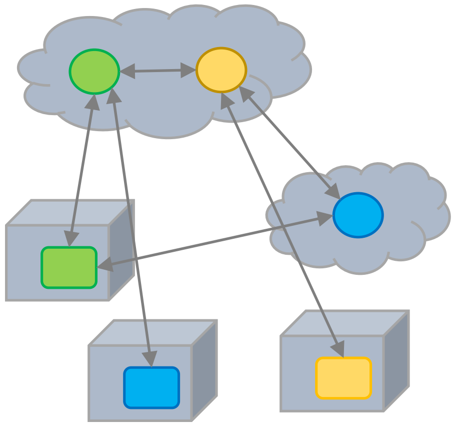
Fog Supported IoT Devices

IoT devices can discover and connect to local “fog services” that facilitate interop.



Ambient Service Ecosystem

Services composed from dynamically discovered devices and services



Ambient Standards Stack

Need to integrate *complementary* standards that promote *interoperability*

Management Services (ex: OpenFog)

- Defines platform for **federated edge and cloud services**
- Common services for discovery, security, monetization, updates, etc.

Metadata Standards (ex: WoT, iotschema.org, Haystack, etc.)

- Enable cross-ecosystem service discovery and composition
- **Supports semantic interoperability** which **enhances usability**

Device Standards (ex: OCF, MQTT, LwM2M, AWS IoT, etc.)

- Define standard local interfaces for devices and services

What?

What is interoperability?

Interoperability: What is it?

- Two entities can be interfaced “directly”
  - Web browser can access a web server and render content from it
    - Despite my never having visited that site before
  - I can plug my laptop into the wall to charge
  - I can replace lightbulbs in my apartment
  - I can buy bolts and nuts and expect them to work together
- For the IoT:
  - I want to use devices (and data from those devices) from different manufacturers in the same system

Levels of Interoperability

Semantic: *Be able to decode the meaning of data.*

- Convert to a representation that has tagged elements with identifiers, properties, and relationships using elements rooted in shared contexts, vocabularies, and ontologies.

Structural: *Be able to decode the organization of data.*

- Convert data into a set of common primitive types (numbers, characters, etc) structured with a set of organizational patterns (lists, sets, arrays, links, objects, tables, etc).

Syntactic: *Be able to decode the encoding of data.*

- Convert data in a consistent way between a serialized (transmissible, storable) representation (eg a sequence of bytes) and an internal data structure (eg a parse tree).

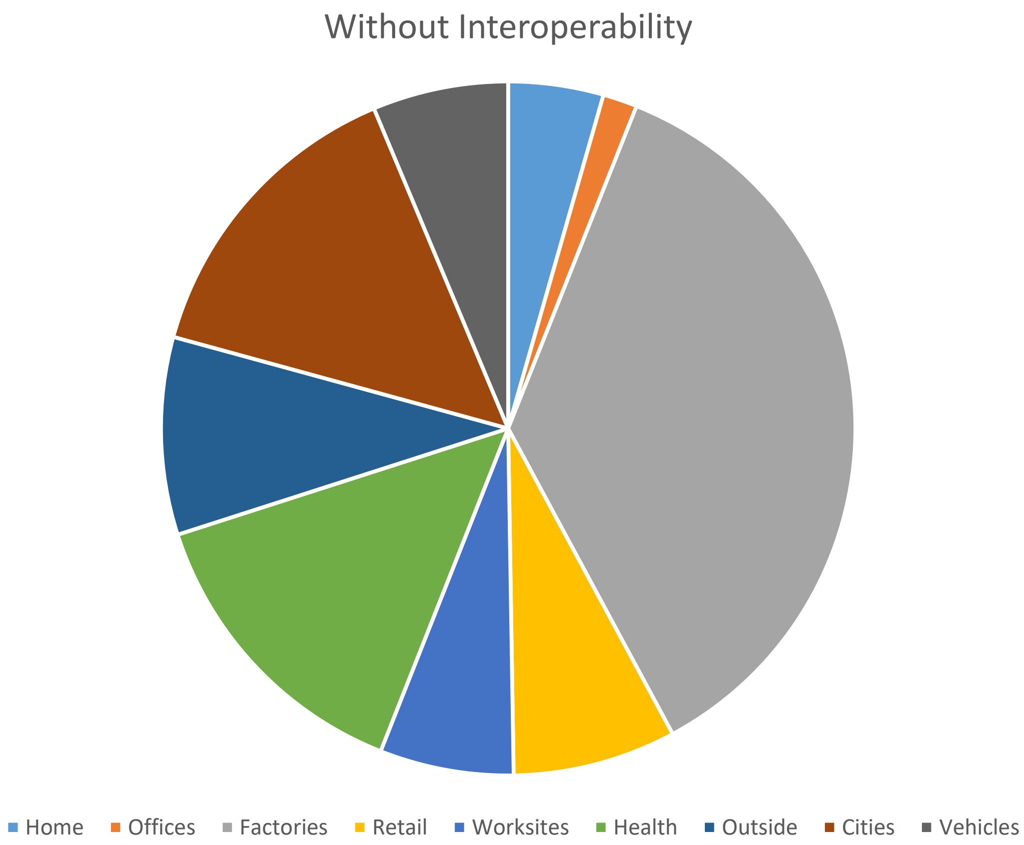
Why?

Why do we want interoperability?

