THE ONEIOTA TOOL

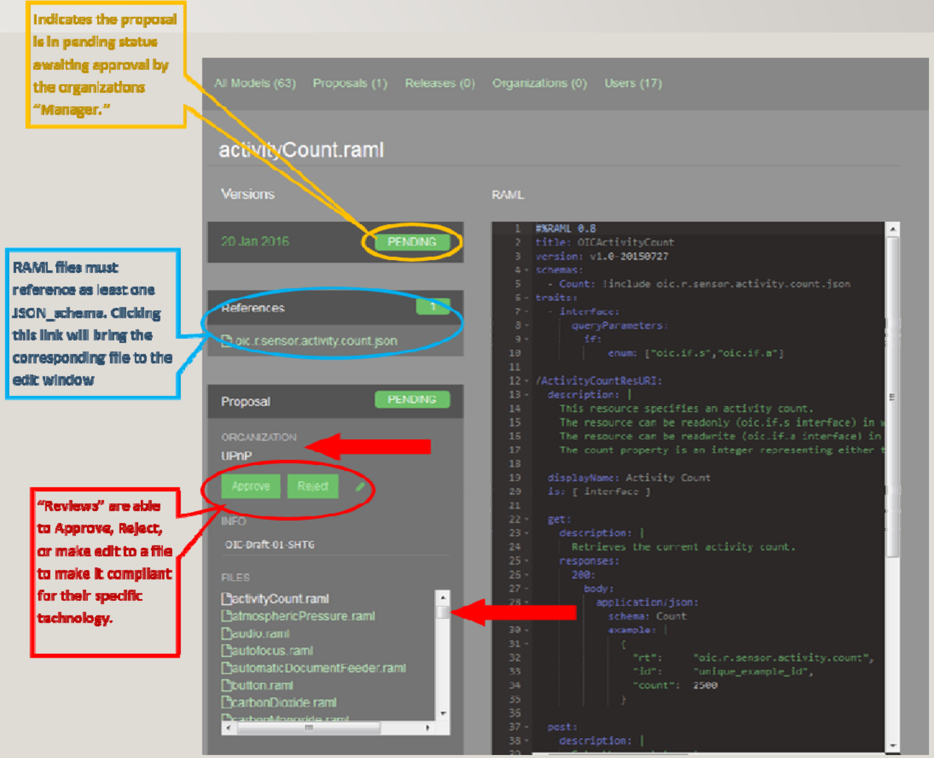
J. CLARKE STEVENS

HOW SHOULD THE IOT WORK?

- Creation of new devices should scale at Internet speed
  - New interfaces should take minutes to develop, not months
- Creation of prototypes should be simple
  - A working prototype is better than a thousand pictures
- All ecosystems and devices should work together
  - The device maker shouldn’t worry about being isolated by a technology choice

THE ONEIOTA TOOL
AN IDE FOR IOT

- A crowd-sourced Integrated Development Environment (IDE) for the Internet of Things device models (oneIoTa.org)
- RAML & JSON validated and syntax aware editors with shared editing
- Referenced files are clickable links
- Full versioning support



Indicates the proposal is in pending status awaiting approval by the organizations "Manager."

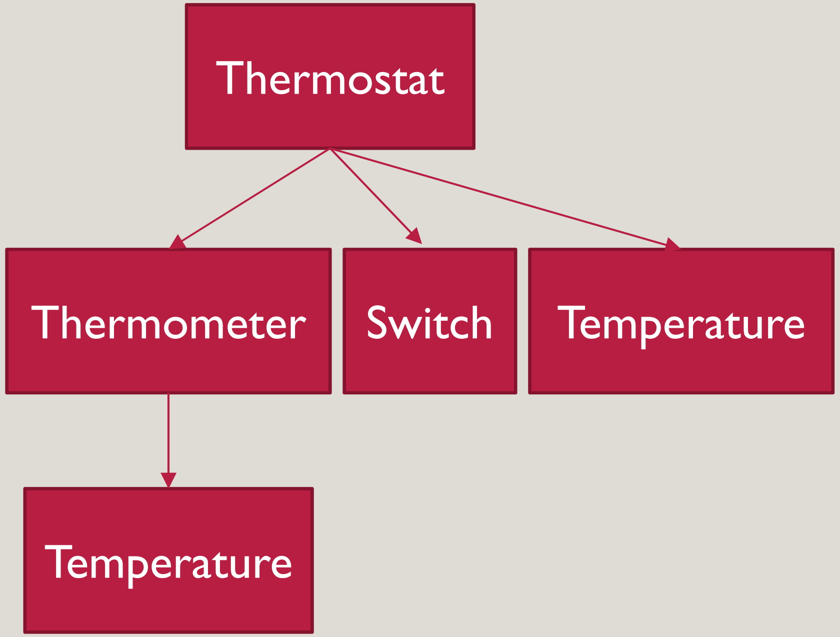
RAML files must reference as least one JSON_schema. Clicking this link will bring the corresponding file to the edit window

