Redfish for Cooling Units

WORK IN PROGESS

DMTF Redfish Forum

May 2021

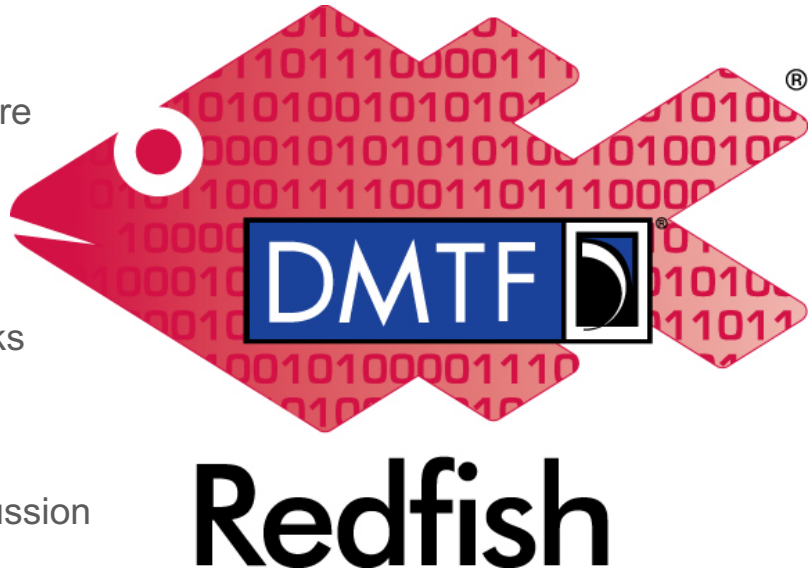
V0.5

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 DMTF website: www.dmtf.org

Getting involved in Redfish

- Redfish Standards page
  - Schemas, Specs, Mockups, White Papers & more
  - http://www.dmtf.org/standards/redfish
- Redfish Developer Portal
  - Redfish Interactive Resource Explorer
  - Educational material, documentation & other links
  - http://redfish.dmtf.org
- Redfish User Forum
  - User forum for questions, suggestions and discussion
  - http://www.redfishforum.com
- DMTF Feedback Portal
  - Provide feedback or submit proposals for Redfish standards
  - https://www.dmtf.org/standards/feedback
- DMTF Redfish Forum
  - Join the DMTF to get involved in future work
  - http://www.dmtf.org/standards/spmf



Introduction

- Proposal to extend Redfish DCIM models to incorporate cooling units
  - Desire support for rack-based Cooling Distribution Units (CDUs)
    - This is the primary use case for this proposal
  - Models should apply generally to other liquid cooling gear
    - Immersion cooling, heat exchangers, etc.
  - Expect the model to also cover air-cooling systems
    - Explicit coverage is not shown in this proposal, but some notes are mentioned
- Leverages existing Redfish DCIM models and style
  - Adapts the Power Distribution Unit concepts, schemas and properties
- Incorporates other work-in-progress topics within DCIM TF
  - Controls – several instances of valves for liquid flow
  - Sensors – expect need for discrete sensor types or another solution
  - Leverages concepts from ElectricalBus proposal

