PLASMA

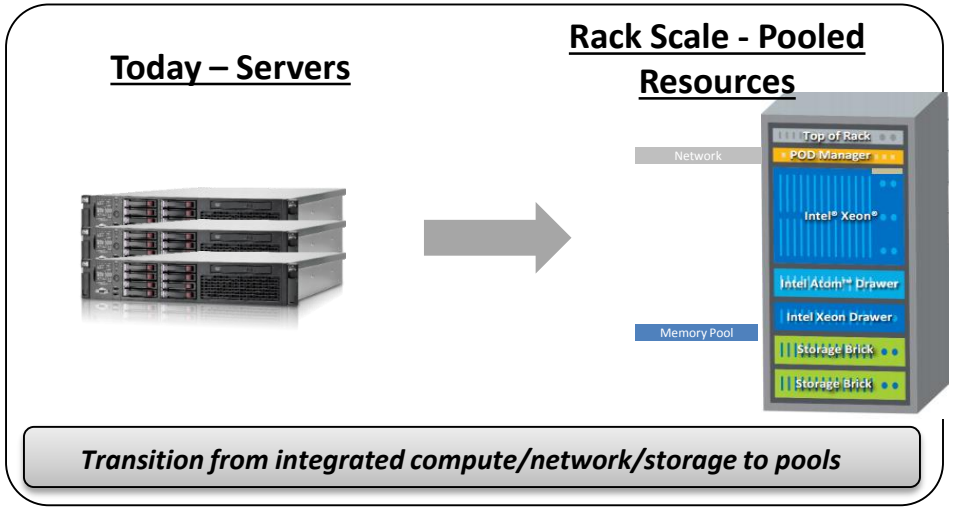
Controller for Rack Scale Design & Redfish

Chester Kuo
Intel @ Datacenter Solution Group

Agenda

- RackScale Design Concept
- Standard
- High Level design
- Why Plasma

Intel® Rack Scale Design

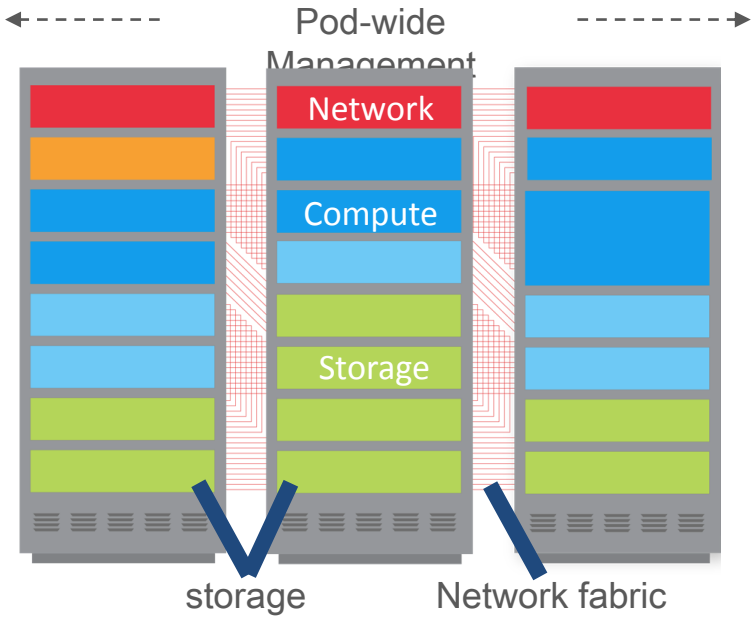


Value Propositions

- **Cost savings** due to resource pooling & avoiding replacement of the full rack due to disaggregated components
- **Simpler Logistics** by avoiding to replace 1000s of racks every couple of years
- **Datacenter level management** allows interoperability of SW and HW layers for rapid deployment

Intel® Rack Scale design & standards like OCP are complementary

Intel RSD is Logical Architecture



OCP is Physical Architecture

Facebook Open Rack V2

- Standard

Industry initiatives – Rack Scale Design alignment

Open Hardware Initiatives

Project Scorpio

opendatacenter.cn

Open Compute Project (OCP)

www.opencompute.org

Standards - Common Management APIs

Redfish

www.redfishspecification.org

DMTF SPMF

Scalable Platforms Management Forum

dmtf.org/join/spmf

Intel builds ecosystems of flexible interoperable solutions being on the leading edge of developing new solutions in partnership with the industry