"Reviews" are able to Approve, Reject, or make edit to a file to make it compliant for their specific technology.

```
#%RAML 0.8
title: OICActivityCount
version: v1.0-20150727
schemas:
  - Count: !include oic.r.sensor.activity.count.json
traits:
  - interface:
      queryParameters:
        if:
            enum: ["oic.if.s","oic.if.a"]

/ActivityCountResURI:
  description: |
    This resource specifies an activity count.
    The resource can be readonly (oic.if.s interface) in w
    The resource can be readwrite (oic.if.a interface) in
    The count property is an integer representing either t

  displayName: Activity Count
  is: [ interface ]

  get:
    description: |
      Retrieves the current activity count.
    responses:
      200:
        body:
          application/json:
            schema: Count
            example: |
              {
                "rt":     "oic.r.sensor.activity.count",
                "id":     "unique_example_id",
                "count":  2500
              }

  post:
    description: |
```

THE CONSTRUCTIVE DEVICE DATA MODEL (SCALES AT INTERNET SPEED)

- Choose a generic description strategy (e.g. RAML, JSON schemas)
- Start with physical properties (e.g. temperature, mass)
- All new devices are defined as collections of physical properties and previously defined devices (e.g. a thermostat is a collection of temperature, thermometer and switch)
- Abstract devices can also be defined (e.g. Clarke's house, upstairs bedrooms)



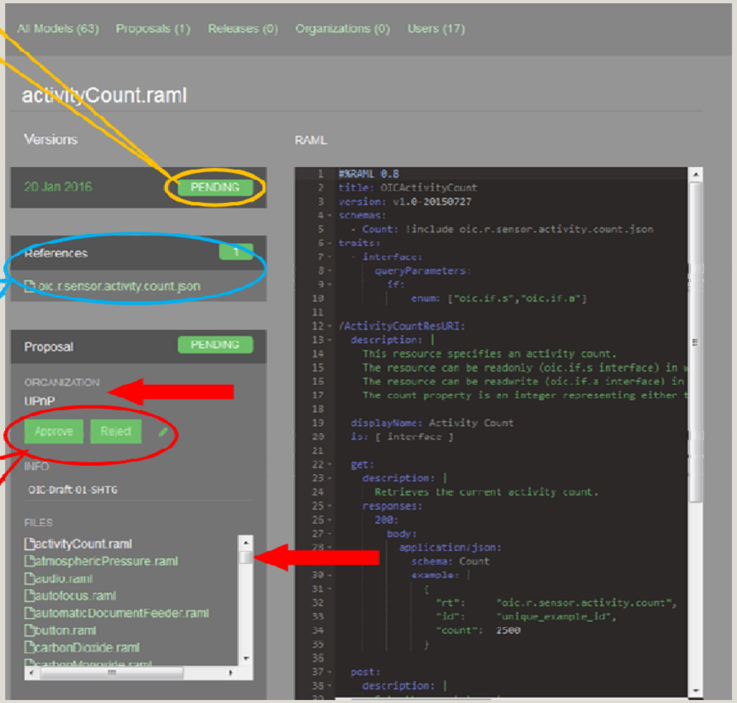
THE ONEIOTA TOOL
PROCESS MANAGEMENT FOR APPROVAL

- Step-by-step process for creation of new proposed resources
- Integrated reuse of existing resources
- Interactive approval process for multiple organizations
- Git repo on the back end, so it fits into your regular development process

Indicates the proposal is in pending status awaiting approval by the organizations "Manager."

RAML files must reference as least one JSON_schema. Clicking this link will bring the corresponding file to the edit window

"Reviews" are able to Approve, Reject, or make edit to a file to make it compliant for their specific technology.

