Document Identifier: DSP2090

Date: 2025-04-09

Version: 1.0.0

Redfish Aggregation Guidance for Compute Expansion Modules

Supersedes: None

Document Class: Informational

Document Status: Published

Document Language: en-US

CONTENTS

Foreword

The Redfish Aggregation Guidance for Compute Expansion Modules white paper was prepared by DMTF's Redfish Forum.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability. For information about DMTF, see http://www.dmtf.org.

Acknowledgments

DMTF acknowledges the following individuals for their contributions to this document:

- Michael Raineri — Dell Technologies

1 Introduction

Compute expansion modules are subsystems that provide additional compute resources to a host system. In many cases, these modules are available on the open market for system vendors to integrate in a solution. From the end user's perspective, while these modules might be serviced or replaced in the field independent of the overall solution, they operate the solution as a single functional unit.

One example of these modules are Open Compute Project-defined Open Accelerator Infrastructure Universal Baseboards, or simply UBBs. These modules also follow the OCP GPU & Accelerator Management Interfaces Specification to provide the system BMC with internal management capabilities.

Compute expansion modules often contain their own BMC or other type of management controller that provides a Redfish interface to the system BMC. To allow an end user to manage the devices on the compute expansion module, *implicit aggregation*, defined in the Aggregation clause of the Redfish Specification, is required. This paper provides guidance and best practices to properly aggregate the Redfish data model of the compute expansion module's BMC into the Redfish data model of the system BMC.

2 Common aggregation patterns

Throughout this document there are numerous cases where the system BMC aggregates entire resources from the compute expansion module BMC. This can either be in the form of copying the members of collections from the compute expansion module BMC and inserting them into a collection in the system BMC or taking singleton resources from the compute expansion module BMC and inserting it into the system BMC. There are several things that are expected to occur as a result of either of these operations:

- Subordinate resources from the aggregated resources in the compute expansion module BMC are also brought into the data model of the system BMC.
- Actions and OEM extensions from these resources are also included.
- The `Id` property in each of these resources may be adjusted to prevent key collisions in the system BMC's data model or due to naming policies.
  - One possible method to resolve collisions is to prepend the `Id` property value of the top-most `Chassis` resource from the compute expansion module BMC, or some other identifier that helps describe the module in relation to the rest of the system, with the resource causing the collision.

3 Chassis

The members of the `ChassisCollection`, found at `/redfish/v1/Chassis`, represent the physical containers monitored by the service. The compute expansion module BMC models its overall container as a `Chassis` resource and might contain other `Chassis` resources that represent smaller modules inside the container. The system BMC is expected to insert all `Chassis` resources discovered in the compute expansion module BMC into its own `ChassisCollection`.

While the collection is flat, the `Chassis` model allows for a service to express the hierarchy of containers with the `Contains` and `ContainedBy` properties inside `Links`. When the system BMC aggregates the members of the `ChassisCollection`, it is expected to create or update the `Contains` property in one of its `Chassis` resources to reference the top-most `Chassis` resource from the compute expansion module BMC. It will also create a `ContainedBy` property in the top-most `Chassis` resource from the compute expansion module BMC to reference one of its own `Chassis` resources. The specific `Chassis` resource linked from the system BMC is dependent upon the system design.

Figure 1 and Figure 2 show an example system with a UBB prior to aggregation where:

- A `Chassis` named `4U` is modeled by the system BMC to represent the overall enclosure.
- A `Chassis` named `UBB` is modeled by the compute expansion module BMC to represent the UBB.
- Two `Chassis` resources named `OAM1` and `OAM2` are modeled by the compute expansion module BMC to represent two OCP Accelerator Modules (OAMs) within the UBB.