Redfish™ & Chinook Background

- Redfish™ and ‘Chinook Extensions to Redfish’ refer to industry standard APIs
  - Redfish is focused on modern RESTful management
  - Chinook extends the capabilities of Redfish™ to support SDI
  - Chinook Extensions are layered above Redfish

- Relationship to Intel Rack Scale Design
  - Intel® Rack Scale Design utilizes and builds upon both specifications
  - Neither are Intel Rack Scale Design-specific
  - Redfish provides Intel Rack Scale Design with a credible *industry-standard* foundation

- Both began as small focused efforts, with the intention to submit to existing Standard Development Organizations (SDOs) for completion
  - Redfish was submitted to SPMF (and 1.0 subsequently announced in Aug’15)

Intel® Rack Scale Design API's vs DMTF evolving API's

Redfish™ & Chinook

Intel® Rack Scale Design

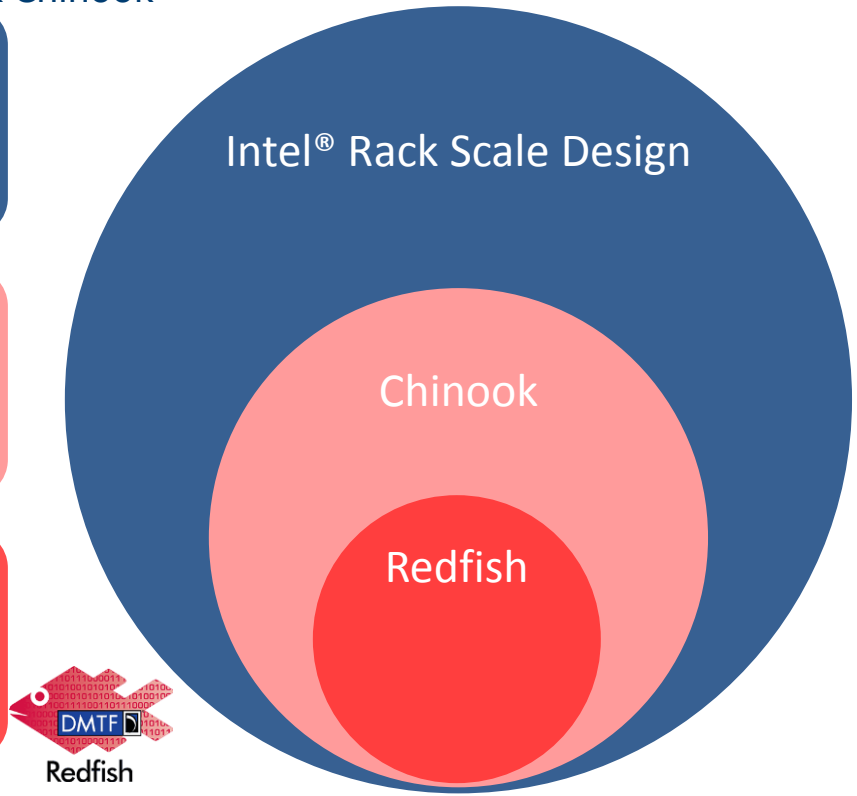
- Partner adoption occurring now – HW & SW
- Incorporates Redfish and Chinook API requirements
- Expected to continue to lead API conformance and contribute to lower levels
- Includes HW requirements
- Includes system behavior requirements

Chinook: Extends Redfish device level interface

- Evolving Standard with A-List team only
- Chinook Extensions do not currently have a public conformance tool available

Redfish: Multi-Node Management API

- Currently Intel® Rack Scale Design encompasses Redfish™ in its entirety for the compute and infrastructure portions. Intel® RSD conformance will mean conformance to the Redfish spec
- Redfish does not currently have a public conformance tool available