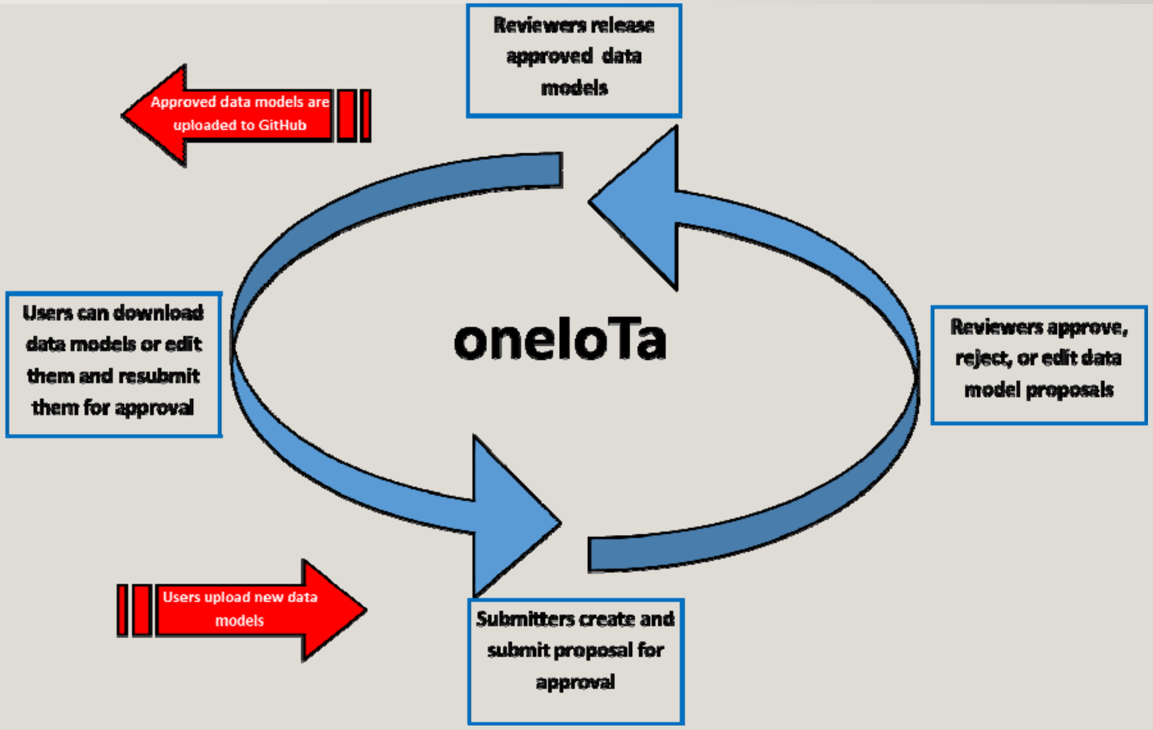


activityCount.raml
atmosphericPressure.raml
audio.raml
autofocus.raml
automaticDocumentFeeder.raml
button.raml
carbonDioxide.raml

```
#%RAML 0.8
title: OICActivityCount
version: v1.0-20150727
schemas:
  - Count: !include oic.r.sensor.activity.count.json
traits:
  - interface:
      queryParameters:
        if:
            enum: ["oic.if.s","oic.if.a"]

/ActivityCountResURI:
  description: |
    This resource specifies an activity count.
    The resource can be readonly (oic.if.s interface) in w
    The resource can be readwrite (oic.if.a interface) in
    The count property is an integer representing either t

  displayName: Activity Count
  is: [ interface ]

  get:
    description: |
      Retrieves the current activity count.
    responses:
      200:
        body:
          application/json:
            schema: Count
            example: |
              {
                "rt":     "oic.r.sensor.activity.count",
                "id":     "unique_example_id",
                "count":  2500
              }

  post:
    description: |
```

PROPOSAL TO INTEGRATED MODEL FOR MULTIPLE ORGANIZATIONS

- Create a proposal
  - Start with existing models or a blank template (RAML & JSON schema)
- Submit the proposal to an organization
- The organization decides if it will be accepted (iterative process)