Business Value of Interoperability

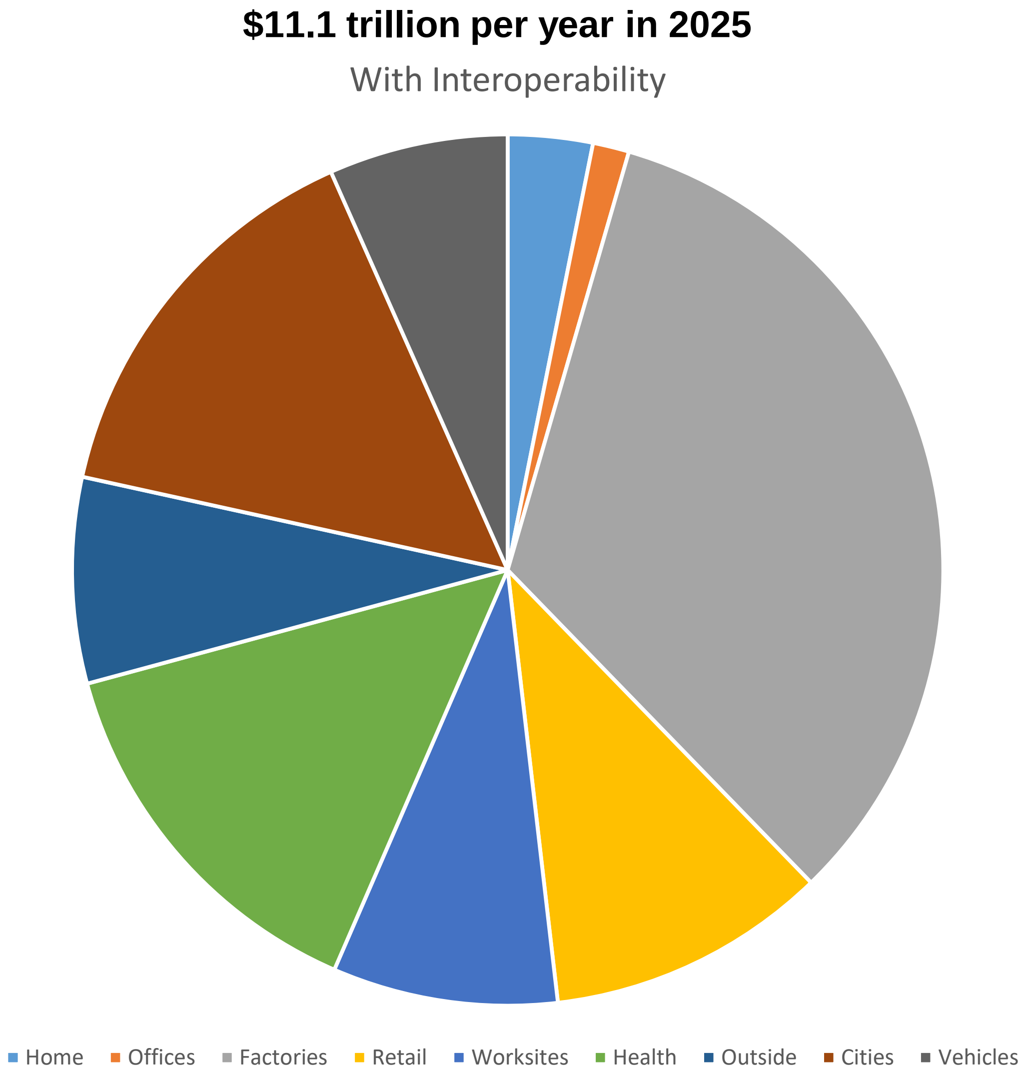
- “Interoperability between IoT systems is critically important to capturing maximum value; on average, interoperability is required for 40% of potential value across IoT applications and by nearly 60% in some settings.” [+$4.1B TAM by 2025]
- McKinsey & Company, *The Internet of Things: Mapping the Value Beyond the Hype*, 2015

$6.5 Trillion per year in 2025



Business Value of Interoperability

- “Interoperability between IoT systems is critically important to capturing maximum value; on average, interoperability is required for 40% of potential value across IoT applications and by nearly 60% in some settings.” [+$4.1B TAM by 2025]
- McKinsey & Company, *The Internet of Things: Mapping the Value Beyond the Hype*, 2015



How?

How to achieve interoperability?

iotschema.org

- Project to develop *simple* and *common* IoT ontologies for semantic interoperability

Scope in the Standards Stack

Functional Layer	Documents	Governing body
Ontologies and Taxonomies	SAREF, SSN, IoT-Lite, Haystack, QUDT	Various
Semantic Interoperability	Simple schemas and common definitions	iot.schema.org community
Modeling and Description Language	Thing Description High Level API	OCF, W3C Web of Things WG
Device Definitions	Clusters, Resource types, Device Types	OCF, Zigbee Alliance (dotdot)
Transfer Protocols	CoAP, Zigbee	IETF, Zigbee
Networks and Transports	IPV6, TCP, UDP, Thread, ZIgbee	IETF, Thread, Zigbee

Principals: Michael Koster (Samsung/Smart Things), David Janes (IOTDB), Darko Anicic (Siemens), Dan Brickley (Google)

See: iotschema.org site, discussion doc, and meeting minutes

IoT Service Semantics as Capability Bundles

Bundle theory, originated by the 18th century Scottish philosopher David Hume, is the ontological theory about objecthood in which an object consists only of a collection (bundle) of properties, relations or tropes.