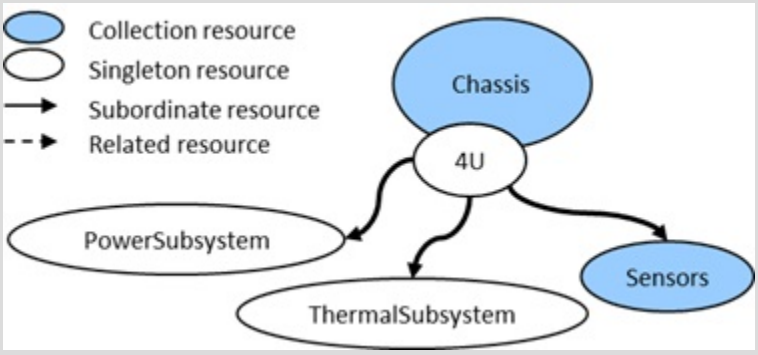

Figure 1 — System BMC chassis model

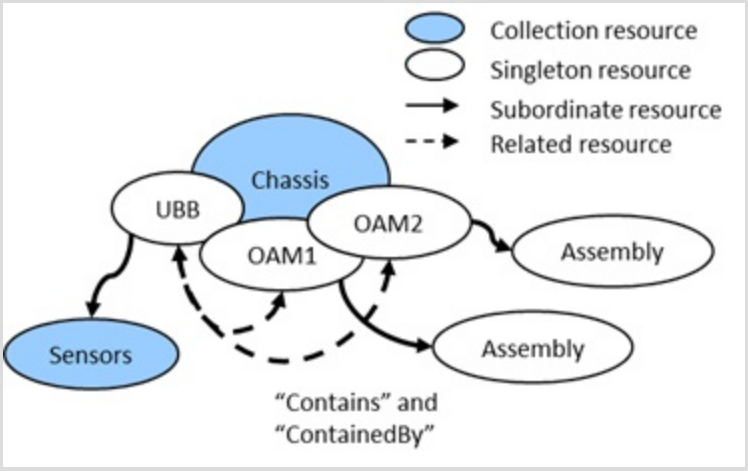

Figure 2 — Compute expansion module BMC chassis model

Figure 3 shows the aggregation result of the example system from the previous figures. There are now four members of the `ChassisCollection` and containment links were added to show the `4U` chassis contains the `UBB` chassis.