COOLING LOOP MODEL

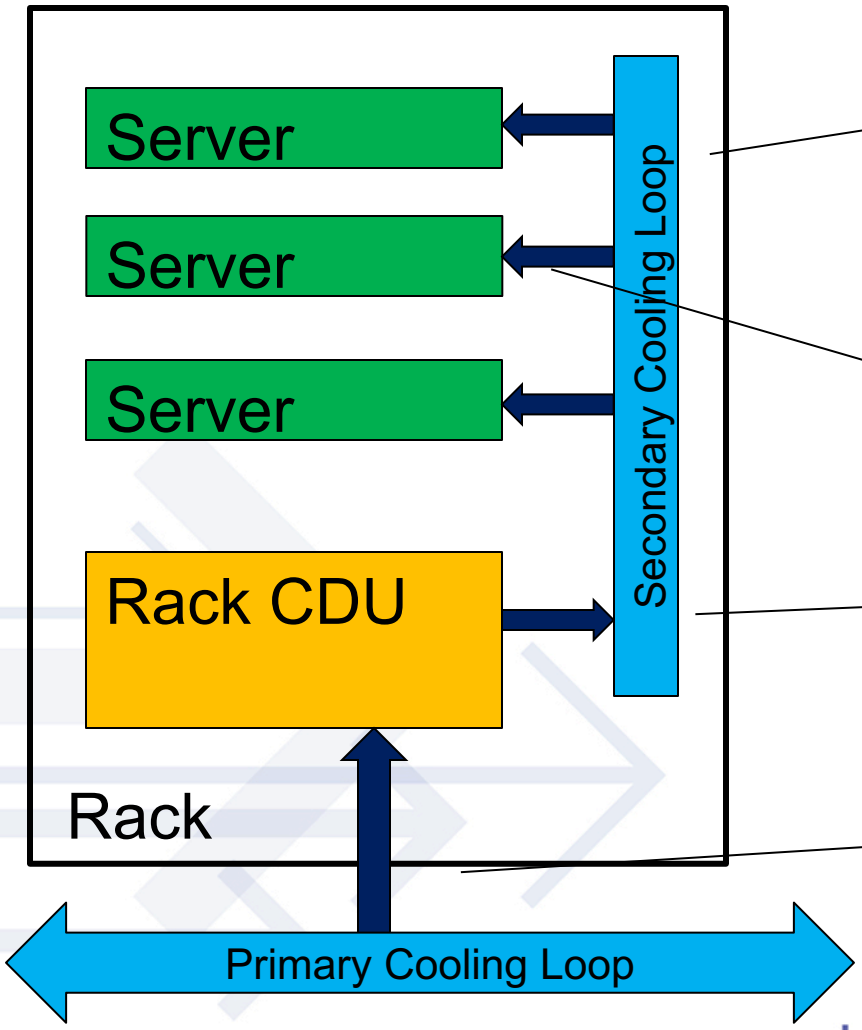
Cooling Loop Model

- Large-scale cooling equipment connects through “loops”
  - Loop can be self-contained (within a rack or group of racks)
  - Or can be facility-wide (primary loops from external chillers, etc.)
- Attempted to leverage Circuit concept, but found two differences:
  - Loops can have many-to-many relationships (Circuit is one-to-many)
    - Multiple “producers” and “consumers” attach to the cooling loop
  - External or facility-level loops not likely to have a Redfish interface
    - May simply be pipes and sensors, monitored by a specialized system
- Create a **LoopInterface** that connects to **CoolingLoop**
  - On both the “supply” and “return” side of the Redfish-managed equipment
  - Metrics are gathered at these interface points
    - Can be different for each piece of equipment connected to the loop
  - Provide information about the connected loop if available
    - User-entered “loop name” provides a connection path through the infrastructure

NEW CoolingLoop schema

- Follows pattern laid out by **ElectricalBus** and **Circuit**
- Resource Collection appears in two places in the model
- Facility-level **CoolingLoopCollection**
  - For facility-level or other types that interconnect among racks of gear
  - Facility manager / aggregator would be the “owner” of the loop
- Cooling Unit-level **CoolingLoopCollection**
  - For Rack CDUs or other intra-rack, self-contained instances
  - An instance of **CoolingUnit** owns the loop
  - Provides links or allows user to list “consuming” equipment
  - Enables self-contained model without requiring **Facility** resource tree
- Three methods to represent connections to related equipment
  - *LoopName* – User-defined string for unmanaged loops
  - *CoolingLoops[]* – R/W array of links to external **CoolingLoop** instances
  - *LoopManagerUri* – User-defined link to a management console (any type)

