Western Digital

Open Composable API

Western Digital – Jeff Nicholson, Mark Miquelon
10.29.2018

Scale-out Data Center Challenges

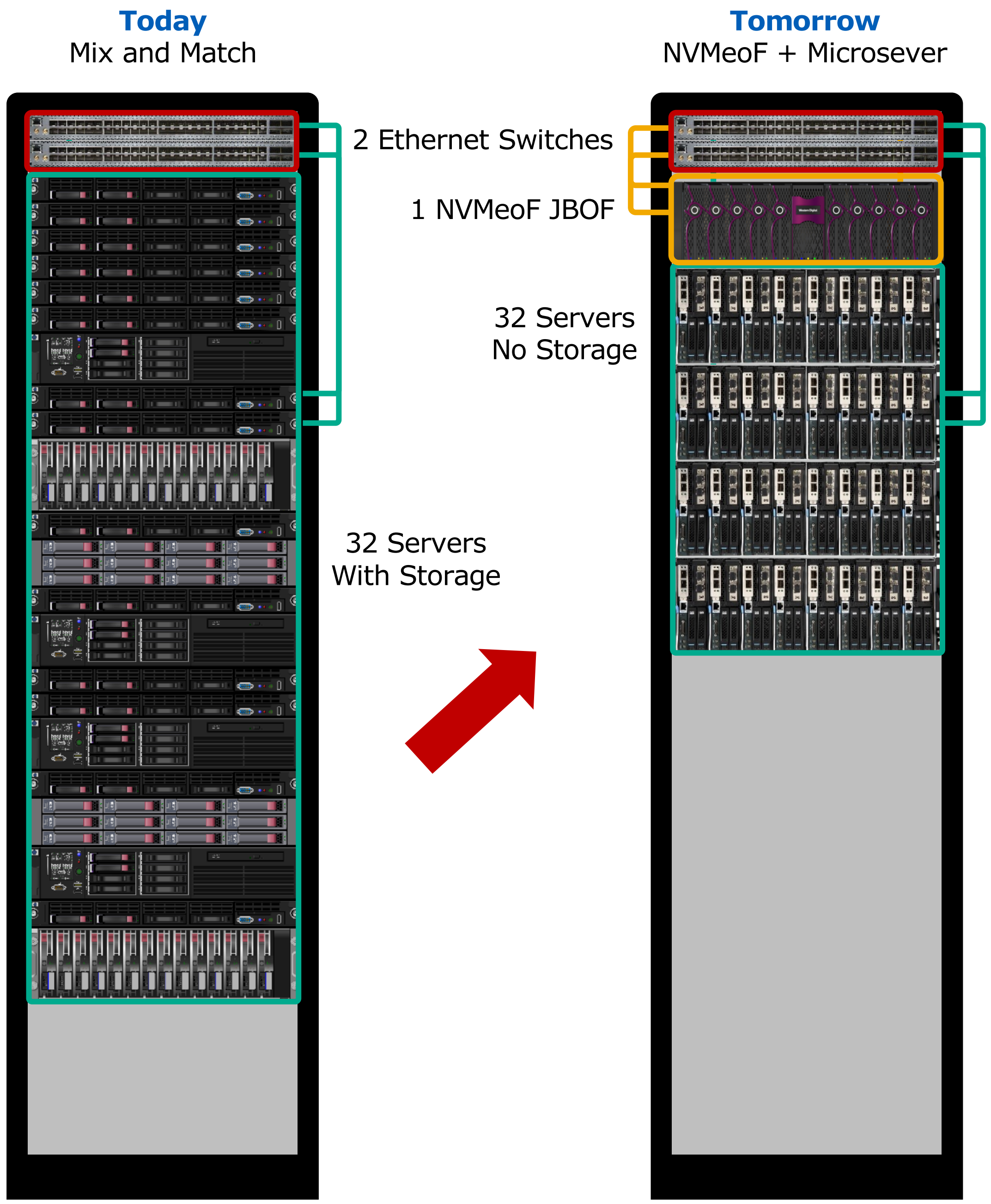
Why we need an open, composable architecture