As opposed to substance theory…

"Things" considered as

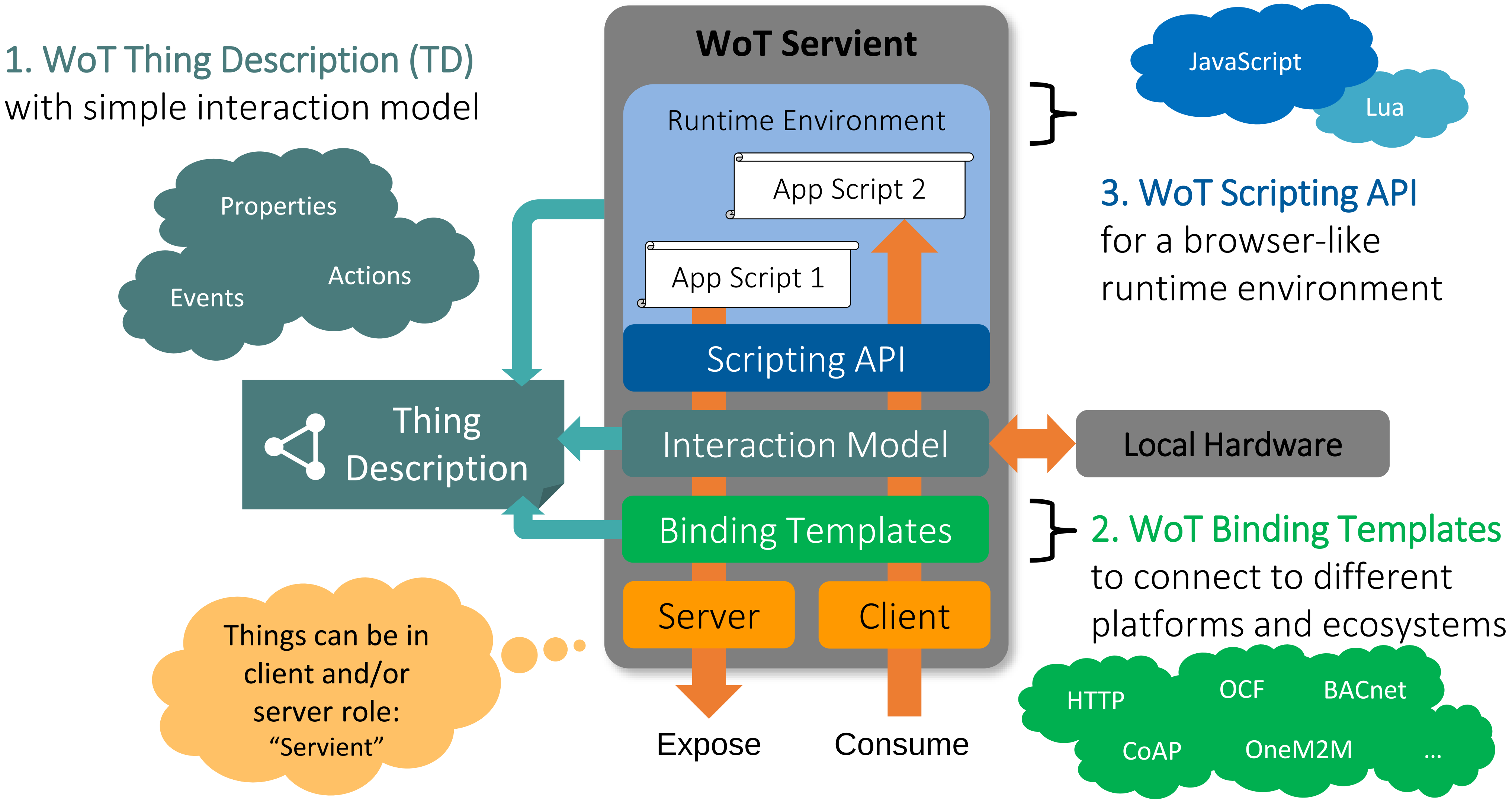
- Bundles of Capabilities
- Capabilities support specific sets of properties, actions, and events
- Services support sets of Capabilities

→ This is a more flexible model than just having definitions of particular Services or Things

→ Ideally we want a way to *model* novel devices, not just *select* them from a list

  → Capabilities allow us to model a novel device by selecting (and perhaps parameterizing) a set of capabilities

Web of Things Architecture



Thing Description Example

JSON-LD (Linked Data)

W3C WoT TD vocabulary

Domain-specific vocabulary

```
{   "@context":["http://w3c.github.io/wot/w3c-wot-tdcontext.jsonld",
                "http://w3c.github.io/wot/w3c-wot-common-context.jsonld",
                {"iot":"http://iotschema.org/"},
                {"http":"http://www.w3.org/2011/http/"}],
    "base":"http://example.ocfgateway.net/api/oic",
    "@type":["Thing","Light","iot:LightControl", // Capabilities
             "iot:Actuator","iot:BinarySwitch"],
    "name":"Intel-OCF-Smart Home LED (2relay)",
    "interaction":[
        {"name":"Switch Status",
         "@type":["Property","OnOffState","iot:SwitchStatus"], // Interactions
         "link":[
             {"href":"/a/led2relay?di=79683ab5-8df1-4b7a-b110-c1b8fe251e7d",
              "mediatype":"application/json",
              "http:methodName":"http:post", // Methods (specific to protocol binding)
              "rel":"setProperty"},
             {"href":"/a/led2relay?di=79683ab5-8df1-4b7a-b110-c1b8fe251e7d",
              "mediatype":"application/json",
              "http:methodName":"http:get",
              "rel":"getProperty"}
          ],
          → Payload structure and semantics
         },
         → Other interactions: Properties, Events, and Actions
     ]
}
```

Thing Description Example

// Payload structure and semantics for a Property interaction

```
"inputData":{
    "type":"object",
    "fields":[
        {"name":"value",
         "value":{"@type":["iot:Toggle"],  // Data
                  "type":"boolean”}
        }
    ]},
"outputData":{
    "type":"object",
    "fields":[
        {"name":"value",
         "value":{"@type":["iot:Toggle"],
                  "type":"boolean”}
        }
    ]}
```

JSON Schema

Now!