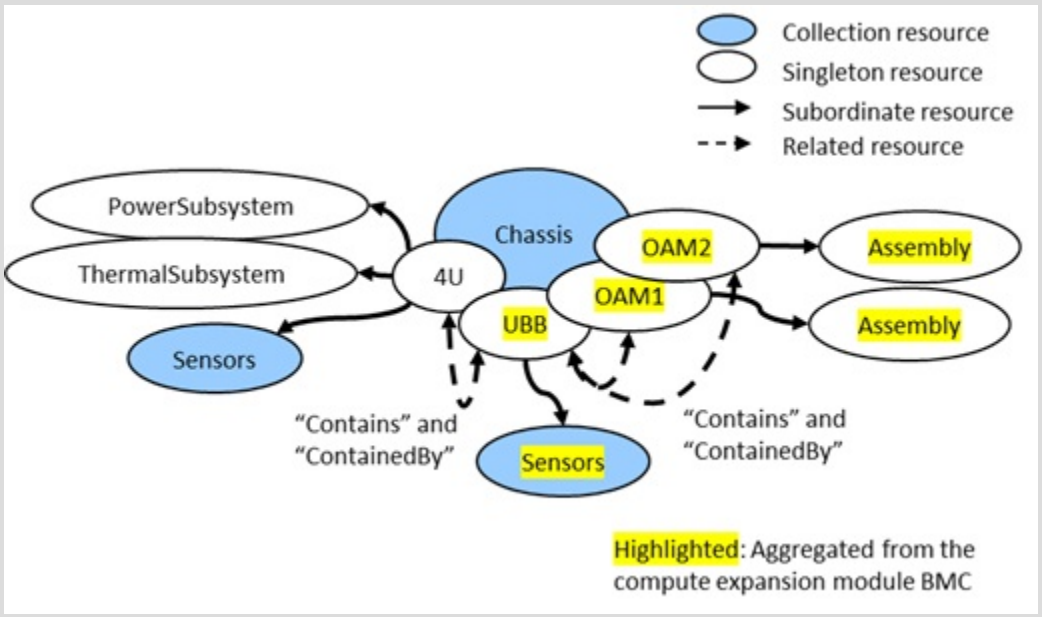

Figure 3 — Aggregated chassis model

4 Systems

The members of the `ComputerSystemCollection`, found at `/redfish/v1/Systems`, represent the functional view of the computers or systems available to the user. While it's not a complete, functional system, the compute expansion module BMC might use the `ComputerSystem` resource as a convient container to place resources for the system BMC to consume. The expectation here is the system BMC will merge the `ComputerSystem` resource with the additional compute resources into its own `ComputerSystem` resource. The `ComputerSystemCollection` would not grow as a result of this, but other resources subordinate to the system BMC's `ComputerSystem` resource would be extended with resources from the compute expansion module BMC. This would include aggregating subordinate resources and members of collections, such as the `ProcessorCollection` and `MemoryCollection`, for the system BMC's `ComputerSystem` resource.

Figure 4 and Figure 5 show an example system with a UBB prior to aggregation where:

- A `ComputerSystem` named `1` is modeled by the system BMC to represent the system and available system peripherals it directly monitors.
- A `ComputerSystem` named `UBB` is modeled by the compute expansion module BMC to represent the GPUs and GPU memory on the module.

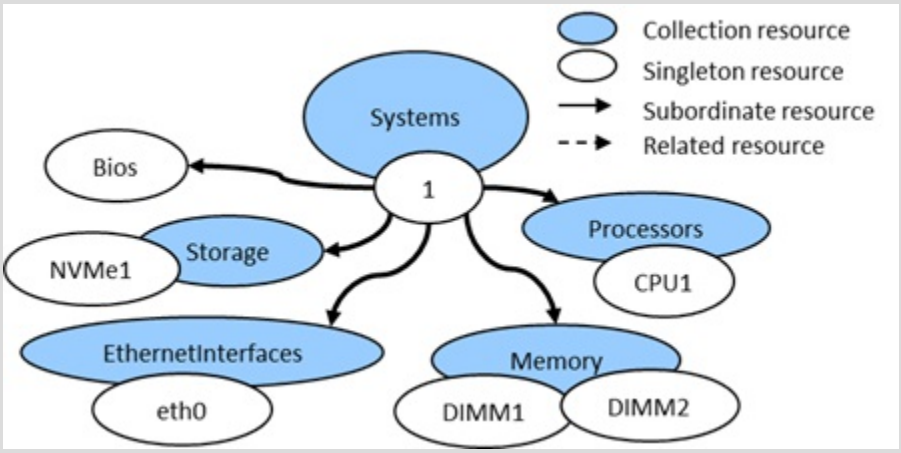

Figure 4 — System BMC systems model

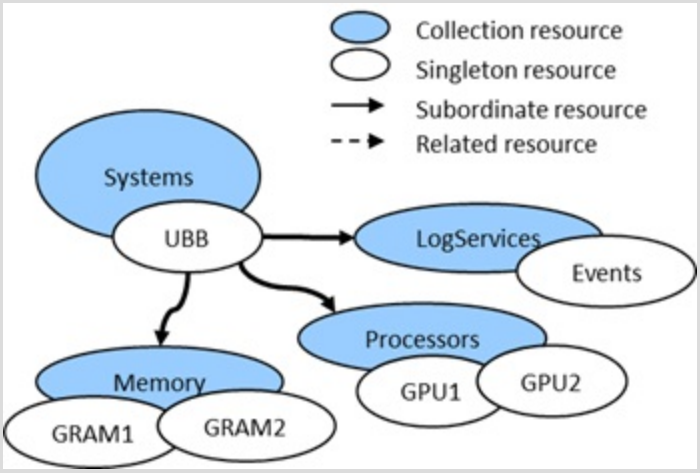

Figure 5 — Compute expansion module BMC systems model

Figure 6 shows the aggregation result of the example system from the previous figures. There is only one `ComputerSystem` resource, with the resources from the compute expansion module are now included in the resultant system. The `LogService` resource named `Events`, `Processor` resources named `GPU1` and `GPU2`, and the `Memory` resources named `GRAM1` and `GRAM2` are aggregated directly from the compute expansion module BMC.