Cooling Loop – Rack-level self-contained example



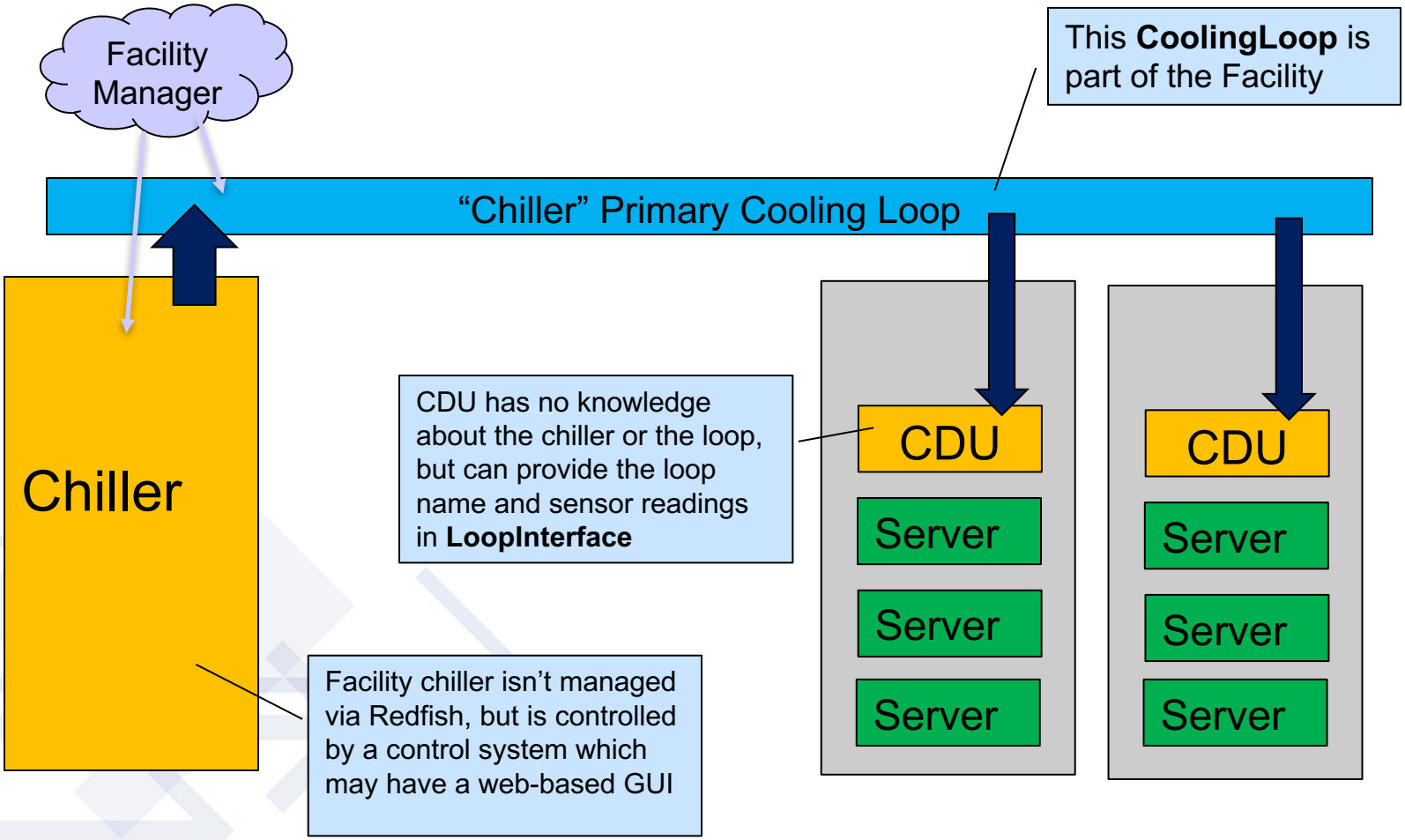
CoolingLoop is within the rack, and can be modeled and populated by the RackCDU's Redfish service

System's **LoopInterface** links to the external **CoolingLoop** (in the RackCDU)

RackCDU *secondary* **LoopInterface** connects to a **CoolingLoop** instance

RackCDU *primary* **LoopInterface** links to a **CoolingLoop** instance in the **Facility**, or provides just the *LoopName*