Proof of Concept Work…

Proof of Concept Development Goals

- Understand *the relationships of different technologies*
- Identify *technology and standards gaps*
- *Overcome specific technical obstacles and gaps*
- Demonstrate *the business value of new technologies*
- *Test integration patterns for multiple technologies*
- Advance *engagements with key ecosystem players*

Stages

1. **Metadata Bridging** (value: increase adoption and applicability)

   → Support *all* existing IoT devices from *multiple* ecosystems

   - Rather than trying to bridge data peer-to-peer, bridge *metadata*
   - Supports end-to-end security since data translation can be pushed to secure endpoints

2. **Semantic Voice Control** (value: demonstrate utility, engage key ecosystem player)

   → Support *any* IoT device with adaptive semantic voice controls

3. **Fog Integration** (value: develop support for ubiquitous localized services)

   → Deploy using local compute resources for proxies, translators, and acceleration

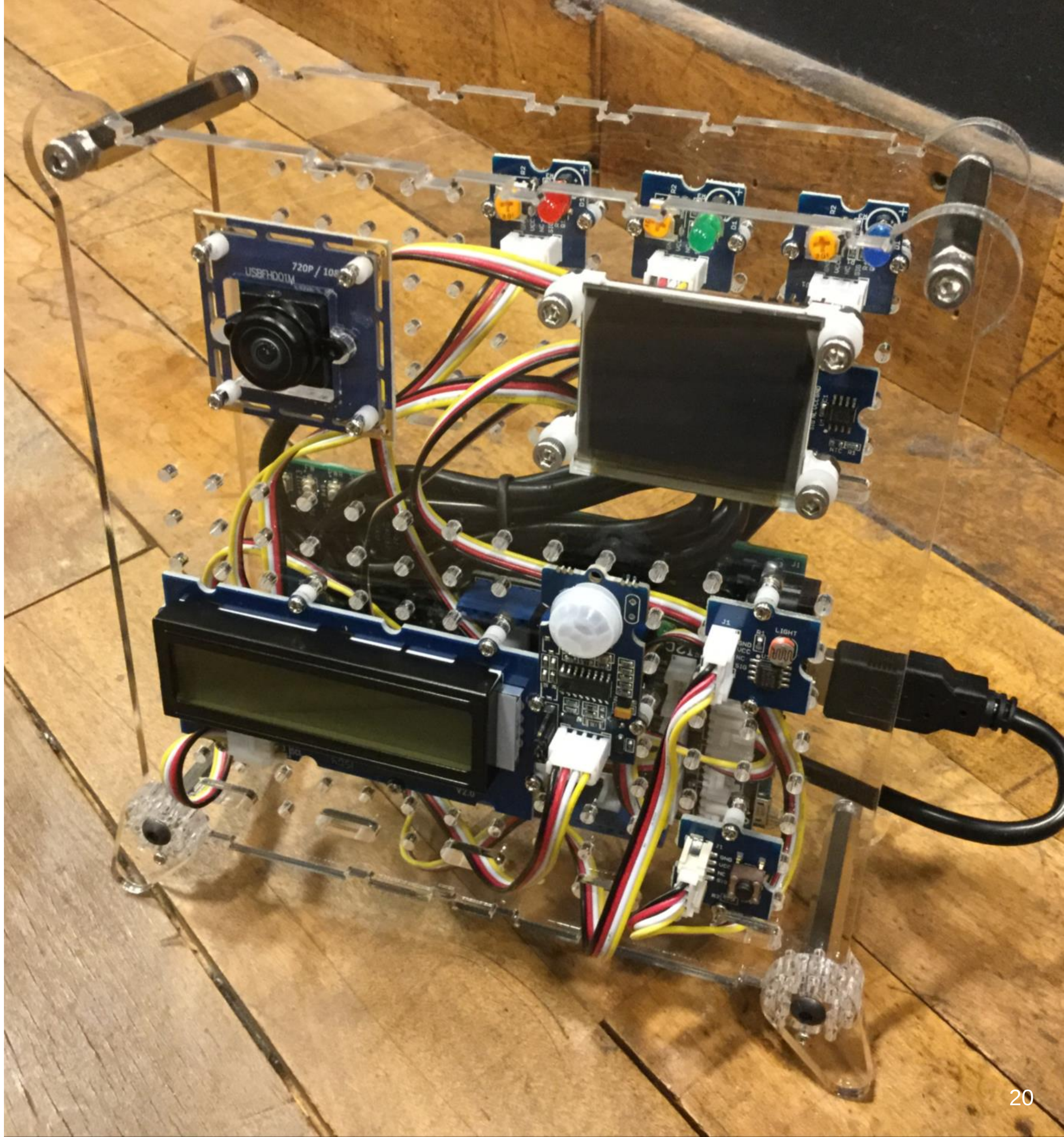
4. **Service-Oriented Development System** (value: support ecosystem development)

