Redfish support for Liquid Cooling Equipment

Jeff Autor (Hewlett Packard Labs, HPE)

November 2024

Redfish

www.dmtf.org

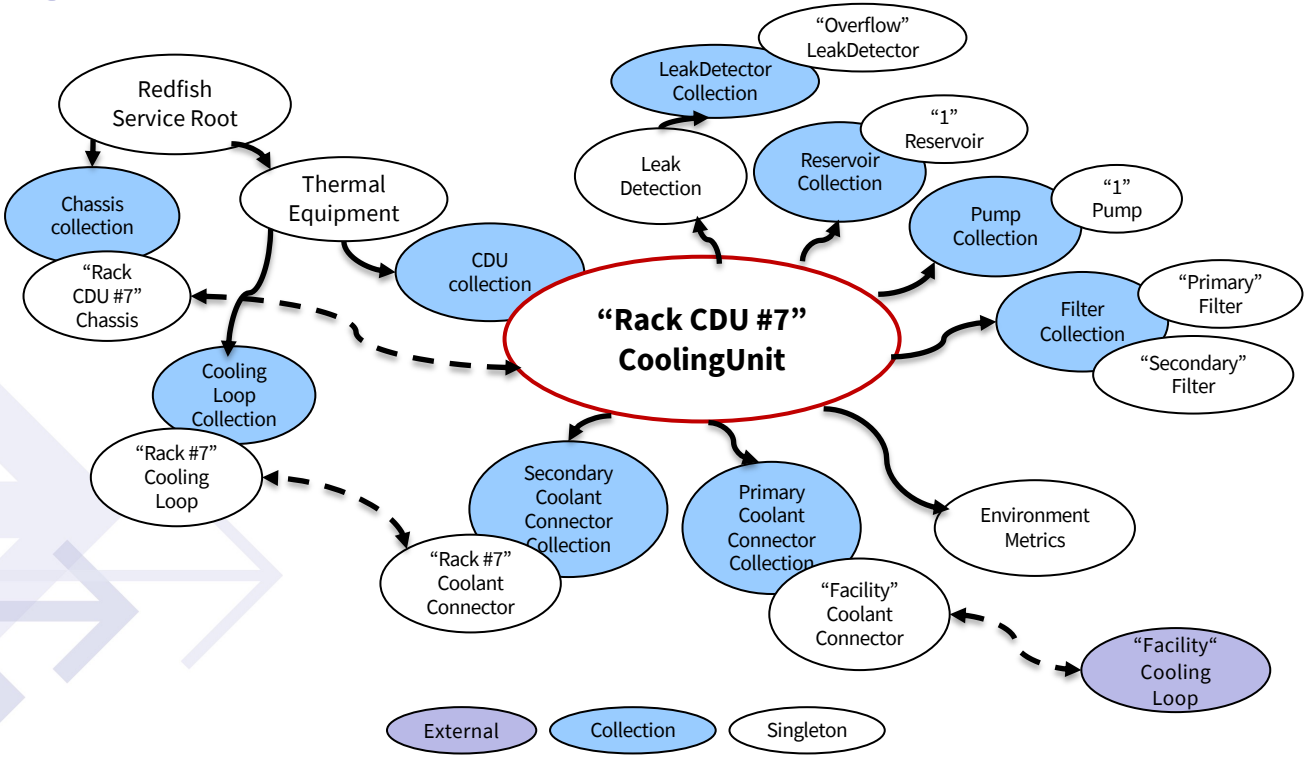
Disclaimer

- 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

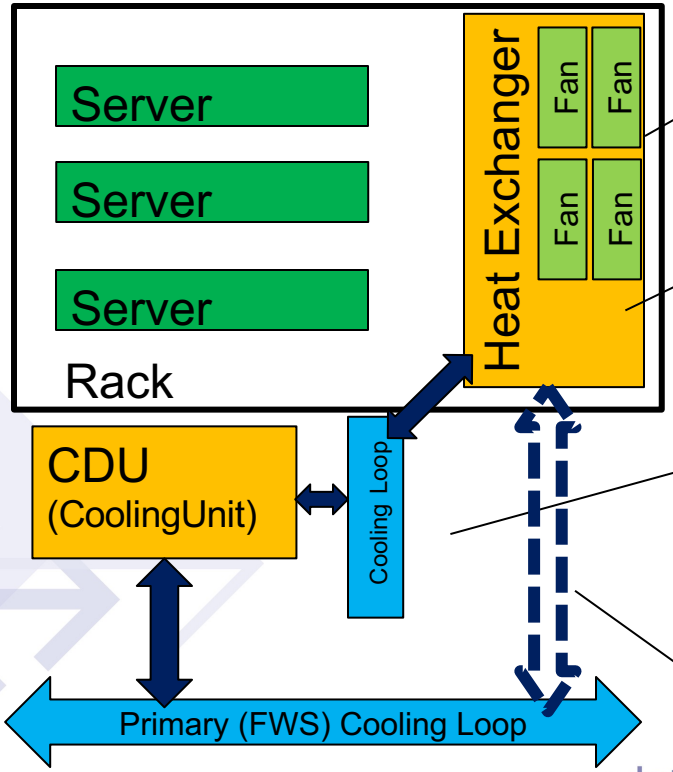
Redfish support for Liquid Cooling equipment

- Broad support added in 2023.1 release
  - Coolant Distribution Units (CDUs)
  - Rear-door heat exchangers
  - Immersion cooling systems
  - Server connections and self-contained / closed loop systems
- Details of subsystems and connections
  - Filters, Pumps, Reservoirs, Manifolds
  - Leak detection
- Expect first vendor implementations to be available by early 2025

Cooling Unit Model



Example: Rear Door Heat Exchanger



The heat exchanger model includes a **Chassis** with a **ThermalSubsystem** that describes the **Fans**

ThermalMetrics has new sensor excerpts to describe airflow and delta pressure

CDU *primary* **CoolantConnector** attaches to the facility cooling loop and its *secondary* **CoolantConnector** connects to a technology loop, which in turn connects to the heat exchanger via **CoolantConnector**

OR the heat exchanger directly connects to the facility loop via a **CoolantConnector**