Cooling Loop – facility level example



NEW CoolingLoop resource

```
{
    "@odata.type": "#CoolingLoop.v1_0_0.CoolingLoop",
    "Id": "Building Chiller",
    "LoopType": "Facility",
    "SupplyingEquipmentNames": [ "Chiller" ],
    “ConsumingEquipmentNames": [ "Rack #1 CDU", "Rack #2 CDU", "Rack #3 CDU", "Rack #4 CDU"],
    “LiquidQuality”: <some classification or spec-level >
    "TemperatureCelsius": {
        "DataSourceUri": "/redfish/v1/Facilities/Room237/Sensors/ChillerTemp",
        "Reading": 13.7
    },
    "PressurePa": {
        "DataSourceUri": "/redfish/v1/Facilities/Room237/Sensors/ChillerPressure",
        "Reading": 3447.4
    },
    "FlowValve": {
        "DataSourceUri": "/redfish/v1/Facilities/Room237/Controls/ChillerFlow",
        "Reading": 1.58,
        "ReadingUnits": “L/s",
        "SetPoint": 80,
        "SetPointUnits": "%"
    },
    "Links": {
        "ConsumingEquipment": [
            {
                "@odata.id": “192.42.23.1/redfish/v1/CoolingEquipment/CDUs/1"
            }
        ]
    },
```

EquipmentNames allow users to manually add non-Redfish devices to help complete the model

Sensor excerpts and *Control* excerpt for valves

Need to explain “percent open” vs “percent closed” – Boolean property or normative language

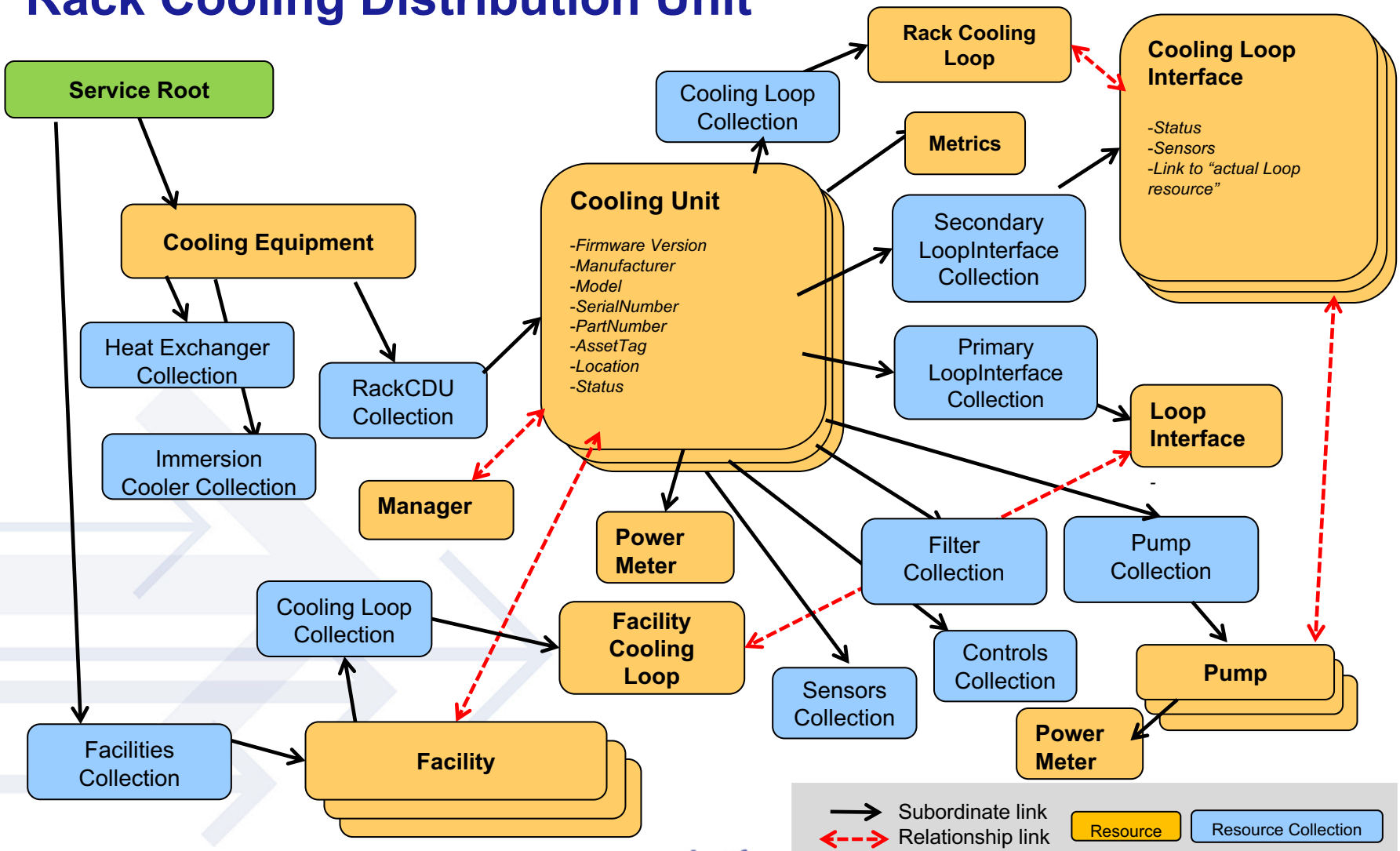
Links to Redfish-managed *Consuming* and *Source* resources – likely references to external Redfish services