   → Support development and deployment of code for services, not (just) devices

1. Metadata Bridging

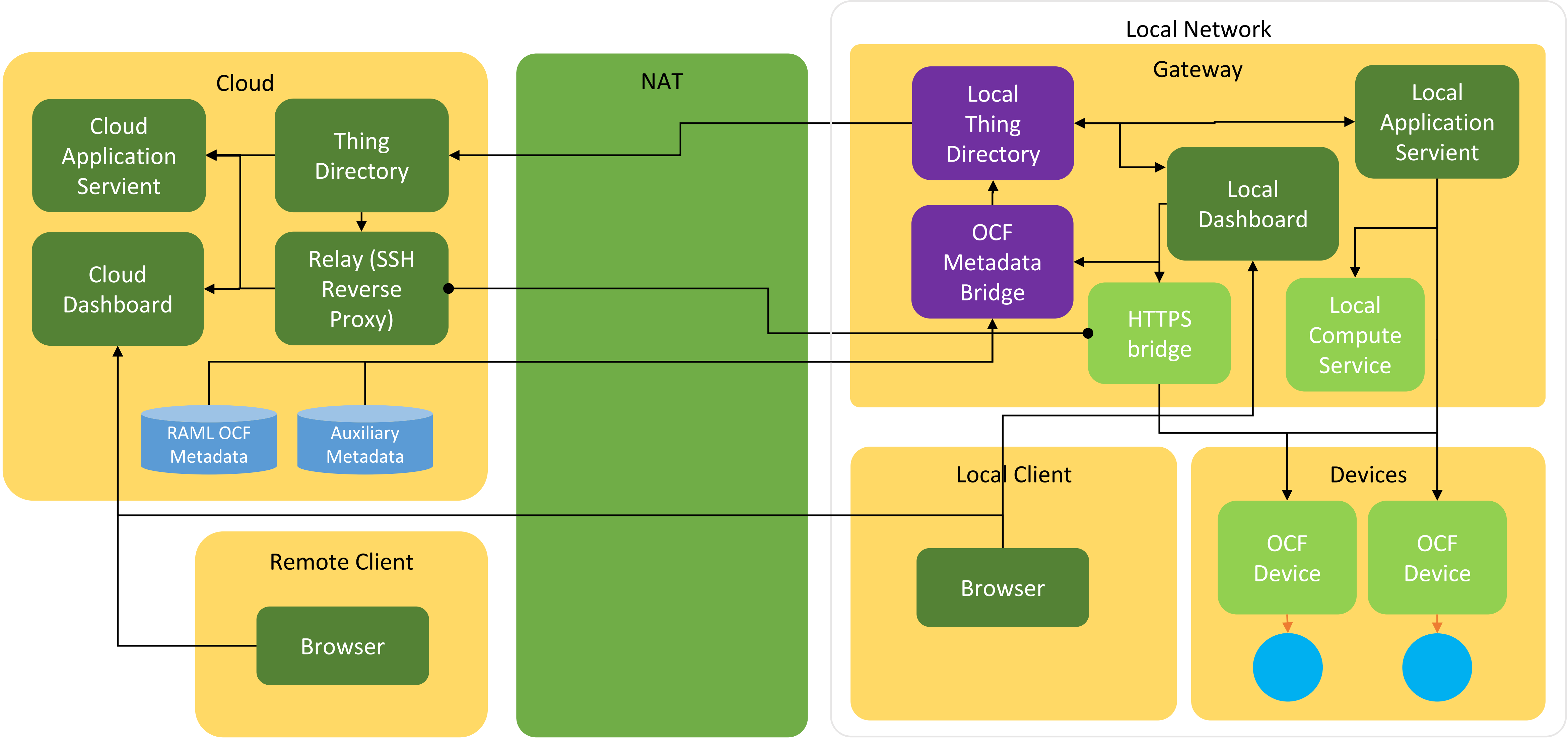
Goal: Maximize number of devices accessible

- Translate all metadata into common format (WoT TD)
- Infer capabilities and annotate with iotschema.org vocabulary
- Register all things with Thing Directory
- Use semantic search to find devices with specific capabilities regardless of device standard
- Support NAT traversal, security, etc.



1. Metadata Bridging

Add: Ingest official OCF data models, traverse NAT, use Thing Directory, …




ThinkPad
Pass

Metadata Bridge

Metadata generator includes:

- Semantic Annotation
    - WoT tdcontext, common context; iotschema.org,
    - HTTP methodName, rel
    - Capabilities (on Things), Interactions, Data
- Protocol Bindings
    - inputData, outputData

WoT Thing Description

…dynamically generated from OCF metadata

```
{   "@context":["http://w3c.github.io/wot/w3c-wot-tdcontext.jsonld",
                "http://w3c.github.io/wot/w3c-wot-common-context.jsonld",
                {"iot":"http://iotschema.org/"}, // Prefix definitions for semantic terms
                {"http":"http://www.w3.org/2011/http/"}],
    "base":"http://example.ocfgateway.net/api/oic",
    "@type":["Thing","Light","iot:LightControl",  // Capabilities
             "iot:Actuator","iot:BinarySwitch"],
    "name":"Intel-OCF-Smart Home LED (2relay)",
    "interaction":[
        {"name":"Switch Status",
         "@type":["Property","OnOffState","iot:SwitchStatus"],  // Interactions
         "link":[
             {"href":"/a/led2relay?di=79683ab5-8df1-4b7a-b110-c1b8fe251e7d",
              "mediatype":"application/json",
              "http:methodName":"http:post",  // Methods (specific to protocol binding)
              "rel":"setProperty"},
             {"href":"/a/led2relay?di=79683ab5-8df1-4b7a-b110-c1b8fe251e7d",
              "mediatype":"application/json",
              "http:methodName":"http:get",
              "rel":"getProperty"}
          ],
          → Payload structure and semantics
        },
        → Other interactions: Properties, Events, and Actions
    ]
}
```

Metadata Bridge

Metadata generator includes:

- Semantic Annotation
  - WoT tdcontext, common context; iotschema.org,
  - HTTP methodName, rel
  - Capabilities (on Things), Interactions, Data
- Protocol Bindings
  - inputData, outputData