Reviewers release approved data models

Approved data models are uploaded to GitHub

Users can download data models or edit them and resubmit them for approval

oneIoTa

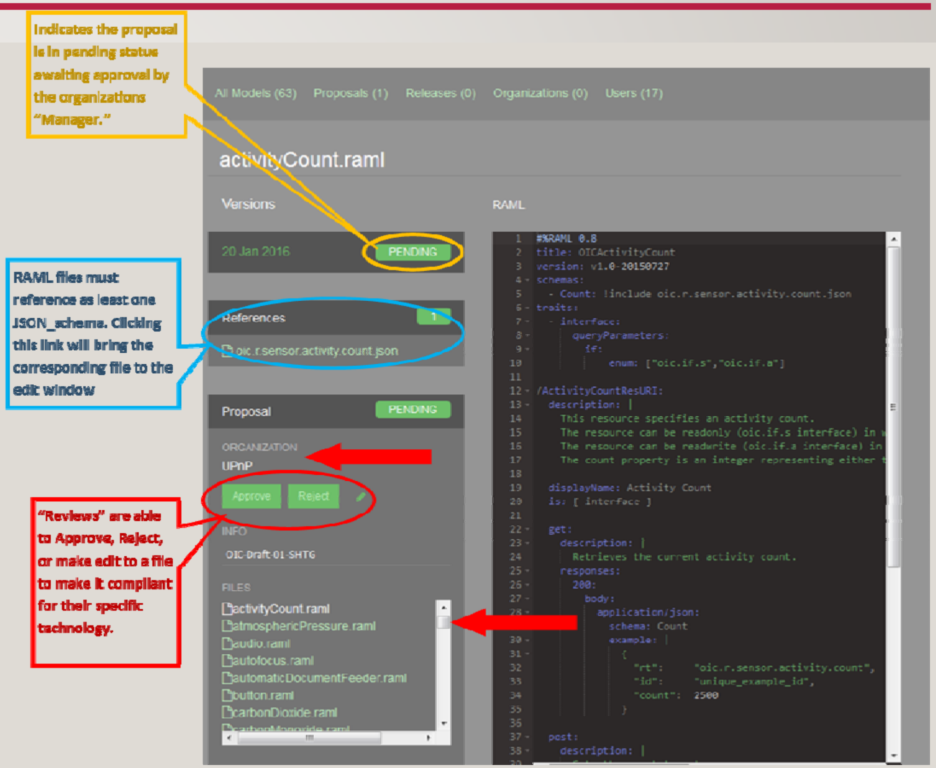
Reviewers approve, reject, or edit data model proposals

Users upload new data models

Submitters create and submit proposal for approval

THE ONEIOTA TOOL
INTEROPERABILITY IS BUILT-IN

- OCF as common data model facilitates interaction between different ecosystems
- Derived device data model syntax describes the relationship between data models of different ecosystems
- Everything referenced to the common data model is automatically interoperable with all other referenced models



Indicates the proposal is in pending status awaiting approval by the organizations "Manager."

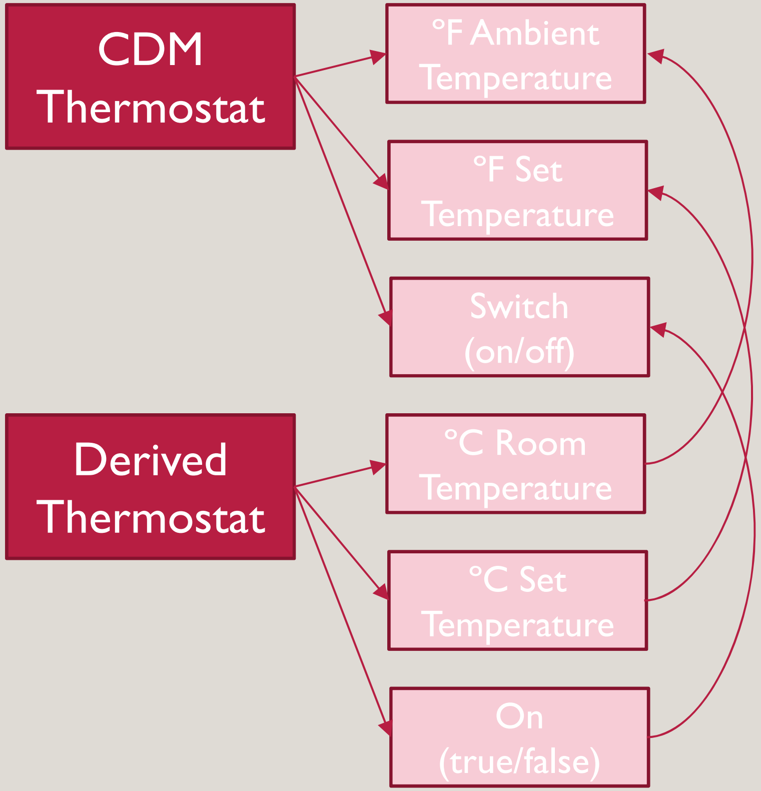
RAML files must reference as least one JSON_schema. Clicking this link will bring the corresponding file to the edit window