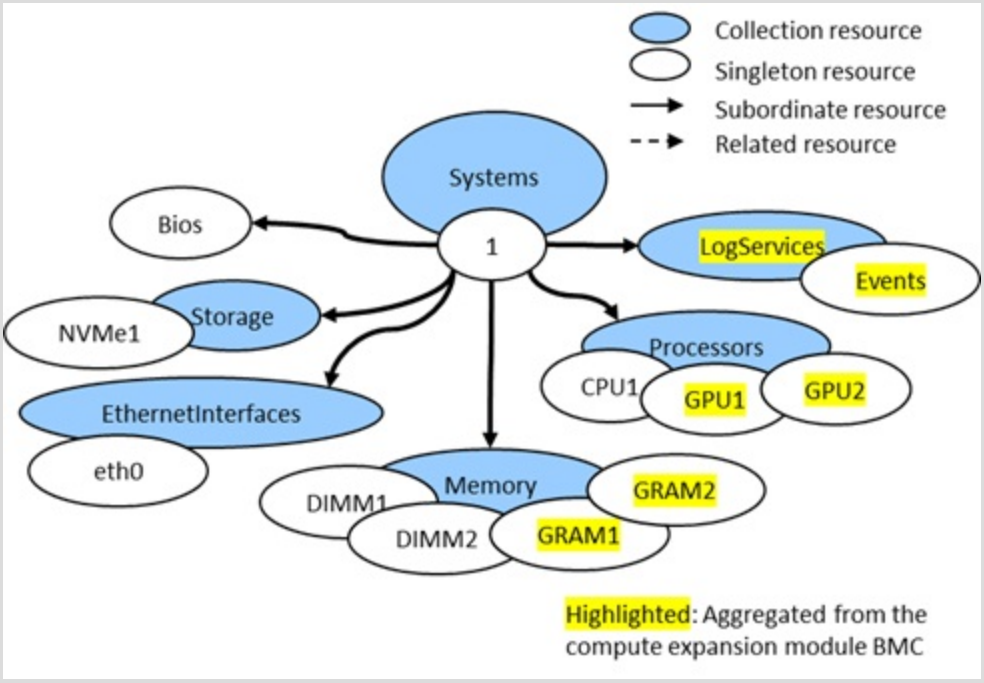

Figure 6 — Aggregated systems model

5 Managers

The members of the `ManagerCollection`, found at `/redfish/v1/Managers`, represent the BMCs or other management controllers monitoring devices in the system. The compute expansion module BMC models itself and other management controllers, if applicable, `Manager` resources. The system BMC is expected to insert all `Manager` resources discovered in the compute expansion module BMC into its own `ManagerCollection`.

Figure 7 and Figure 8 show an example system with a UBB prior to aggregation where:

- A `Manager` named `BMC` is modeled by the system BMC to represent itself.
- A `Manager` named `BMC` is modeled by the compute expansion module BMC to represent itself.