// Payload structure and semantics for a Property interaction

```
"inputData":{
    "type":"object",
    "fields":[
        {
            "name":"value",
            "value":{
                "@type":["iot:Toggle"],  //
Data
                "type":"boolean"
            }
        }
    ]
},
"outputData":{
    "type":"object",
    "fields":[
        {
            "name":"value",
            "value":{
                "@type":["iot:Toggle"],
                "type":"boolean"
            }
        }
    ]
}
```

Semantic Annotation: iotschema.org Issues

- Missing Terminology:
  - TemperatureSensor? IlluminanceSensor?
- Inconsistent Terminology:
  - SwitchStatus/Toggle; Light Colour/Current Color; Temperature/TemperatureData; Light/TemperatureSensing

→ Most of these are due to lack of maturity in IoT ontology development

→ *Not* blocking issues…

→ Semantic tooling can bridge multiple ontologies

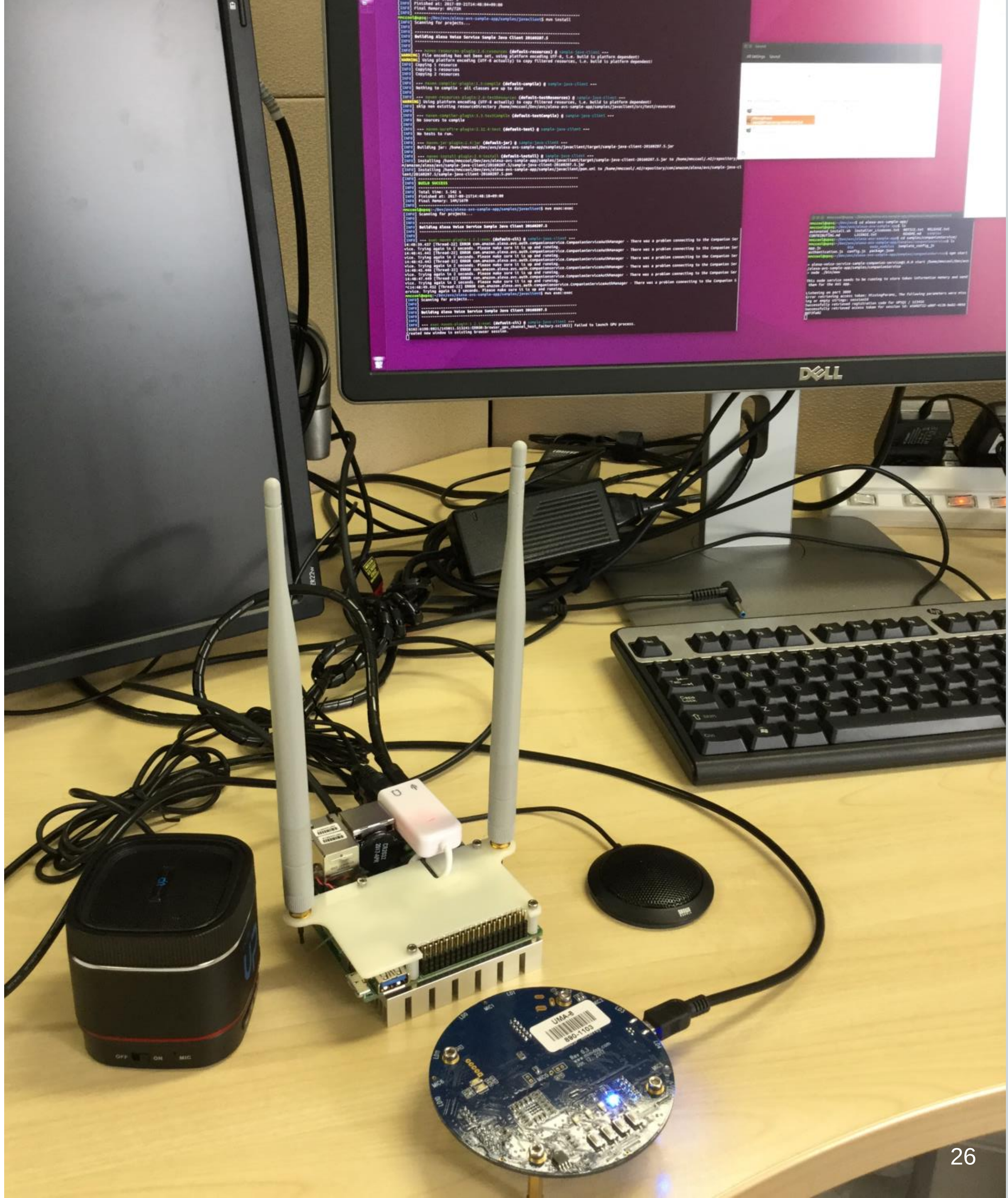
→ New and more precise ontologies can be used as they become available

