### ComputerSystem creation via Constrained Composition through the Composition Service

Diagram, engineering drawing

Description automatically generated

Figure 1: The Composition Service defines Resources Blocks objects and handles Composition Requests

|  |  |
| --- | --- |
| Use Case Description | * User requests a new ComputerSystem creation in the form of a Constrained Composition request to the Composition Service. |
| Actors | * User (System administrator, Infrastructure Providers), Composition Service, OFMF & Provider. * The Composition Service is a client of the OFMF that generates ResourceBlocks objects and implements a policy to compose new ComputerSystems. It receives Constrained Composition requests containing the resource constraints that the new system must meet (# CPU, GBs of volatile memory, storage,etc..). |
| Description | * The Composition Service reads Fabrics and Fabric related objects and **based on a set of assumptions (see Comments section)** defines ResourceBlock objects that store separately. * The user describes the resources required by the new CompusterSystem and lists them as part of the request to be sent to the Composition Service. * The Composition Service, scans the resource blocks, selects the suitable ones, and identifies how to setup connection between components. Finally, it requests the creation of a new system and all the connections required to interface the requested components. |
| Comments | * The Composition Service uses extensions for Composition as defined in the RedFish schema standard. ResourceBlocks are defined as the minimum entity for composition of a system; i.e. a computer system will be composed of one or more resource blocks. * The definition of a ResourceBlock depends on attributes of the physical resources. Example: CPU and local DRAM could be part of the same resource block. * For the sake of this use-case take the following assumptions **(to be discussed):**   + A Chassis containing CPUs,local memory and a connection in the fabric (via Port or FabricAdapter) can be represented as ResourceBlock   + A MemoryChunk belonging to a MemoryDomain connected to the fabric via a FabricAdapter can be represented as a ResourceBlock   + Other kind of resources (Accelerators) connected to the Fabric via a Fabric Adapter can be represented as a ResourceBlock * The Composition Service in this use-case must define ResourceBlocks and Links to the physical resources belonging to the each ResourceBlock. * The list of assumption above is far from exhaustive and we should explore more complex and dynamic cases to see if they hold. Examples:   + Dynamic creation/deletion of Memory Chunks   + Multiple MemoryChunks under the same MemoryDomain   + Multiple resources connected to a single MemoryDomain or FabricAdapter * The final list of assumptions will represent a method to describe composable resources that both Clients, OFMF and Providers must adhere. * To support QoS based composability, Fabric and Fabric connected resources must export enough information to allow the Composition Service to apply the desired composition policy. Some of these information could be latency, bandwidth, etc.. Also, the Composition Request must be able to deduce the expected latency between fabric attached resources if it wants to implement latency-based composition policies. |
| Input Data | * The User provides a description of the resources required by the new ComputerSystem * Example: (taken from accompanying slide deck)   POST /redfish/v1/Systems HTTP/1.1 {  "Processors":   {"Members": [{  "@Redfish.RequestedCount": 2,  "ProcessorType": "CPU",  "TotalCores": 8} ]},  “Memory”:  {“Members”: [{   "@Redfish.RequestedCount": 2,  “MemoryType": “DRAM",  “CapacityMiB": 20480}]} } |
| Preconditions | * Composition Service can explore Fabrics and Fabric attached Resources. * The Composition Service inspects Fabric and fabric attached objects. Applying the assumption described before, it creates ResourceBlock representations.   + It saves such representation in a data store separated from the OFMF. |
| Postconditions | * OFMF Redfish tree contains the description of a new ComputerSystem matching the criteria described in the Constrained Composition request. * OFMF Redfish tree contains an updated view of the Fabric. This could include new Connections. * Actual Fabric hardware state matches the OFMF Redfish description and desired Fabric functionalities are enabled (resource are accessible as part of the new ComputerSystem) |
| Trigger | * Composition Service receives a Constrained Composition Request |
| Normal Flow | Note: For simplicity, flows are described as synchronous operations. Implementation details are beyond the scope of this first draft. Always for simplicity, in both figures I am omitting updates to Redfish trees.   1. When a user sends a Constrained Composition Request, the Composition Service selects the subset of ResourceBlocks that fulfills the request for a new ComputerSystem. In the example, we assume these are called RBA and RBB    1. The policy for composability is beyond the scope of this use-case Note: ResourceBlocks must contain enough resource metadata to implement the desired policy (latency, bandwidth, etc...) 2. It reserves the resources belonging to the selected ResourceBlocks RBA and RBB 3. It connects RBA and RBB by creating a Connection between fabric attached resources belonging to the respective ResourceBlocks. This translates to a POSTs/PATCHs Connection sent to the OFMF. 4. The OFMF forwards each request to the Provider in charge of the Fabric attached resources. 5. The Provider translates each request in actions to be carried out by the Fabric Manager. 6. Once all resources are configured and ready, the Composition Service creates a new ComputerSystem. 7. Finally, the Composition Service returns the new ComputerSystem description to the user. |
|  |  |