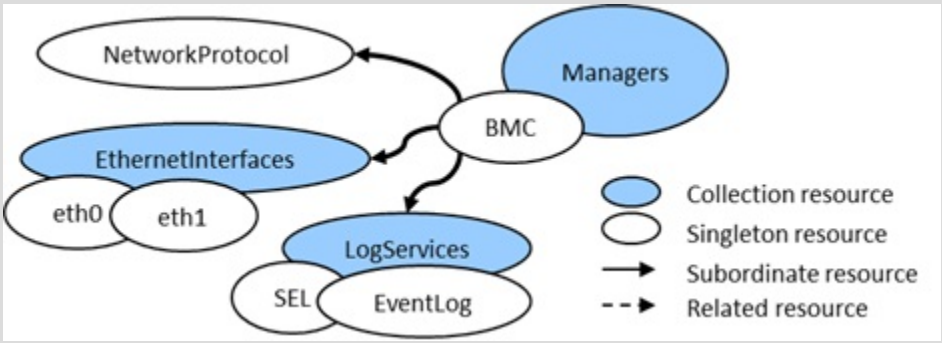

Figure 7 — System BMC managers model

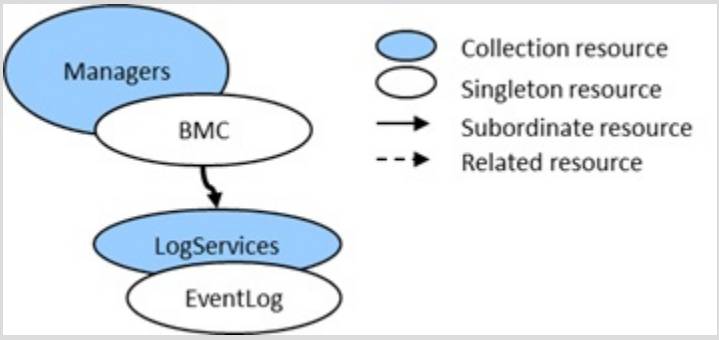

Figure 8 — Compute expansion module BMC managers model

Figure 9 shows the aggregation result of the example system from the previous figures. There are now two members

of the `ManagerCollection` and the `Id` of the compute expansion module BMC was renamed to `UBB-BMC` to prevent naming collisions.

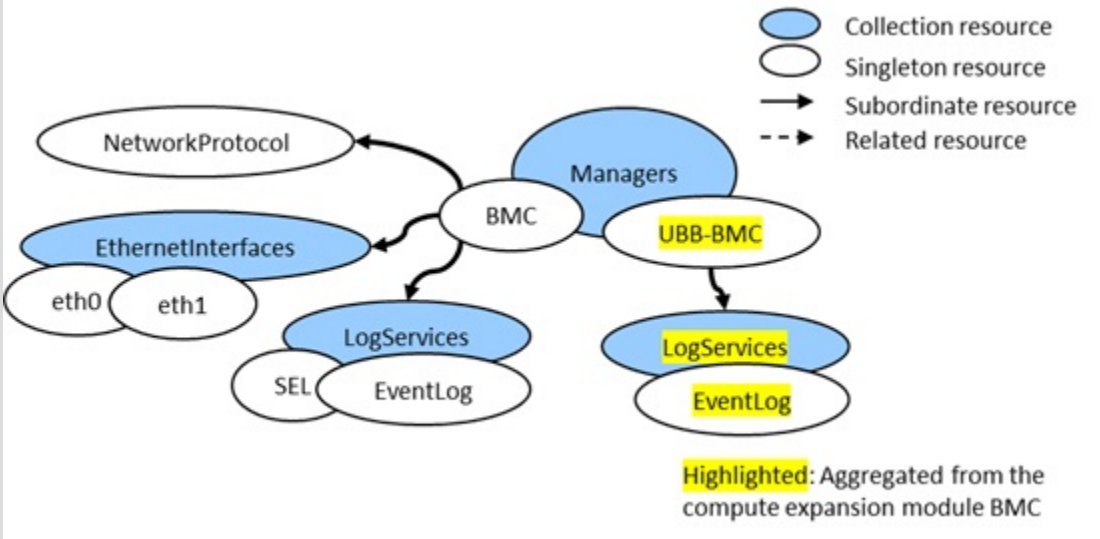

Figure 9 — Aggregated managers model

6 Fabrics

The members of the `FabricCollection`, found at `/redfish/v1/Fabrics`, represent the different data planes managed by the service.

A compute expansion module contains elements of a fabric that share the same data plane as the rest of the system. However, the compute expansion module BMC is unlikely to be aware of elements of the fabric outside of its module and only represent its view of the fabric. The compute expansion module BMC might also break down the fabric representation on a port-by-port basis due to this limitation. For these types of fabrics, the system BMC is expected to merge the respective `Fabric` resources of the compute expansion module BMC into its appropriate `Fabric` resources based on the design of the system. The `FabricCollection` would not grow as a result of this, but other resources subordinate to the system BMC's `Fabric` resources would be extended with resources from the compute expansion module BMC. This would include aggregating subordinate resources and members of collections, such as the `SwitchCollection`, for the system BMC's `Fabric` resources.