Cooling controls – work in progress

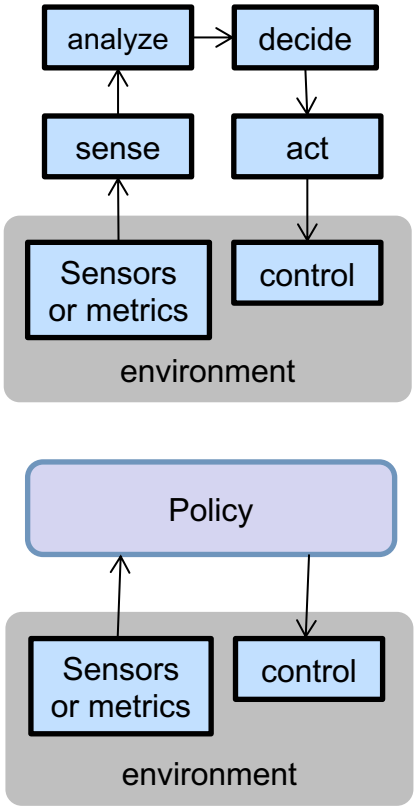
- Adding cooling controls to the data model
- Two common, independent set points for CDUs were identified
  - Secondary loop target flow rate
  - Secondary loop target supply temperature
- Add **Control** excerpts in **CoolantConnector** for secondary loop:
  - *TargetFlowLitersPerMinute*
  - *TargetSupplyTemperatureCelsius*
- Add action to **Pump** to enable/disable hot swappable pumps
  - *Pump.SetMode* - *"Enabled"* or *"Disabled"*, allow for future expansion
- Download proposal presentation "***Standard CDU Controls***"

Redfish Policy Model – work in progress

- Add support for exposing and adjusting policies for automated application
- **Policy** resource
  - Supports 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
- Enable “simple policy” support
  - Ability to include light-weight policy (a single reaction) in a **Sensor** resource

Applying Policy to Control Loops

- Using **Metric, Sensor** and **Control** resources
- 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**
  - Analyze inputs and decide which reactions or controls to perform
- Download proposal: **DSP-IS0028**



