Abstract

- A Redfish policy resource can be created to form control loops with the existing sensor and control resources. A policy management model has been proposed to manage these policy resources and allow Redfish clients to delegate autonomous policies to a Redfish service. This session will review the v0.9 work-in-progress model.

Policy model for Control Loops

John Leung

Disclaimer

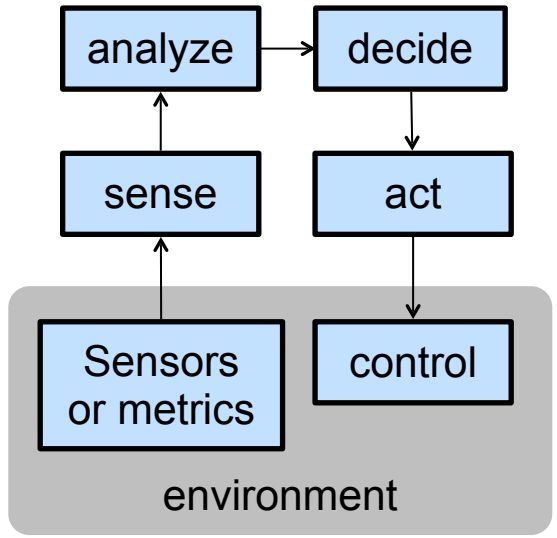
- The information in this presentation represents a snapshot of work in progress within the DMTF.
- This information is subject to change without notice. The standard specifications remain the normative reference for all information.
- For additional information, see the Distributed Management Task Force (DMTF) website.

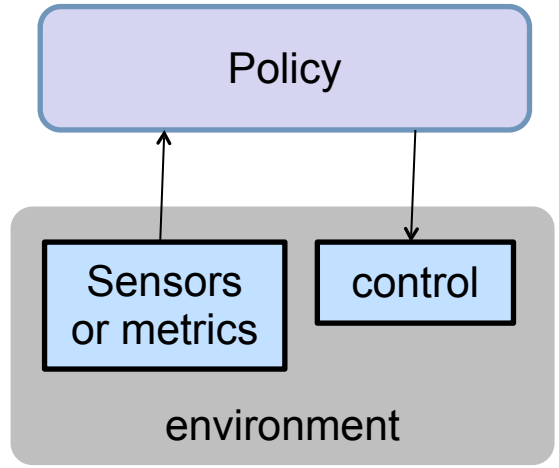
Contents

- Policy concepts: control loops and delegated policy
- Redfish policy model

Control Loops

- Redfish resources exist for metrics, sensors and controls
- A control loop can be constructed between sensors/ metrics and controls
  - Sense input(s)
  - Analyzed the input(s)
  - Decide on action(s), if any
  - Perform action(s) via control(s)
- The control loop can be structured as a policy resource
  - Analyze inputs and decide which reactions/controls to perform