A compute expansion module might contain other fabrics that are self-contained in the module. For these types of fabrics, the system BMC is expected to insert the respective `Fabric` resources discovered in the compute expansion module BMC into its own `FabricCollection`.

Figure 10 shows an example fabric topology of a system with a compute expansion module. In this figure there are two fabrics: a PCIe fabric shown with red lines and a cache coherency fabric shown with blue lines. The PCIe fabric crosses the boundary between the compute expansion module and the rest of the system, whereas the cache coherency fabric is entirely inside the module.

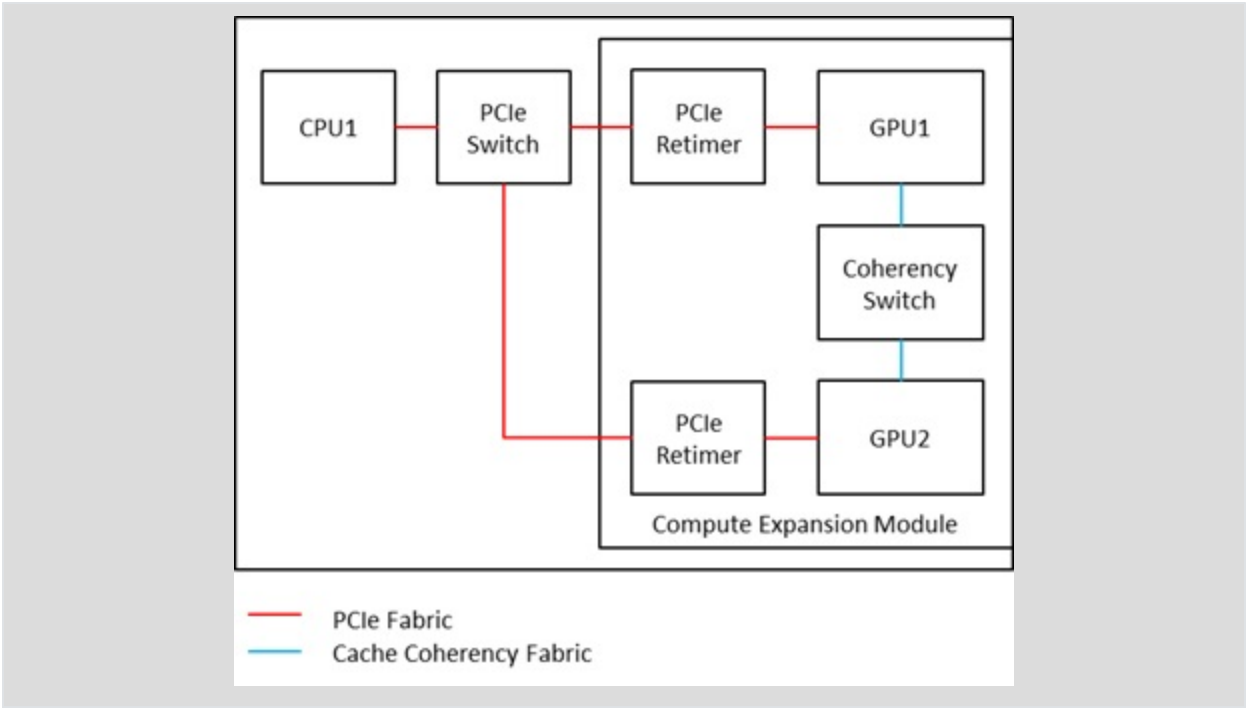

Figure 10 — Example fabric topology

Figure 11 and Figure 12 show the above example system prior to aggregation where:

- A `Fabric` named `PCIe` is modeled by the system BMC to represent the PCIe view of the system.
  - This contains a `Switch` named `Switch1` to represent the PCIe switch directly connected to the CPU in the diagram.
- Two `Fabric` resources named `PCIe-1` and `PCIe-2` are modeled by the compute expansion module BMC to represent the two PCIe segments on the module.
  - The compute expansion module BMC's view of the PCIe topology does not go far enough to know they are part of the same data plane.
  - Each `Fabric` resource also contains a `Switch` named `Retimer` to represent the PCIe retimers connected to the GPUs.
- A `Fabric` named `Coherency` is modeled by the compute expansion module BMC to represent the cache coherency fabric inside the module.
  - This contains a `Switch` named `1` to represent the cache coherency switch connecting the two GPUs in the diagram.