COOLING EQUIPMENT MODEL

NEW CoolingEquipment resource

- Single resource instance under **ServiceRoot**
  - Follows design pattern used for **PowerEquipment**
  - Contains links to all cooling systems or equipment
  - Used primarily for discovery of managed equipment
- Links to Resource Collections of:
  - Rack-level Cooling Distribution Units (CDU's)
  - Heat exchangers
  - Immersion cooling units
  - Computer Room Air Handler (CRAH) units
  - Computer Room Air Conditioners (CRAC) units
  - Other cooling equipment?

Rack Cooling Distribution Unit



NEW CoolingUnit schema and resources

- Unified schema defines several types of cooling gear
  - Share common modeling and property definitions
  - *EquipmentType* property provides specific identification
  - Separate collections of each type linked from CoolingEquipment resource
- Resource contents
  - General product identification – model, manufacturer, serial number, etc.
  - Versioning – Hardware revision, firmware version, date of manufacture
- Links to subordinate Resources and Resource Collections
  - Sensor Collection, Metrics (entire unit)
  - Primary (input) Loop Interface(s)
  - Secondary (output) Loop Interface(s)
  - Subsystems: Pumps, Filter, Reservoirs
  - Cooling Loops – if owned and managed by the CoolingUnit

NEW CoolingUnit schema


```
{
    "@odata.type": "#CoolingUnit.v1_0_0.CoolingUnit",
    "Id": "1",
    "EquipmentType": "RackCDU",
    "Name": "Rack #4 Cooling Distribution Unit",
    "FirmwareVersion": "3.2.0",
    "Version": "1.03b",
    "ProductionDate": "2020-12-24T08:00:00Z",
    "Manufacturer": "Contoso",
    "Model": "BRRR4000",
    "SerialNumber": "29347ZT536",
    "PartNumber": "ICE-9",
    "UUID": "32354641-4135-4332-4a35-313735303734",
    "AssetTag": "PDX5-92381",
    "Status": {
        "State": "Enabled",
        "Health": "OK"
    },
    "PrimaryLoopInterfaces": { “@odata.id”: < Link to LoopInterfaceCollection > },
    "SecondaryLoopInterfaces": { “@odata.id”: < Link to LoopInterfaceCollection > },
    "CoolingLoops": { “@odata.id”: < Link to CoolingLoopCollection > }.
    "Pumps": { “@odata.id”: < Link to PumpCollection > },
    "Filters": { “@odata.id”: < Link to FilterCollection > },
    "EnvironmentMetrics": { “@odata.id”: < Link to EnvironmentMetrics > },
    “PowerMeter”: { “@odata.id”: < Link to Circuit > },
    "Sensors": { “@odata.id”: < Link to SensorCollection > },
    “Controls”: { “@odata.id”: < Link to ControlCollection > },
    < TRUNCATED >
```