→ … and as we learn how to write good ones and what the requirements are

2. Semantic Voice Control

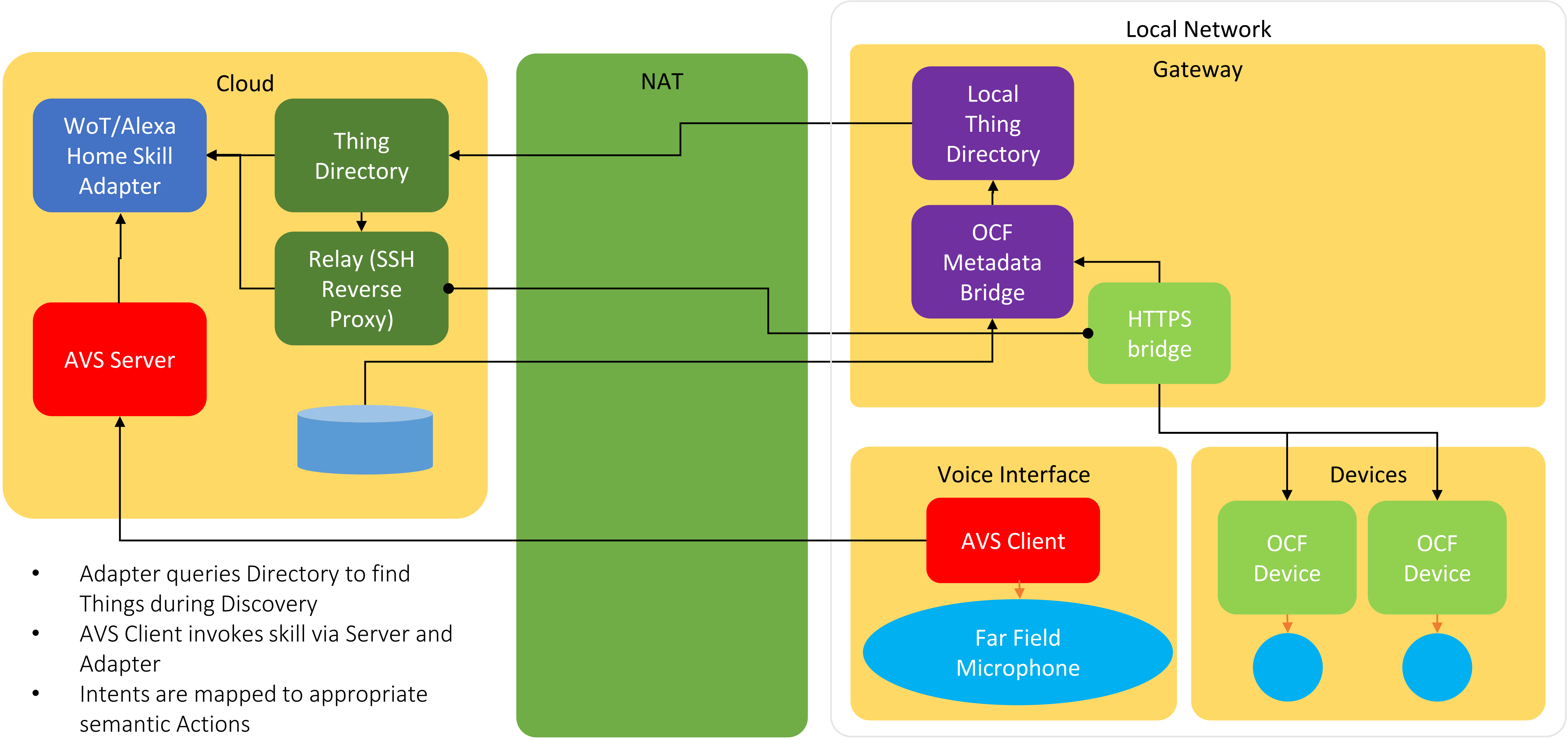
Goal: Enable *automatic* voice control of *any* WoT-enabled device

- Demonstrate use of semantic markup of Thing Description
  - Using iot.schema.org, SSNO ontologies, and semantic inferencing tools
- Generate adaptive AVS Alexa skill, bridging with Alexa Home Skill
- Layer with WoT metadata bridges to control devices from multiple ecosystems (including OCF)



2. Semantic Voice Control

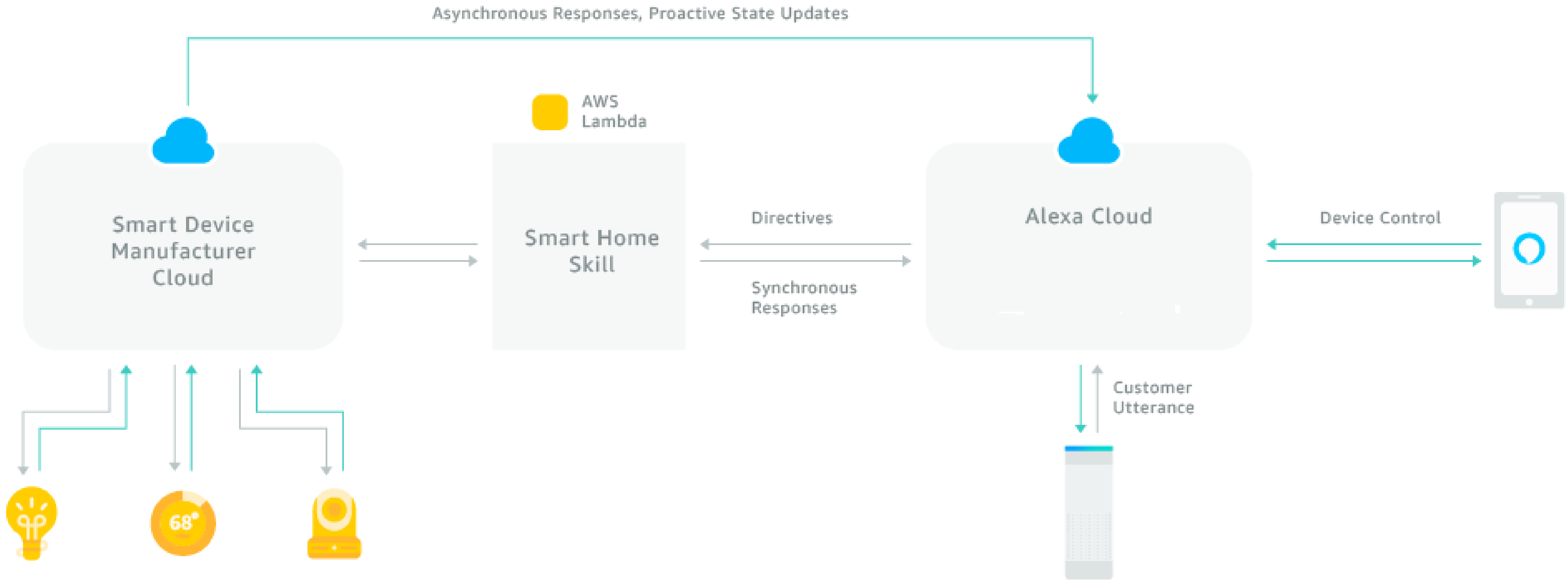
Goal: Provide generic voice interface using WoT Thing Descriptions