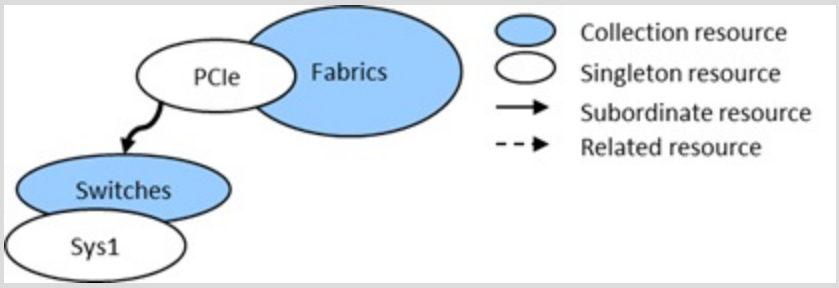

Figure 11 — System BMC fabrics model

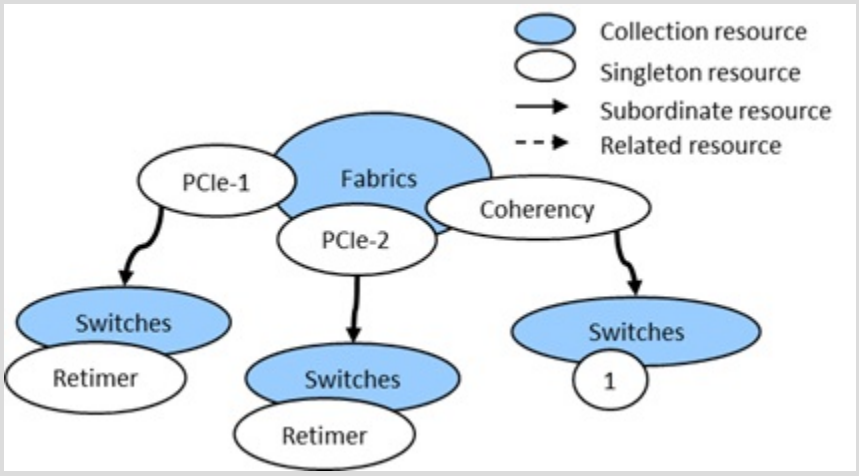

Figure 12 — Compute expansion module BMC fabrics model

Figure 13 shows the aggregation result of the example fabrics from the previous figures. The `Fabric` resources `PCIe-1` and `PCIe-2` were merged into the existing `Fabric` resource named `PCIe` on the system BMC and incorporated the two PCIe retimers, renamed to `Retimer1` and `Retimer2`. The `Fabric` named `Coherency` was aggregated as its own `Fabric` resource since it did not connect to any of the existing data planes in the rest of the system.