SKU Cancer	• Many unique or customized platforms with continual qualification efforts • Different HW/BIOS/Firmware/Drivers/OS versions cripple interoperability matrix • Management scripts constantly need to be adjusted for hardware specific handling
Stranded Capacity	• Standardized platforms don’t match application needs exactly • Resources go underutilized (Idle CPU cores, unused DRAM or SSD capacity) • Cannot reassign resources where they are needed as trapped in hyperconverged node
Unpredictable Growth	• Hardware deployment decisions must be made long before hardware is used • Application growth is often unpredictable but hardware capabilities are fixed • Cannot easily grow or shrink resources to adjust to growth rate

Scale-Out DAS Replacement

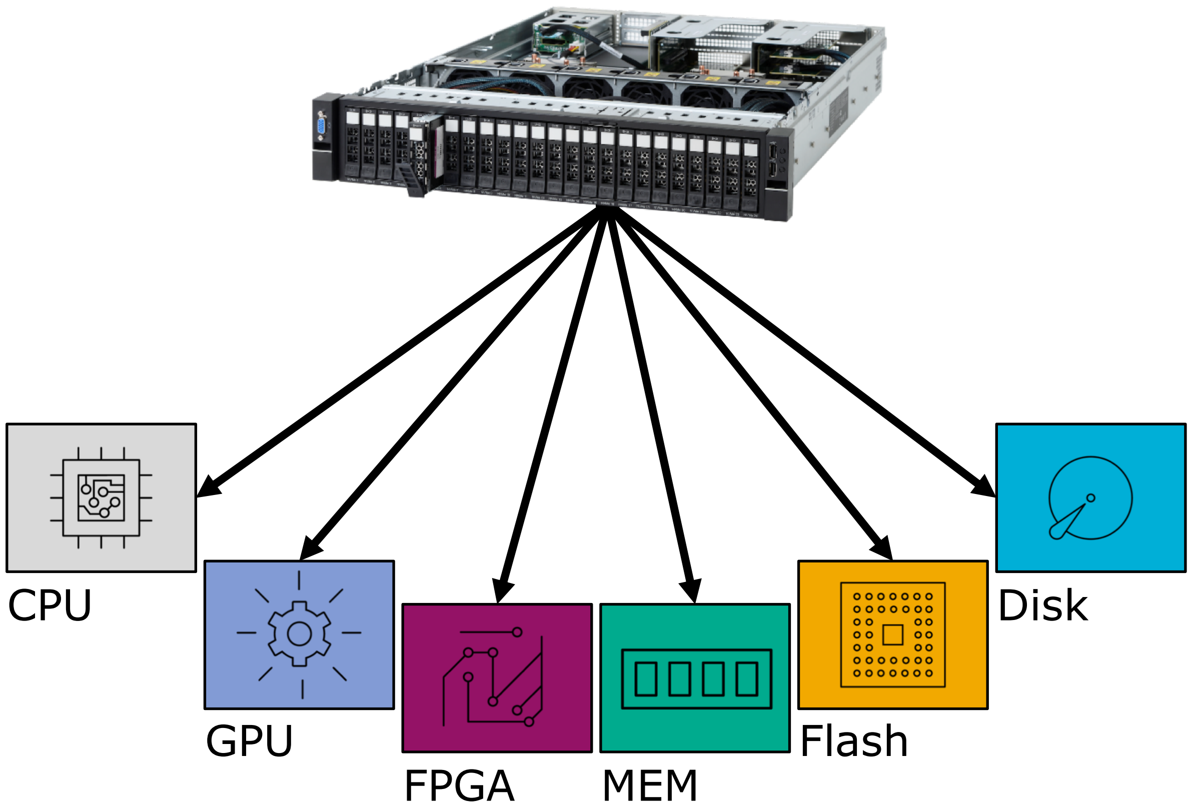
How NVMeoF addresses customer objectives