NEW Cooling Loop Interface resource

- The connection between the cooling unit and a CoolingLoop instance
  - Analogous to **Circuit** in relation to proposed **ElectricalBus** resource
  - Provides numerous sensor readings and controls
    - Flow, Temperature, Pressure on both supply and return
    - Valve controls, drain, etc.
  - If known, provide link to **CoolingLoop**
    - Or the loop name and Manager URI if known and populated by end user
- Primary monitoring resource for the cooling unit's functionality
  - Primary loop interfaces – input from facility chillers or other sources
  - Secondary loop interfaces – output from the distribution unit to feed "consuming" equipment

NEW LoopInterface schema

```
{
    "@odata.type": "#LoopInterface.v1_0_0.LoopInterface",
    "Id": "Chiller",
    "Name": “Primary Input from Chiller",
    "Status": {
        "Health": "OK"
    },
    "LoopInterfaceType": “Primary",
    “LiquidType”: “GlycolMixture”,
    “GlycolPercent”: 20,
    "RatedFlowLSeconds": 30,
    "SupplyValveControl": {
        "DataSourceUri": "/redfish/v1/CoolingEquipment/RackCDUs/1/Controls/ChillerSupplyValve",
        "SetPoint": 70,
        "SetPointUnits": "%",
        "Reading": 9.5,
        "ReadingUnits": “L/s"
    },
    "SupplyTemperatureCelsius": {
        "DataSourceUri": "/redfish/v1/CoolingEquipment/RackCDUs/1/Sensors/LoopASupplyTemp",
        "Reading": 14.8
    },
   "PressurePa": {
        "DataSourceUri": "/redfish/v1/CoolingEquipment/RackCDUs/1/Sensors/LoopASupplyPressure",
        "Reading": 319.6
    },
    "ReturnTemperatureCelsius":  < SENSOR EXCERPT >
    "ReturnPressurePsi":         < SENSOR EXCERPT >
    < TRUNCATED >
```

As *LiquidType* is really a description of the loop – this may become a more general *LoopType* that describes any loop, and could be populated if CoolingLoop is managed.

Sensor excerpts and *Control* excerpt for valves

Cooling Loops and Interfaces – open questions

- Sensor coverage:
  - Temperature, Pressure (abs and diff), Flow, Humidity, Dew point
  - Leak detection – will need sensor presence / working
    - Will add a number of *PhysicalContext* values (Floor, sub-floor, etc.)
  - Water quality – Need SME feedback here –several possible sensor types
    - Optical clarity, pH / alkalinity, salt %, hardness, dissolved solids ppm, etc.
- Support for Air / Phase Change / Liquid loops
  - Phase change (refrigerant) loops can be modeled with this pattern as well
    - Not recommended for initial release, wait for industry feedback
  - LoopType = Air, Liquid, Primary / Secondary?
    - An “air loop” could be the room, plenum, or ductwork
    - Air handling may not be ‘contained’, but the “loop interface” does still apply
  - FluidType = Water, GlycolMixture, Dielectric (immersion), others?
    - Would like an enumeration of fluid types (refrigerants) if possible

NEW PowerMeter resource

- **EnvironmentMetrics** has ability to show single *Power* sensor reading
- But cooling units are complex power systems
  - 3-phase power is typical for rack-level components
  - Will require monitoring of power, current, voltage, frequency, energy
  - This is likely true for individual pump units as well
  - May have breakers, perhaps power controls
- Leverage existing **Circuit** schema for this purpose
  - Add new *CircuitType* of "PowerMeter"
  - *EnergykWh exc*erpt property allows true "power meter" use cases
    - Can be reset using the *ResetMetrics* action
    - Also allows for a *LifetimeReading*
  - Allow Circuit instance under **CoolingUnit** or **Pump**
  - Expect further re-use in other industrial-scale equipment models