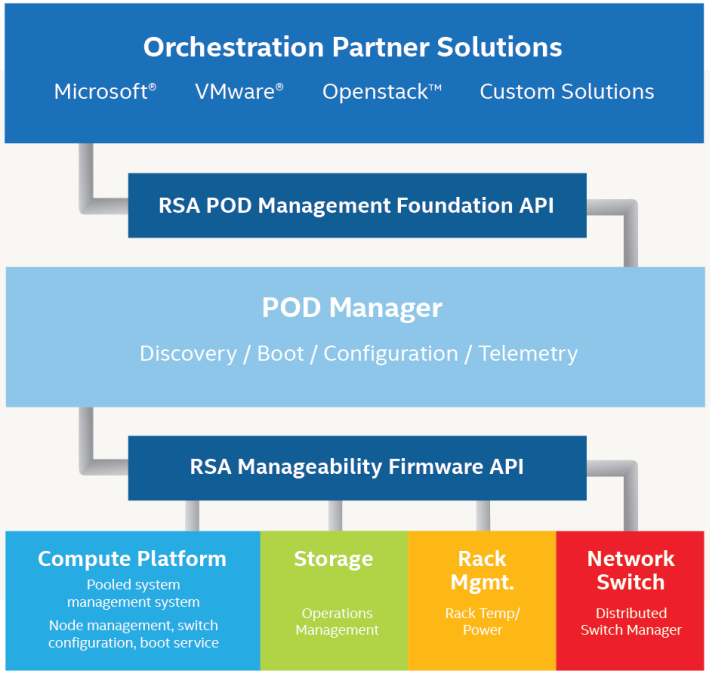


- High Level design & Open source

Management Software Framework

Flexible management architecture allowing for range of implementation options

- Asset & location discovery
- Disaggregated resource management
- Composable system support
- Support compute, network, and storage,



RSA 1.2 Foundational Use Cases

1 - Asset Management

Description: POD Manager provides a detailed inventory of assets in every rack within the POD through a consistent API to the Cloud Manager. PSME and RMM provide PODM the details of all individual components in their respective domains.

1. Discovery: Gather and store asset description: CPU's, Memory, Periph, Storage

2. Rack Management and Pod Manager communicate via RSD API interface for consistent asset management across the POD

RSA 1.2 Foundational Use Cases

2- Machine Assembly

Description: PODM can dynamically create a machine based on the Cloud Managers specified characteristics. Once the assets have been reserved, PODM then can assemble and boot a custom machine.

1. Assemble machine using characteristics supplied by Cloud Mgr through API request. (eg. CPU type, CPU speed, Storage Type, Storage size, Memory type, memory rank)
2. Configure Storage assets, Logical Volumes, and boot targets as requested by the Cloud Mgr.
3. Configure network assets as requested by the Cloud Mgr
4. Boot Machine after its been assembled

RSA 1.2 Foundational Use Cases

3- Datacenter Operations

Description: Allow the Cloud Manager, and Infrastructure operations manager to maintain and effectively operate RSD. Provide visibility and physical asset management across racks.

1. Expose RMM raw telemetry data through PODM: eg. Power & Thermal information
2. Hot Swap assets: detect removal/failure of sled or drawer while rack is powered on
3. Multi-Rack Support – allow PODM to manage several Racks and all of their assets
4. Health status monitoring