- Flexibility
  - Add/Reduce/Reassign storage without reboot
  - Applications can move within the rack
  - All servers can see all drives
  - Apps restart on different server if a server fails
- Simplicity
  - Multiple server SKUs reduced to a single model
  - Faster development time with less HW variants
  - Reduced interoperability matrix to test
- Efficiency
  - Partially populate compute & storage enclosures
  - Grow compute & storage at predictable intervals
  - Mix app types in the cluster (CPU or IO heavy)
  - Maximize CPU cycle utilization
  - No boot SSDs – Single R/O boot image



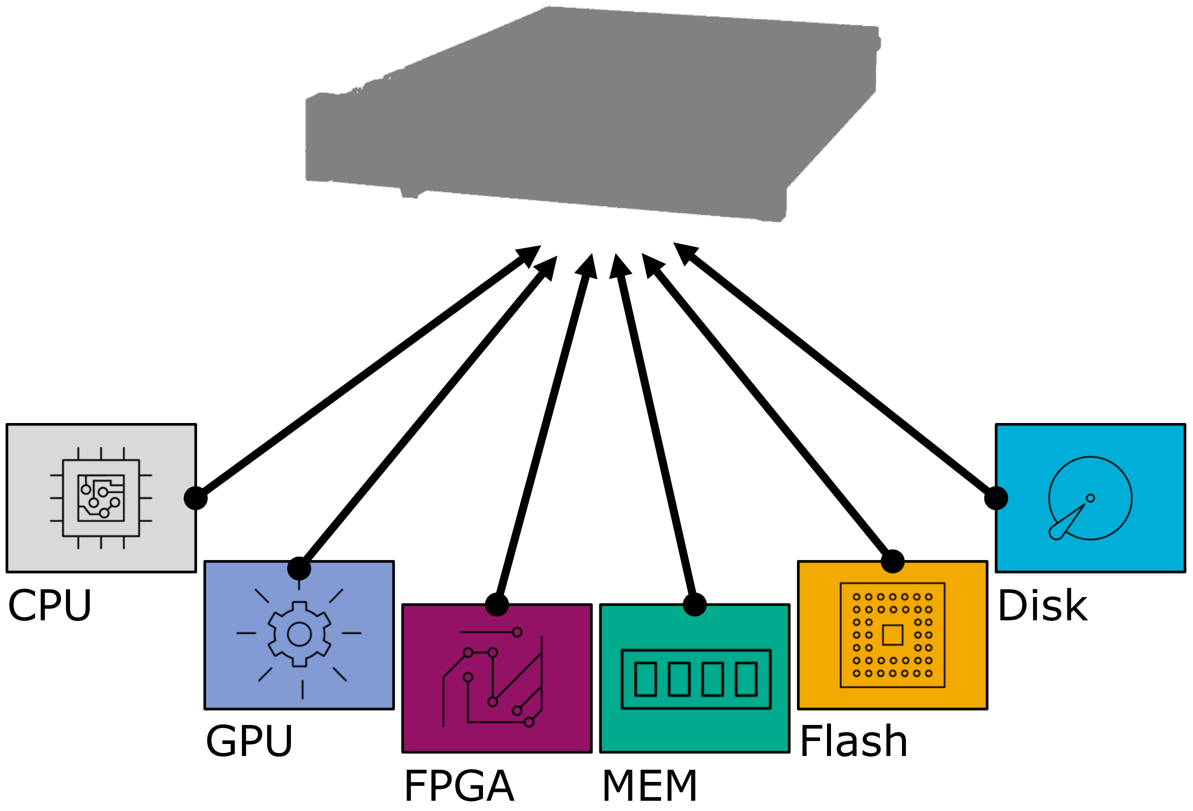
Terms

Disaggregation



Pull hardware components from the server so they can be efficiently pooled

Composability