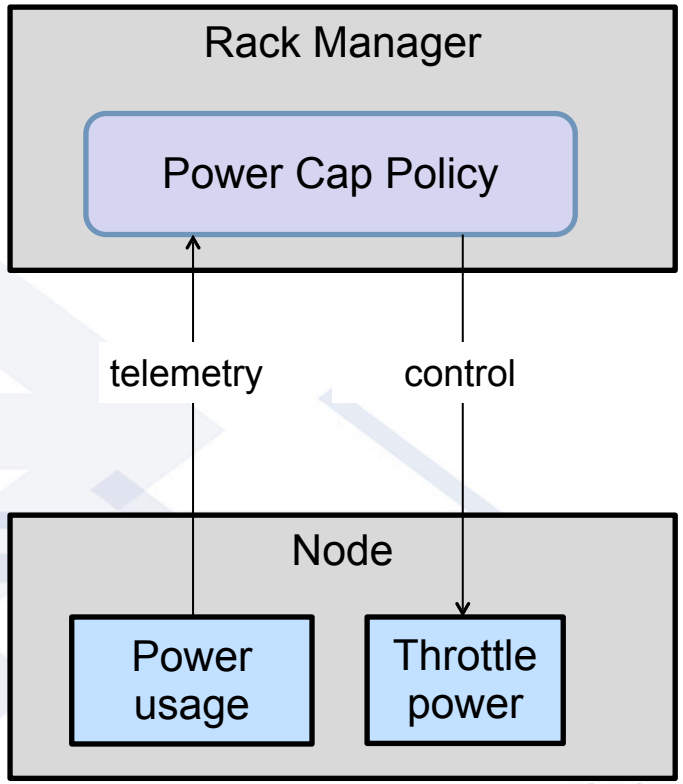


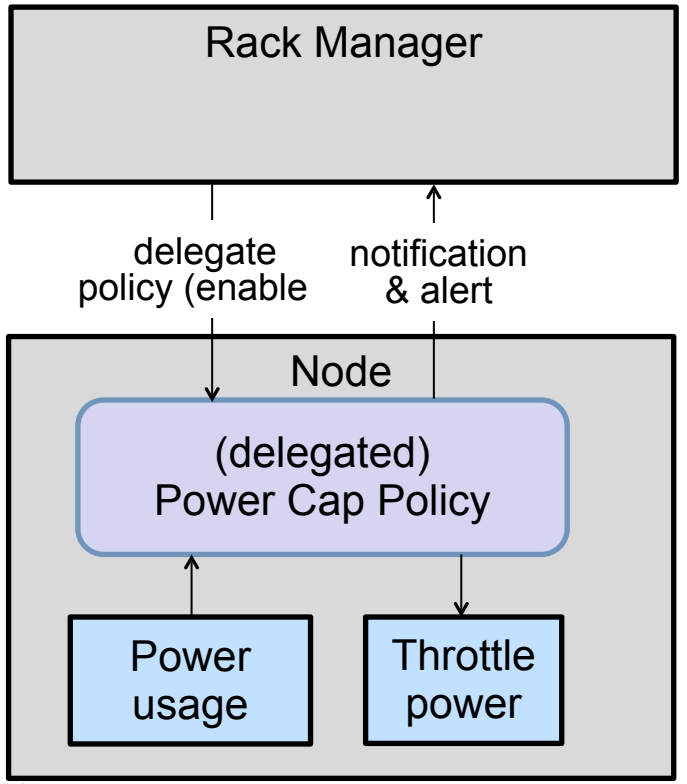


Policies can be Delegated

- The policy construct can be delegated down a hierarchy
- A delegated policy authorizes a node to enforce a policy, locally

Power cap example





Survey of Policy Management Models

Details in Redfish Policy Proposal (2021)[1]