- Adapter queries Directory to find Things during Discovery
- AVS Client invokes skill via Server and Adapter
- Intents are mapped to appropriate semantic Actions

Alexa (AVS) Smart Home Skill:

Service Architecture



AVS Home Skill Semantics: Capabilities

Device functionality is modeled by specifying a set of standard capabilities.

- Each capability has a specific interaction model and payload
  …really more like an “interface” than just a semantic category
- Devices can support more than one capability
- “Device clouds” can have more than one device…

There are also a set of standard capabilities to get/set properties

Many capabilities support both absolute and relative mechanisms to adjust properties

Many “actions” also report the current state

AVS Home Skill Capabilities

Capability	Description
PowerController	TurnOn or TurnOff a device (has associated powerState)
PowerLevelController	SetPowerLevel (absolute) or AdjustPowerLevel (relative) on a device. Has an associated powerLevel property that can be set to a valuel between 0 and 100
PercentageController	Generic interface similar to PowerLevelController or BrightnessLevelController, but used when “power” or “brightness” is not a valid description of the controlled property. Assumes a value between 0 and 100 in an “percentage” property. Can also be used to read sensors.
TemperatureSensor	No controls, just use ReportState/StateReport to query “temperature” property.
BrightnessController	SetBrightness (absolute) or AdjustBrightness (relative) on a light. Has an associated brightness property that can be set to a value between 0 and 100
ColorTemperatureController	SetColorTemperature (absolute) or DecreaseColorTemperature, IncreaseColorTemperature (relative, no value) on a device. Has an associated colorTemperature property that can be set to a temperature in Kelvin (voice interface understands color names, “warm”, “daylight”, etc).
ColorController	SetColor of a light using “color” whose type is a hue, saturation, brightness (HSB) triple. Note that mapping from this to RGB involves a matrix transform and a nonlinearity. Voice interface uses color names.

AVS Home Skill Capabilities: Additional Examples

Capability	Description
LockController	Lock or Unlock a lock device
ThermostatController	Supports thermostats with one, two, or three setpoints. Can also be queried to find the current temperature using a property and ReportState/StateReport. There is also a “mode” property: ECO, AUTO, COOL, HEAT, etc.
InputController	Select AV input to a TV using set of standard names (HDMI, etc)
ChannelController	Select Channel on a TV (both absolute and relative)
PlaybackController	Start and stop audio source
SceneController	Select a scene by name. Some limits on discovery based on device type.
Speaker	SetVolume, AdjustVolume, SetMute
StepSpeaker	AdjustVolume, SetMute (a subset of “Speaker” with no absolute volume, for when range of volume is not known).
CameraStreamController	Start and stop streaming video

AVS Home Skill Semantics: System Messages

Directives can also create messages corresponding to “system capabilities”.

Capability	Description
Discovery	Every device: reports initial identification and capabilities
Authorization	AcceptGrant, provides bearer token, currently only OAuth2.0 supported
Response	Respond to directive - success
ErrorResponse	Respond to directive – error/failure, gives type of error, message for logging. Errors can include system issues (eg connectivity) but also payload issues (value out of range).
ReportState	Request state report
StateReport	Reply to state report
DeferredResponse	Used to indicate that will respond asynchronously to request (typ. >8s needed)
EndPointHealth	Check if physical endpoint can be reached from cloud shadow. Has associated Boolean property, “connectivity”

Summary

- Interoperability has significant business value in IoT
- Semantically-annotated WoT TD supports interoperability
  - Supports semantic search for Things
  - Once Things are found, TD describes how to connect
- WIP: Map voice intents to semantic capabilities…
- *We need to work though some concrete scenarios (POCs)*
  - Ideally, include fog computing in system architecture

WoT: Related W3C Standards

- RDF: Resource Description Framework
  - General mechanisms for defining data semantics and vocabularies
  - Useful for working with metadata
- SSNO: Semantic Sensor Network Ontology
  - Vocabularies (ontologies) and semantics for sensor data
- JSON-LD: JSON (JavaScript Object Notation) Linked Data
  - Mechanism for encoding RDF in JSON, used for Thing Description serialization
- See also:
  - iot.schema.org: Vocabularies and semantics for IoT, defined using RDF

Web of Things: Resources and Links

- W3C: World Wide Web Consortium: https://www.w3.org
- Web of Things Interest Group: https://www.w3.org/WoT/IG/
  - Charter: Leverage web standards and technology to enable IoT interoperation
  - Web architecture: https://www.w3.org/standards/webarch/
- Web of Things Working Group in the W3C to develop standard recommendations:
  - https://www.w3.org/2016/09/wot-wg-charter.html
  - Co-chairs: Matthias Kovatsch (Siemens), Kazuo Kajimoto (Panasonic), Michael McCool (Intel)
  - White paper on WoT architecture: http://w3c.github.io/wot/charters/wot-white-paper-2016.html
- WoT current practices: http://w3c.github.io/wot/current-practices/wot-practices.html