Orchestrate virtual systems that can be optimally sized to the task

Composable Device Tenets

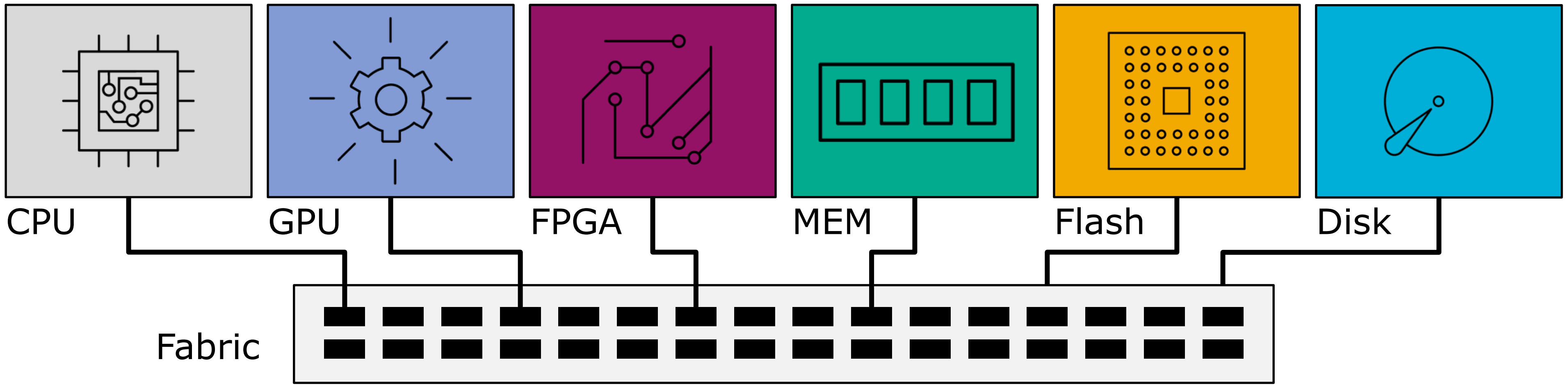
All are required for a device to be classified as "Composable"

Disaggregated Resource	• Single type of pooled resource • Storage, compute, network, memory, PDU
Fabric Attached	• Directly connected to the fabric – Has an address, WWN, etc. • Ethernet, Infiniband, FC, Gen-Z, PCIe
Self Partitioned	• Has capability to partition its resource • Abstracts the underlying hardware – e.g. SMR/MAMR/HAMR
Multitenant Sharing	• Can share resource partitions with many different initiators • Enforces fairness / QoS to prevent noisy neighbor challenges
Device Focus	• Device focuses on a single function (i.e. not a system) • Data services & orchestration happen at a higher level

Western Digital.