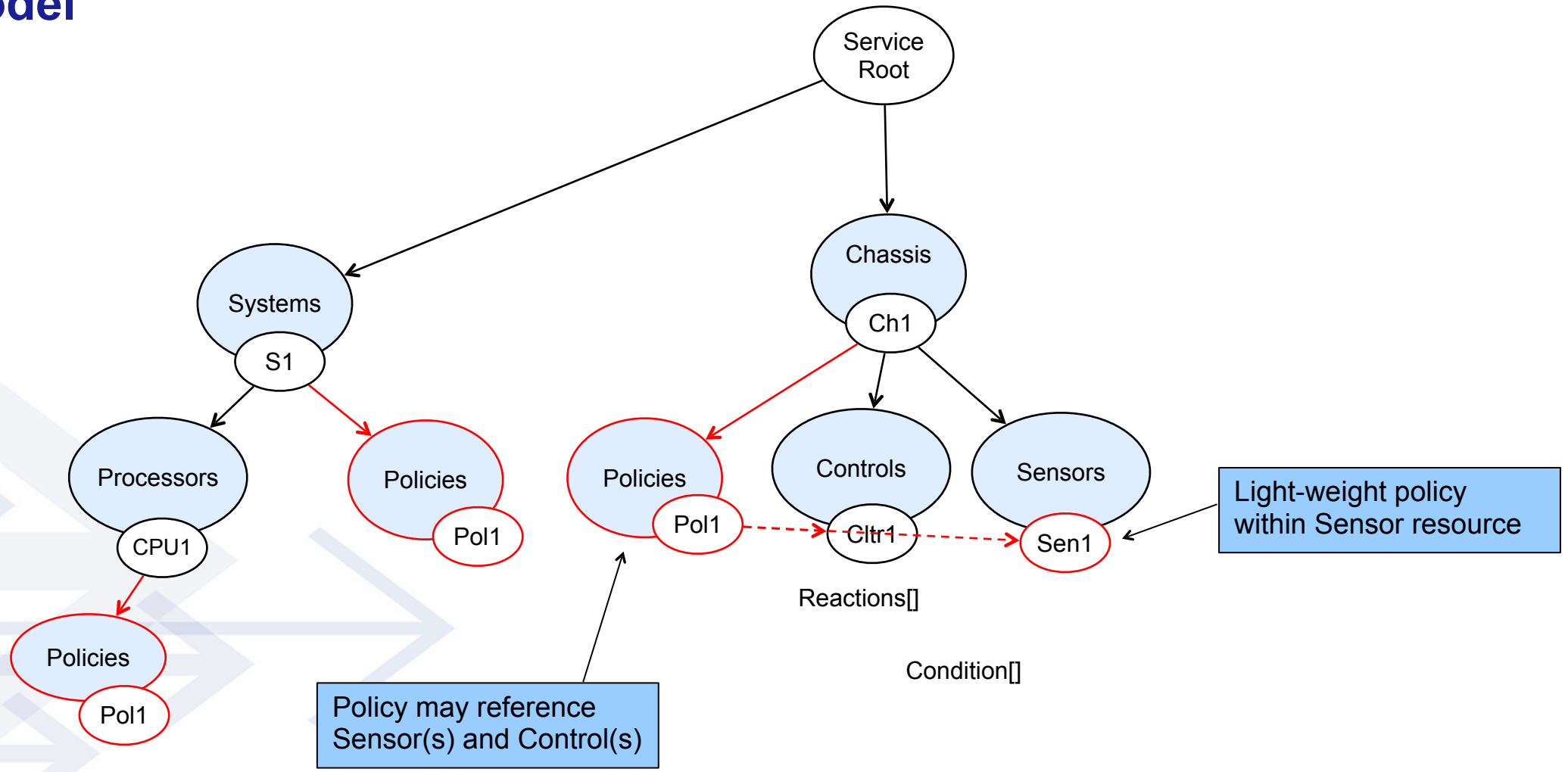
- DEN/COPS Policy Statements (1996-1998)
- DMTF/IETF Policy Framework (circa 2001) – introduced PDP and PEP
- TM Forum - GB922 R18.0.2 "Shared Information/Data Model" (2018)
- ETSI - Context-Aware Policy Management Gap Analysis (2018)
- ONF - The Policy Framework for ONOS (2019)
- DMTF DSP1048 - Network Policy Management Profile (2021)
- ETSI – GR NFV-IFA042 v4.1.1 "Policy Model" (2021)

[1]Redfish Policy Proposal - https://www.dmtf.org/sites/default/files/Policy_Model_Proposal_v10.pdf

Redfish Policy Model Requirements

- Policy
  - Can have multiple conditions and reactions
  - Can be subordinate to multiple resources (System, Chassis, Manager, etc)
  - Provides notification of policy exceptions or issues
- Conditions and Reactions
  - Can reference Sensor and Control resources, when applicable
  - Conditions can be local or remote (i.e. on another Redfish service)
  - Reactions can be delayed
- Simple policy
  - Ability to include light-weight policy (a single reaction) in the Sensor resource

Policy Model



Policy Resource

```
{
        "@odata.type": "#Policy.v1_0_0.Policy",
        "Name": "Policy for primary supply pressure",
        "Id": "PrimarySupplyPressurePolicy",
        "PolicyEnabled": true,
        "PolicyTriggered": false,
        "Status": {
                "State": "Enabled",
                "Health": "OK"
        },
        "PolicyConditions": [
                {
                        "PolicyConditionType": ". . ",
                        . . .
                }
        ],
        "PolicyReactions": [
                {
                        "Control": "/redfish/v1/Chassis/CDU/Controls/PowerUsage",
                        "SetPoint": 0
                },
                {
                        “CommonReaction": "SendEvent"
                }
        ],
        "@odata.id": "/redfish/v1/Chassis/CDU/Policies/PrimarySupplyPressurePolicy"
}
```

PolicyConditionType's - Sources for a Policy Condition

- Policy condition can be local or from a remote source (peer)
- Policy condition can be a sensor, a property or an event

PolicyConditionType	Properties
Sensor	Sensor (resource), Threshold (per Threshold object)
Threshold	Property (resource property), PropertyValue, PropertyStringValue, Activation
Event	MessageKey, OriginOfCondition
PeerSensor	PeerSource, Sensor, TriggerThreshold
PeerThreshold	PeerSource, Property, PropertyValue, Activation
PeerEvent	PeerSource, MessageKey, OriginOfCondition