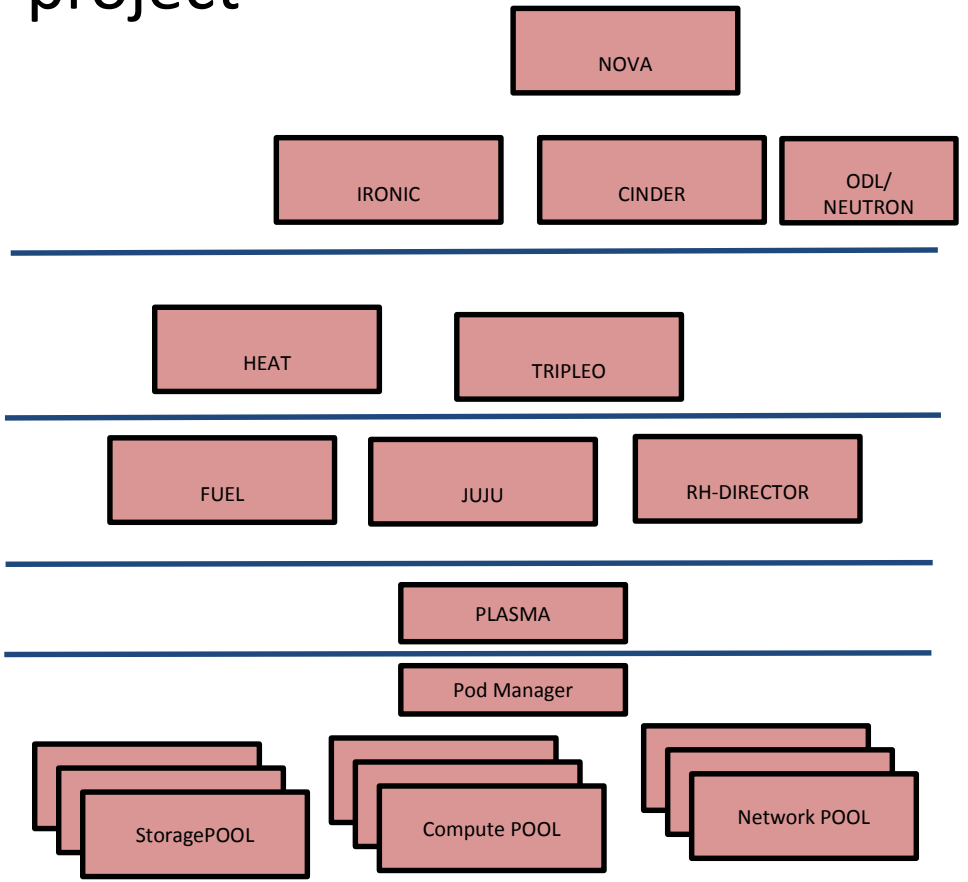
- https://01.org/intelrackscalearchitecture
- http://www.intel.com/content/www/us/en/architecture-and-technology/intel-rack-scale-architecture.html