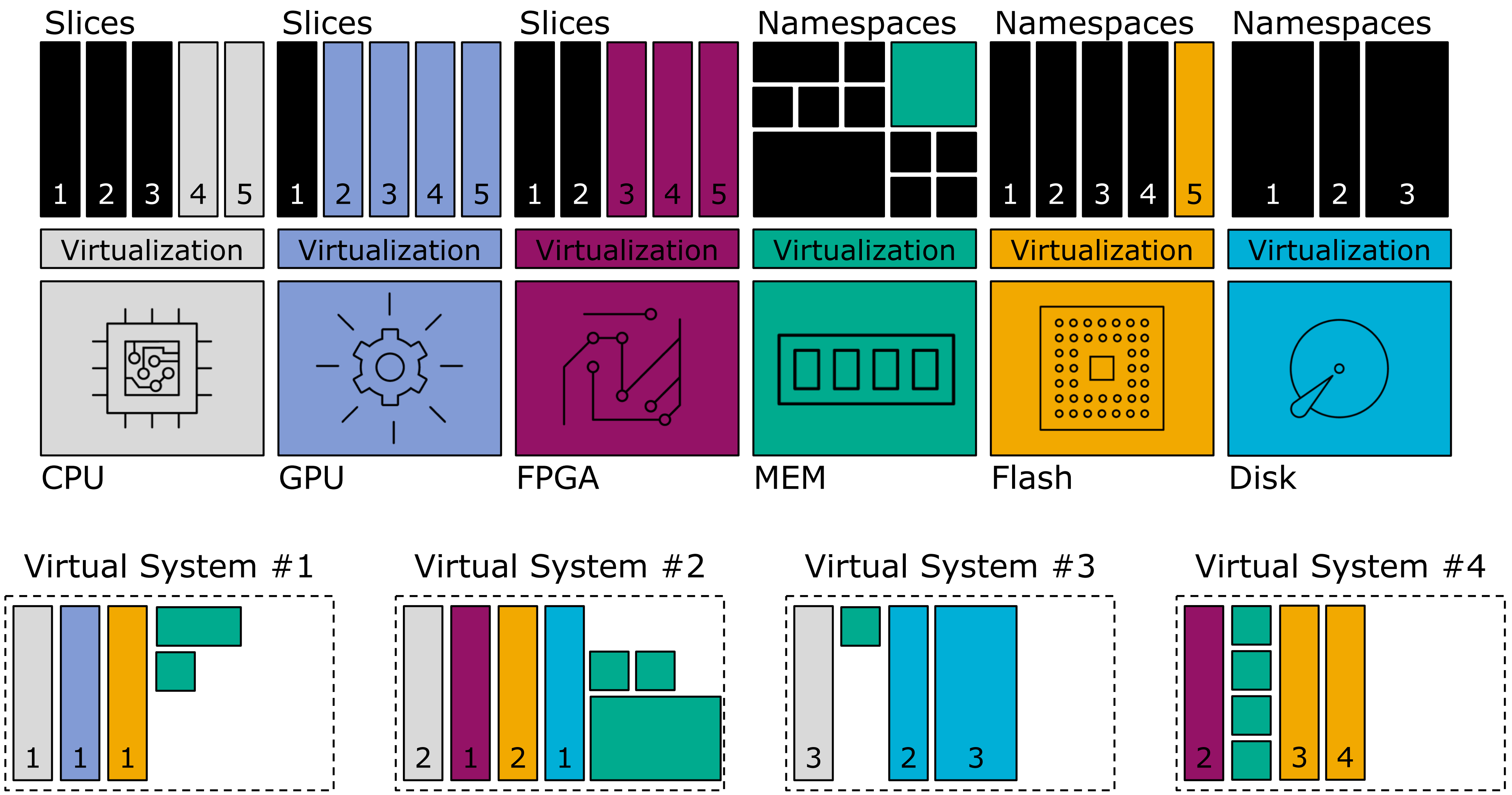
Logical Composability

Virtual systems composed of device partitions



- No physical systems – Only virtual systems – Procured from separate suppliers
- Each element provides a service that is offered over the network
- No established hierarchy – CPU doesn’t ‘own’ the GPU or the Memory
- All elements are peers on the network & they communicate with each other

Logical Composability



Open Composable API

REST Schema Guiding Principles

- Simple URI patterns for all resources
- Enable rapid discovery of resources
- Reduce complexity in the model
- Aggregated responses from composed devices
- Interactive topology traversal (User at a browser)

Open Composable API

CIM Model – Enhancement Opportunities

- Compress the model in 3 ways:
  - "Collections" are rolled into the Resource Type as "plural" of the type:
    - GET /Volumes returns the full list of Volume Resources
    - GET /Volumes/{id} returns the specific Volume instance
  - “Management Services" are rolled into the Resource Type:
    - Create, Modify, and Delete are executed on the Resource:
    - Create: POST /Volumes (params)
    - Modify: PUT /Volumes/{id} (params)
    - Delete: DELETE /Volumes/{id}
  - "Associations" inherently provided by the Resource Type:
    - Media contains links to Storage Pools
    - Storage Pools contain links to Media that makes up the Pool; contains links to Volumes exported by the Pool
    - Volumes contain links to Storage Pools that make up the Volume; contains the links to Storage Endpoints exposing the Volume
    - Storage Endpoints contain links to attached Volumes; contains links to Paths to remote/consuming Hosts
    - Paths contain links to Storage Endpoints; contains links to Host Endpoints