Circuit schema as PowerMeter example

```
{
   "@odata.type": "#Circuit.v1_3_0.Circuit",
   "Id": “PowerMeter",
   “Name": “Pump #1 Power Meter",
   "Status": { < Status object> },
   "CircuitType": “PowerMeter",
   "PhaseWiringType": "TwoPhase3Wire",
   "NominalVoltage": "AC240V",
   "RatedCurrentAmps": 16,
   "BreakerState": "Normal",
   “PowerState”: “On”,
   “VoltageSensor”: { < Single-phase voltage sensor > },
   "PolyPhaseVoltageSensors": { < Voltage per phase sensors > },
   "CurrentSensor": { < Total Current sensor > },
   "PolyPhaseCurrentSensors": { < Current per phase sensors > },
   "PowerSensor": { < Total Power sensor > },
   "PolyPhasePowerSensors": { < Power per phase sensors > },
   "FrequencySensor": { < Frequency sensor > },
   "EnergySensor": { < Energy sensor > },
   “Actions": { < ResetBreaker, ResetStatistics > }
   "@odata.id": "/redfish/v1/CoolingEquipment/RackCDUs/1/Pumps/1/PowerMonitor",
}
```

NEW Subsystem schemas for Cooling Unit

- **Pump** Resource Collection
  - Will have differential pressure / absolute pressure, flow, etc.
    - Can be a physical sensor or a synthesized value (model as sensor)
  - Variable Frequency Drive may need an object
  - *PowerMeter* (Circuit) subordinate resource
    - May be 3-phase, have a breaker, etc.
- **Filter** Resource Collection
  - Pressure sensors
  - Service time / install time, life etc.
  - ASHRAE requirements / classifications
  - Flush / clean actions?

NEW Subsystem schemas for Cooling Unit, continued

- **HeatExchanger** Resource Collection
  - For air-liquid systems
  - Likely a unit with a **Fan** collection with redundancy information
  - Temperature, power, other sensors
  - Controls and policies
- **Reservoir** Resource Collection
  - Fill level, pressure sensors
  - Air bleed value (controls), fill valve, drain valve
  - May have connections between reservoirs (balancing)

Additions to existing schema

- **Sensor**
  - Additional Sensor types
    - Leak detectors and other state-based sensors
  - Describing opened/closed valve positions
    - Represent as percent, want something to describe “Normal open” / “Normal closed” positions to allow for consistent displays
  - Support for calibration, redundancy, grouping are already in process
- **EnvironmentMetrics**
  - Add DewPointCelsius temperature-based sensor
    - Calculated value from temperature and humidity
    - Represent as sensor for threshold purposes

DISCRETE SENSORS

Discrete Sensor investigation

- Redfish avoided definition of discrete sensors
  - Want boolean values to map to a property with descriptive values instead
    - Ex: DoorState= Open/Closed/Locked, instead of DoorSensor = True/False
- But there will be metadata for many of these types of sensors
  - Can be a sensor device with status, part number, location data, etc.
  - Leak detectors certainly fall into this category
  - Presence detection is also important
    - Ex: System will not allow startup unless sensors are present & functional
- Not a replacement for the *Status* properties (*Health*, *State*) or similar
- Suggest adding support to Sensor as a *ReadingType* of “Discrete”
  - Add *State* / *ReadingState* property (suggestions?) for discrete values:
    - Active, Inactive, Open, Closed, Normal, Error/Alert, others?
  - *Reading* is not included (or use 0/1 values – does that help or hurt?)
  - Add *SupportedReadingStates* to show capabilities

QUESTIONS FOR INDUSTRY

Open topics and further work expected

- Expect additional type(s) for *CoolingUnitType*
  - Provide support for other form factors or product categories
- Expect support for air-liquid cooling units
- Immersion cooling support
  - Expect “tank”, “bag”, etc. container to model as a secondary CoolingLoop
  - LoopInterface applies in the same manner
  - But may be odd to call it a “Loop” (analogous to Circuit vs Outlet)
- Significant number of common messages to define for Events / Alarms
  - Expect to define new message registries
    - Should be able to harvest existing SNMP trap definitions as a starting point

Q&A & Discussion