Docs & github source including

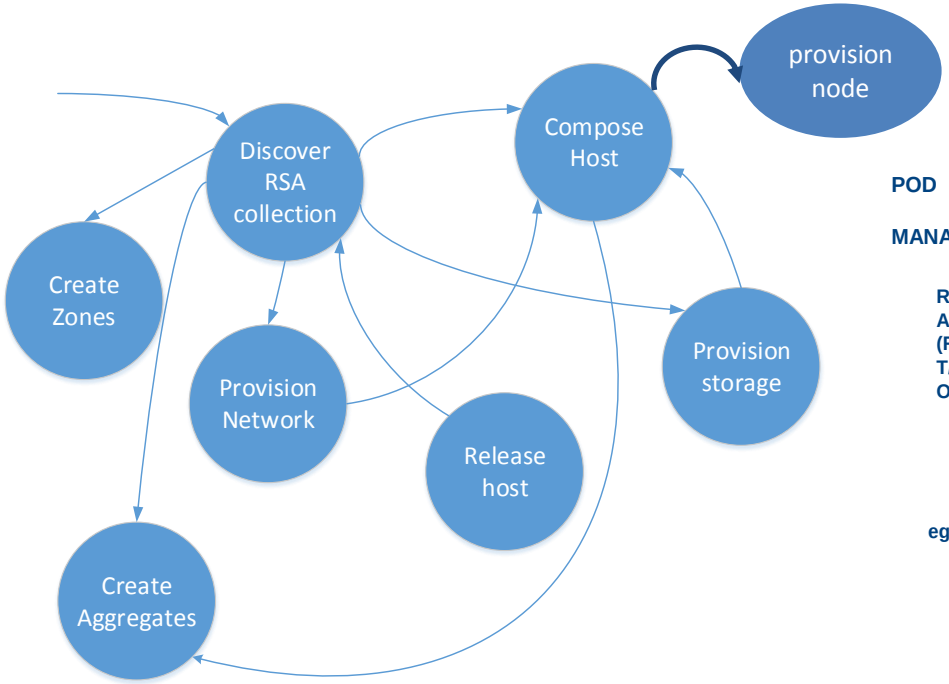
- Why Plasma

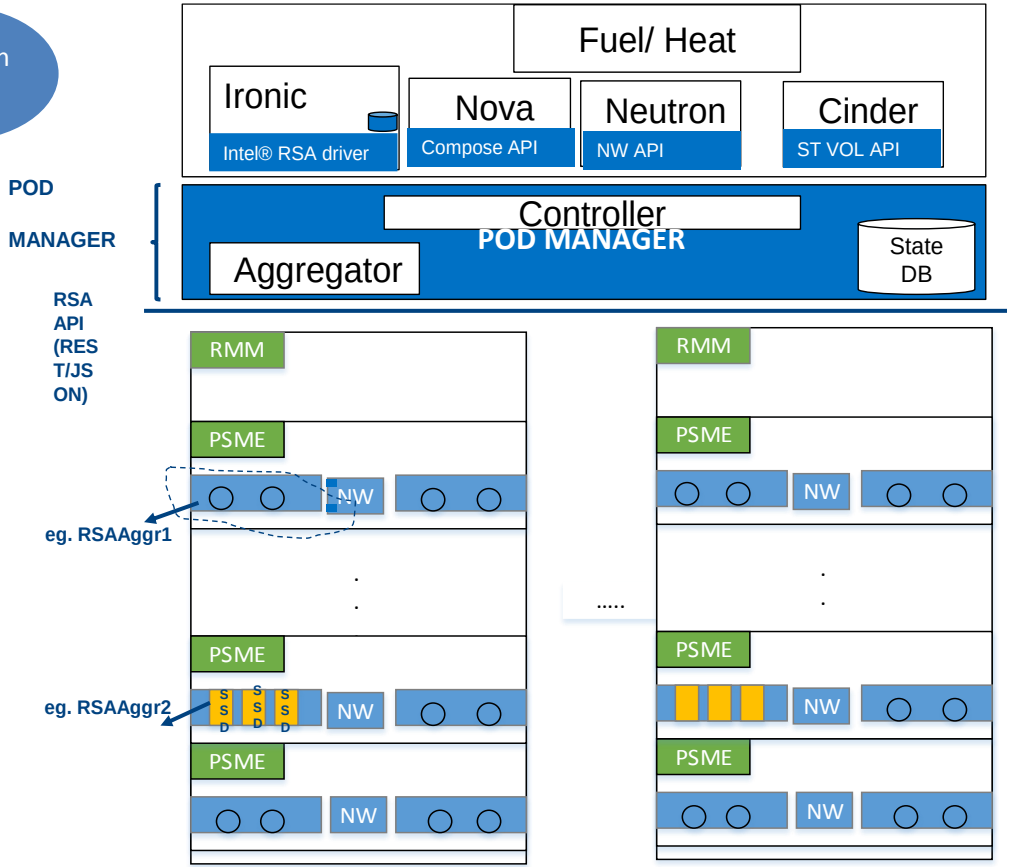
Openstack “Plasma” project

- Elastic HW lifecycle management for Pooled infrastructure – Compose and release
- A controller solution which uses Redfish and RackScale pooling APIs
- Any template based mechanism TripleO, Heat can interact with it
- Partners: OEMS, Openstack ISVs, developers..



RackScale Openstack Integration and lifecycle





Some Blueprints targeted

1. Automatic deployment of bare metal systems using “discovery” and “compose” APIs
2. Infrastructure for a service which does “ undercloud pooled system” management
3. Keystone integration- bp of keystone of Horizon..
4. Lifecycle state management of a system HW using OOB mechanisms
5. Host aggregates based on hierarchy or capability or other
6. OOB network service provisioning with ODL or native Neutron solution
7. Scalable storage provisioning with native library (cinder and Ceph)
8. Create namespace in Open Stack for aggregates key value pairs.
9. Enhance Open Stack to specify scheduler hints as part of launch via the ‘flavor creator”. Instance flow in Horizon

Plasma user story

1. Authentication – similar to keystone
2. Create composed node
    1. with specifc details on cpu type, ram etc as filter
    2. default which is Plasma db
3. GET Discovered Nodes (from DB)
4. POST ReleasePooledSystem
5. Attach storage to Composed node
    1. Type of storage- SSD, NVMe…
    2. Option for full or partial drive
6. POST ComputeStorageMapping
    1. Node ID list to attach
    2. Type of storage
    3. Output: enable volumes on this node to on storage attached to it
7. GET NetworkCompList – pasthrough to PODManager

Plasma project

https://launchpad.net/plasma