Open Composable API

REST-Based Resource-Oriented Architecture

- Simplified URI patterns based on Resource-Oriented Architecture (ROA):
  - All resources have an address; specific URI to directly get to the resource representation
  - All resources provide linkable navigation or associations to other resources
  - All resources provide a uniform interface; HTTP GET, POST, PUT, DELETE (HEAD, OPTIONS)
  - All resources operate statelessly; no prior or post state requirements (i.e. no sessions)

- HTTP Methods for the "verbs"
  - GET (Retrieve)
  - POST (Create or Add)
  - PUT (Update or Modify)
  - DELETE (or Remove)
  - HEAD (Ping)
  - OPTIONS (Report which Methods are Allowed per Resource Type)
    - Also returns Resource Schema in Response Body

- HTTP Responses
  - 200 OK
  - 201 Created
  - 202 Accepted
  - 204 No Content
  - 302 Found
  - 304 Not Modified
  - 400 Bad Request
  - 401 Unauthorized
  - 403 Forbidden
  - 404 Not Found
  - 405 Method Not Allowed
  - 409 Conflict
  - 412 Precondition Failed
  - 415 Unsupported Media Type
  - 500 Internal Server Error
  - 501 Not Implemented
  - 502 Service Unavailable
  - 503 Gateway Timeout

Western Digital.

Open Composable API

URI Structure

- Uniform Resource Identifiers (URI) are built with the following pattern:
  - scheme, host, domain, resources, resourceId, …
  - Generic Example:
    - GET http(s)://ip:port/domain/resources/resourceID[/resources/resourceID]...
- Physical Storage Platform Example for a specific Platform instance:
  - GET https://10.20.30.40/Storage/Devices/5000cca232178670
- Logical/Virtual Storage Volume Example for a specific Open Composable API instance:
  - GET $\underbrace{\text{https:}}_{\text{scheme}}\underbrace{\text{//10.20.30.40}}_{\text{host}}\underbrace{\text{/Storage}}_{\text{domain}}\underbrace{\text{/Volumes}}_{\text{resources}}\underbrace{\text{/85023099407f9ac0}}_{\text{resource instance ID}}$
- System, Platform, Device, and Component-Level Resource Identifiers
  - Noun-based URIs that uniquely identify the managed elements with a domain prefix
  - Prefix Types:

/Storage	– Storage Resource Management domain (Platforms, Enclosures, Arrays, HDDs, SSDs, SCMs, Volumes, Pools, Paths, Hosts, Endpoints)
/Compute	– Server Resource Management domain (Compute Server devices, Processors, Endpoints)
/Network	– Network Resource Management domain (Ethernet and Fabric Switches, Ports, Endpoints)
/System	– Physical & Virtual System Resource and Grouping Lists, Overall Converged System Resource Disaggregation Information

Open Composable API

Fabric Device Discovery (WIP)

- Needed a way for a composable device to be easily discovered without authentication
- Use REST verbs for discovery
- GET http://<ip>/Query
- Doorbell response provides current API information including the status and version
- Doorbell response provides the "next level" links to go further into the device
- Client walks network subnet with a GET request to each IP Address (very fast)

```
{
   "Self":"http://<ip>/Query/",
   "SchemaInformationStructure":{
      "Description":"Open Composable API",
      "Version":"0.9.2",
      "URI":"/Query/",
      "OwningOrganization":"TBD",
      "Status":"Preliminary"
      },
   "SystemPlatforms":{
      "Self":"http://<ip>/System/Platforms/",
      "Members":[{. . .}]
   }
}
```

Western Digital.