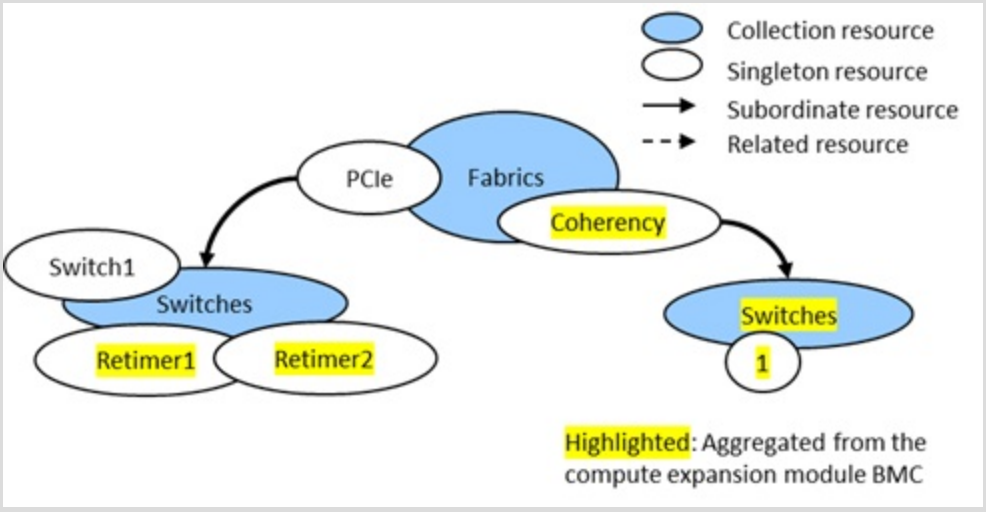

Figure 13 — Aggregated fabrics model

7 Telemetry service

The `TelemetryService` resource, found at `/redfish/v1/TelemetryService`, contains methods for providing metadata for metric-related properties and building metric reports for collecting into time series databases. If supported by the compute expansion module BMC, there are several ways a system BMC can aggregate this service. This list is not exhaustive and there are other possible solutions.

If the system BMC does not implement the `TelemetryService` resource, the system BMC is expected to aggregate the `TelemetryService` resource from the compute expansion module BMC. No special merging is required for this approach.

If the system BMC implements the `TelemetryService` resource, but does not provide external users with fine-tuned configuration for populating metric reports with user-specified properties, the system BMC is expected to insert the following resources discovered in the compute expansion module BMC into its own resource collections:

- `MetricDefinition` resources found at `/redfish/v1/TelemetryService/MetricDefinitions`.
- `MetricReportDefinition` resources found at `/redfish/v1/TelemetryService/MetricReportDefinitions`.
- `MetricReport` resources found at `/redfish/v1/TelemetryService/MetricReports`.

If the system BMC implements the `TelemetryService` resource and allows external users to specify the contents and scheduling of metric reports, the system BMC is expected to grow its `MetricDefinitionCollection` resource based on the new metrics that can be acquired from the compute expansion module BMC. The system BMC might use the `MetricDefinition` resources from the compute expansion module BMC to supply pertinent metadata about metrics, but it's also possible for the system BMC to generate these entries on its own. The system BMC can also either create entirely new `MetricReportDefinition` resources, update existing `MetricReportDefinition` resources, or leave this entirely to the external user to configure. `MetricReport` resources are generated based upon the policies and configuration of the telemetry service of the system BMC.

8 Other services

There are other services that the compute expansion module BMC might implement for the system BMC to leverage for its own purposes. Since these other services are internal to the overall solution, the system BMC is not expected to aggregate them or their subordinate resources.

The `EventService` resource, found at `/redfish/v1/EventService`, contains event notification settings and subscriptions. The system BMC might use this as a method of receiving alerts from the compute expansion module BMC so that it does not have to periodically poll it.

The `SessionService` resource, found at `/redfish/v1/SessionService`, contains session information and methods to create new sessions for a Redfish service. The system BMC might use this to establish communication with the compute expansion module BMC to avoid using HTTP Basic authentication with every request.

The `AccountService` resource, found at `/redfish/v1/AccountService`, contains user accounts, roles, and other access policies for a Redfish service. The system BMC might use this to manage its credentials and other access controls with the compute expansion module BMC.

9 Appendix A: References

- Open Accelerator Infrastructure (OAI) - Universal Baseboard (UBB) Base Specification r2.0 v1.0, https://www.opencompute.org/documents/oai-ubb-base-specification-r2-0-v1-0-20230919-pdf
- OCP GPU & Accelerator Management Interfaces Version 0.9, https://www.opencompute.org/documents/ocp-gpu-accelerator-management-interfaces-v0-9-pdf
- DMTF DSP0266, *Redfish Specification*, https://www.dmtf.org/dsp/DSP0266

10 Appendix B: Change log

Version	Date	Description
1.0.0	2025-04-09	Initial release.