Policy - Sensor

```
{
        "@odata.type": "#Policy.v1_0_0.Policy",
        "Name": "Policy for primary supply pressure",
        "Id": "PrimarySupplyPressurePolicy",
        "PolicyEnabled": true,
        "PolicyTriggered": false,
        "Status": {
                "State": "Enabled",
                "Health": "OK"
        },
        "PolicyConditions": [
                {
                        "PolicyConditionType": "Sensor",
                        "Sensor": "/redfish/v1/Chassis/CDU/Sensors/PrimarySupplyPressure",
                        "TriggerThreshold": "UpperCritical"
                }
        ],
        "PolicyReactions": [
                {
                        "Control": "/redfish/v1/Chassis/CDU/Controls/PowerUsage",
                        "SetPoint": 0
                },
                {
                        “CommonReaction": "SendEvent"
                }
        ],
        "@odata.id": "/redfish/v1/Chassis/CDU/Policies/PrimarySupplyPressurePolicy"
}
```

Policy - Peer Sensor

PeerSource1


```
{
        "@odata.id": "/redfish/v1/PeerService/PeerSources/PeerSource1",
        "@odata.type": "#PeerSource.v1_0_0.PeerSource",
        "Id": "PeerSource1",
        "Name": "PeerSource One",
        "HostName": "https://Someserver.Contoso.com/redfish/v1",
                "UserName": "root",
        "Password": null,
        "Links": {
                "ConnectionMethod": {
                        "@odata.id": "/redfish/v1/PeerService/
ConnectionMethods/ConnectionMethod1"
                },
                "ResourcesAccessed": [
                        {
                                "@odata.id": "/redfish/v1/Managers/1"
                        }
                ]
        }
}
```



```
{
        "@odata.type": "#Policy.v1_0_0.Policy",
        "Name": "Policy for primary supply pressure",
        "Id": "PeerPrimarySupplyPressurePolicy",
        "PolicyEnabled": true,
        "PolicyTriggered": false,
        "Status": {
                "State": "Enabled",
                "Health": "OK"
        },
        "PolicyConditions": [
                {
                        "PolicyConditionType": "PeerSensor",
                         "Peer": "/redfish/v1/PeerService/PeerSources/PeerSource1",
                        "Sensor": "/redfish/v1/Chassis/CDU/Sensors/PrimarySupplyPressure",
                        "TriggerThreshold": "UpperCritical"
                }
        ],
        "PolicyReactions": [
                {
                        "Control": "/redfish/v1/Chassis/CDU/Controls/PowerUsage",
                        "SetPoint": 0
                },
                {
                        “CommonReaction": "SendEvent"
                }
        ],
        "@odata.id": "/redfish/v1/Chassis/CDU/Policies/PeerPrimarySupplyPressurePolicy"
}
```

Policy - Threshold


```
{
        "@odata.id": "/redfish/v1/Chassis/CDU/Policies/PeerPrimarySupplyPressurePolicy",
        "@odata.type": "#Policy.v1_0_0.Policy",
                "Name": "Policy for primary supply pressure",
        "Id": "PrimarySupplyPressurePolicy",
                "PolicyEnabled": true,
                "PolicyTriggered": false,
                "Status": {
                "State": "Enabled",
                "Health": "OK"
        },
        "PolicyConditions": [
                {
                        "PolicyConditionType": "Threshold",
                        "Property": "/redfish/v1/Chassis/CDU/Sensors/PrimarySupplyPressure#/Reading",
                        "PropertyValue": 1400,
                        "Activation": "Increasing"
                }
        ],
        "PolicyReactions": [
                {
                        "Control": "/redfish/v1/Chassis/CDU/Controls/PowerUsage",
                        "SetPoint": 0
                },
                {
                        "Reaction": "SendEvent"
                }
        ]
}
```

Policy - Peer Event

```
{
    "@odata.id": "/redfish/v1/Chassis/CDU/Policies/PeerLeakDetectionPolicy",
    "@odata.type": "#Policy.v1_0_0.Policy",
        "Name": "Policy for leak detection",
    "Id": "LeakDetectionPolicy",
        "PolicyEnabled": true,
        "PolicyTriggered": false,
        "Status": {
        "State": "Enabled",
        "Health": "OK"
    },
    "PolicyConditions": [
        {
            "PolicyConditionType": "Event",
            "EventPeerSource": "PeerSource1",
            "MessageKey": "…",
            "OriginOfCondition": "…"
        }
    ],
    "PolicyReactions": [
        {
            "Control": "/redfish/v1/Chassis/CDU/Controls/PowerUsage",
            "SetPoint": 0
        },
        {
            “CommonReaction": "SendEvent"
        }
    ]
}
```