"Reviews" are able to Approve, Reject, or make edit to a file to make it compliant for their specific technology.

```
#%RAML 0.8
title: OICActivityCount
version: v1.0-20150727
schemas:
  - Count: !include oic.r.sensor.activity.count.json
traits:
  - interface:
      queryParameters:
        if:
            enum: ["oic.if.s","oic.if.a"]

/ActivityCountResURI:
  description: |
    This resource specifies an activity count.
    The resource can be readonly (oic.if.s interface) in w
    The resource can be readwrite (oic.if.a interface) in
    The count property is an integer representing either t

  displayName: Activity Count
  is: [ interface ]

  get:
    description: |
      Retrieves the current activity count.
    responses:
      200:
        body:
          application/json:
            schema: Count
            example: |
              {
                "rt":     "oic.r.sensor.activity.count",
                "id":     "unique_example_id",
                "count":  2500
              }

  post:
    description: |
```

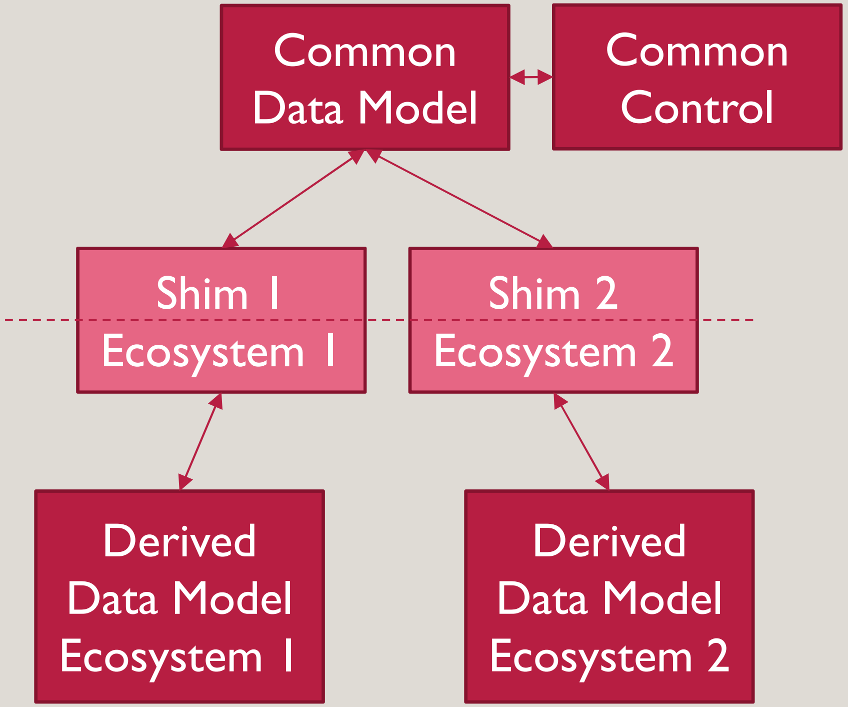
THE DERIVED DEVICE DATA MODEL (ALL ECOSYSTEMS WORK TOGETHER)

- ALL interoperable devices are defined exactly once in the common data model (CDM)
- Devices defined in other ecosystems (AllSeen, UPnP, etc.) are derived from devices in the common data model
- The definition of derived devices allows for differences in ecosystems (property names, variable types, range differences and conversions)



THE DERIVED DEVICE DATA MODEL (CONT.) (ALL ECOSYSTEMS WORK TOGETHER)

- In operation, a shim layer (code stubs automatically generated from the device data model) provides for conversion between ecosystems
- Since all ecosystems derive from the common data model, there are at most two conversions
- The conversion can happen in a gateway, in the cloud or in end devices



CALL TO ACTION

- Looking for developers who want to contribute new device data models
- Looking for advocates who want to develop code generators for specific target prototype platforms (e.g. Edison, Raspberry Pi, Arduino, etc.)
- Looking for advocates who want to develop code generators for specific target UI platforms (iOS, Android, Linux, etc.)
- Looking for developers who want to integrate data models for other platforms by defining derived models

Q&A

CONTACT INFORMATION

Clarke Stevens

jclarke.stevens@yahoo.com