Telemetry improvements – work in progress

- Redfish primarily built for inventory, configuration, and ad hoc monitoring
  - Continuous polling Redfish resources for telemetry is not efficient
  - Existing **TelemetryService** and **MetricReport** needs improvement
- Desire to increase telemetry ecosystem adoption and interoperability
  - Redfish support must operate well with popular telemetry clients
    - e.g. Prometheus, Telegraf, OpenTelemetry, etc.
  - Need a simpler scheme to encourage support on small-footprint devices
- Proposal leverages existing schema-backed mechanisms to define "records" that can be streamed to clients or collected as a report
  - Download proposal: **DSP-IS0027**
- Work underway to better comprehend requirements of time-series database clients (*OpenTelemetry*)

Example: Sensor telemetry record

```
GET /redfish/v1/Chassis/1/Sensors/ServerTemp?telemetry=Compact


{
   “@Redfish.Id”: “J93KM8”,
   “@Redfish.Time”: 1696261238,
   "Reading": 21.3
}
```

@Redfish.Id annotation provides unique value to replace bulky @odata.id URI in records

@Redfish.Time annotation included in telemetry payload

For a **Sensor** resource, the *Reading* is the primary piece of data, which can change frequently.

Additional sensor data (average, peak, lowest values) would be available in the “detailed” telemetry record

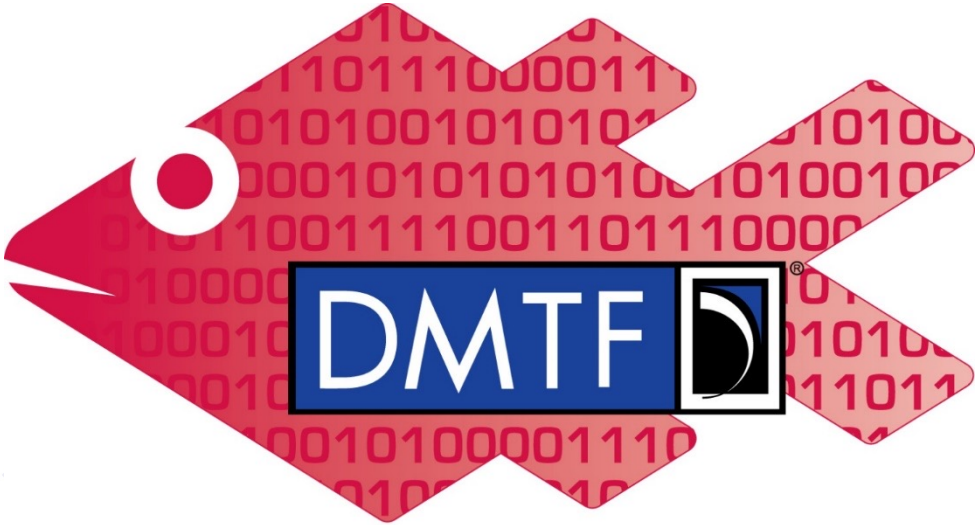
Other work in progress topics

- Standard event messages for liquid cooling systems
  - Additions to Redfish Message Registry (Environment registry)
  - Includes new messages for cooling units, subsystems, leak detection, and sensor threshold violations (pressure, flow rate, etc.)
  - Download work-in-progress Registry Guide: **DSP2065 2024.4WIP90**
- Enhancing support for “rope” style leak detectors
  - Provide details on “leak location” and detailed event message
  - Requirements for recalibration or resetting detectors
  - Group needs feedback from manufacturers and subject matter experts

Call to Action

- Download Work-in-progress materials
  - Under “Work in Progress” from http://www.dmtf.org/standards/redfish
  - Most of this material is expected to be finalized by early 2025
- Provide feedback to DMTF Redfish Forum
  - Any comments or suggestions on this material
  - https://www.dmtf.org/standards/feedback
  - Or post questions on public forum: https://www.redfishforum.com
- Explore and use Redfish support on all your IT and DCIM gear
  - Get started: https://github.com/DMTF/Redfish-Tacklebox/

Q&A & Discussion



Redfish