Simple Policy - within Sensor resource

```
{
  "@odata.type": "#Sensor.v1_9_0.Sensor",
  "Id": "PrimarySupplyPressure",
  "Name": "Primary Supply Pressure",
  "ReadingType": "PressurekPa",
  "Status": { "State": "Enabled", "Health": "OK" },
  "Reading": 827,
  "ReadingUnits": "kPa",
  . . .
  "Thresholds": {
     "UpperCritical": {
        "Reading": 1380,
        "Activation": "Increasing",
        "DwellTime": "PT1M",
        "PolicyReaction": {
            {
                "Control": "/redfish/v1/Chassis/CDU/Controls/PowerUsage",
                "SetPoint": 0
            }
        }
     }
  }
}
```

Call to Actions

- Review and profile feedback on the Redfish Policy Model WIP
  - DMTF Feedback portal

Backup

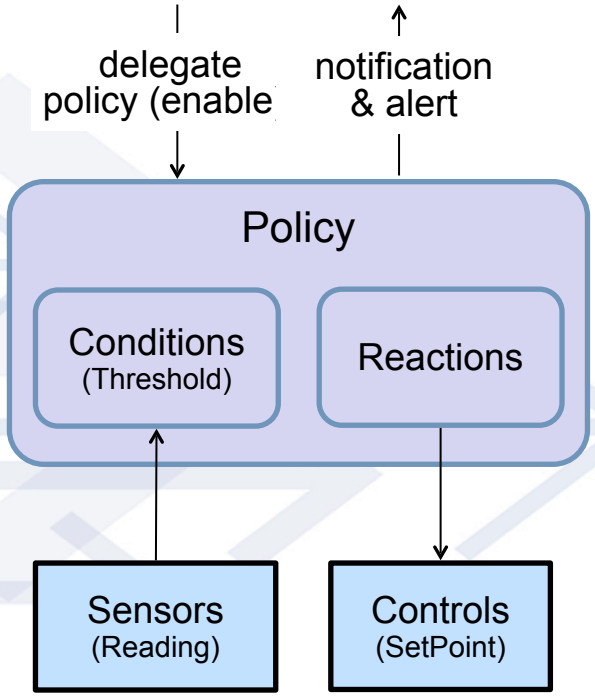
Example Sensor
Primary Supply Pressure Sensor


```
 {
  "@odata.type": "#Sensor.v1_9_0.Sensor",
  "Id": "PrimarySupplyPressure",
  "Name": "Primary Supply Pressure",
  "ReadingType": "PressurekPa",
  "Status": { "State": "Enabled", "Health": "OK" },
  "Reading": 827,
  "ReadingUnits": "kPa",
  "ReadingRangeMin": 0,
  "ReadingRangeMax": 1500,
  "ReadingBasis": "Zero",
  "SensingInterval": "PT5S",
  "PhysicalContext": "LiquidInlet",
  "Thresholds": {
     "UpperCritical": {
        "Reading": 1380,
        "Activation": "Increasing",
        "DwellTime": "PT1M"
     },
     "UpperCaution": {
        "Reading": 1240,
        "Activation": "Increasing",
        "DwellTime": "PT1M"
     }
  },
  "@odata.id": "/redfish/v1/Chassis/CDU/Sensors/PrimarySupplyPressure",
}
```

Redfish Policy requirements

Delegated Policy
- Policy schedule
- Conditions
  - dwell time
- Reactions
- Time to achieve
- Policy exception actions

Notification & alerts
- Condition triggered (event?)
- Policy triggered (event?, prop?)
- Policy exception (event?)
  - Unable to achieve
- Invoking reaction (event?)
- Loss of sensor(s)



- Delegated Power Limit Policy example
  - If power usage exceeds X, use delegated controls to reduce power usage below X within 50 ms (achieve goal within a timeframe)
  - If unable to achieve, notify the delegator
- Additional Policy Variations
  - Multiple conditions
    - Need to indicate AnyOf, AllOf, etc
  - Multiple reactions
    - Priority of policy reaction
    - Simultaneous vs stepwise reactions

Added since previous WIP

- Remote sensors and controls
  - Either the sensor or control or both can be on another Redfish service
- Simple sensor policy
  - A single policy reaction within a sensor threshold