OCP Test and Validation: Community Update

EMPOWERING OPEN.

OCP GLOBAL SUMMIT

OCTOBER 18-20, 2022
SAN JOSE, CA

OCP Test and Validation: Community Update

TEST & VALIDATION

Daniel Alvarez Wise, SWE, Meta
Dan Frame, SWE Manager, Google
Arun Koshy, Senior Engineer, HPE

OPEN PLATINUM™

Introduction

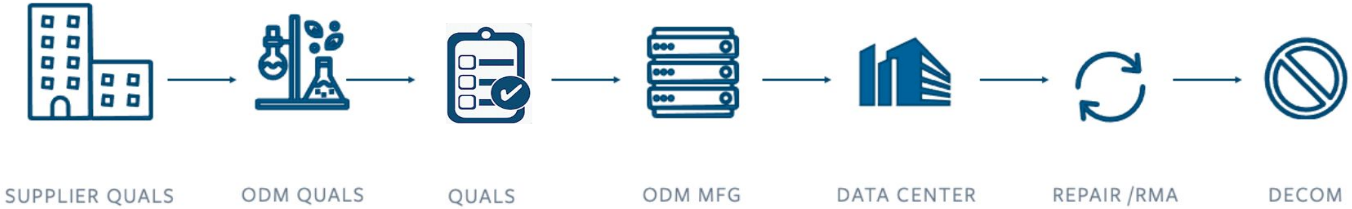
Last year, several different companies with an interest in data center hardware testing got together under the OCP umbrella to solve a common problem: creating portable hardware diagnostics that can work across multiple environments and provide a rich set of standardized result data.

Today, we'd like to give you an update on the effort.

Refresher from Last Year

TEST & VALIDATION

- Problems we are attempting to solve:
  - Acceleration/re-use of diagnostic development and integration efforts at all stages of the product life-cycle.
  - Diagnostic portability across multiple products, environments, and use-cases.
  - Reproduction of test and validation issues across multiple hardware and software partners.
  - Simple sharing of component vendor tests to accelerate RMA and root-cause analysis.



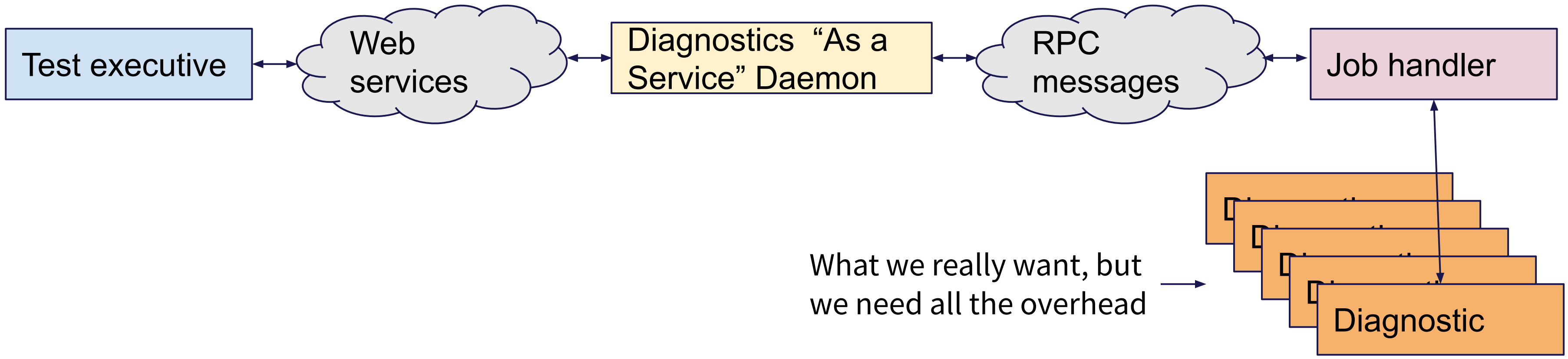
Audience

- Do you provide hardware and tests for your data center partners?
- Do you develop diagnostics and tests for L6/L10 testing?
- Do you work on product design, and qualification?
- Do you develop software, analytics, and dashboards used for product quality?
- Are you interested in data analytics on test data in manufacturing and data center operations?

OCP Test and Validation is for you!

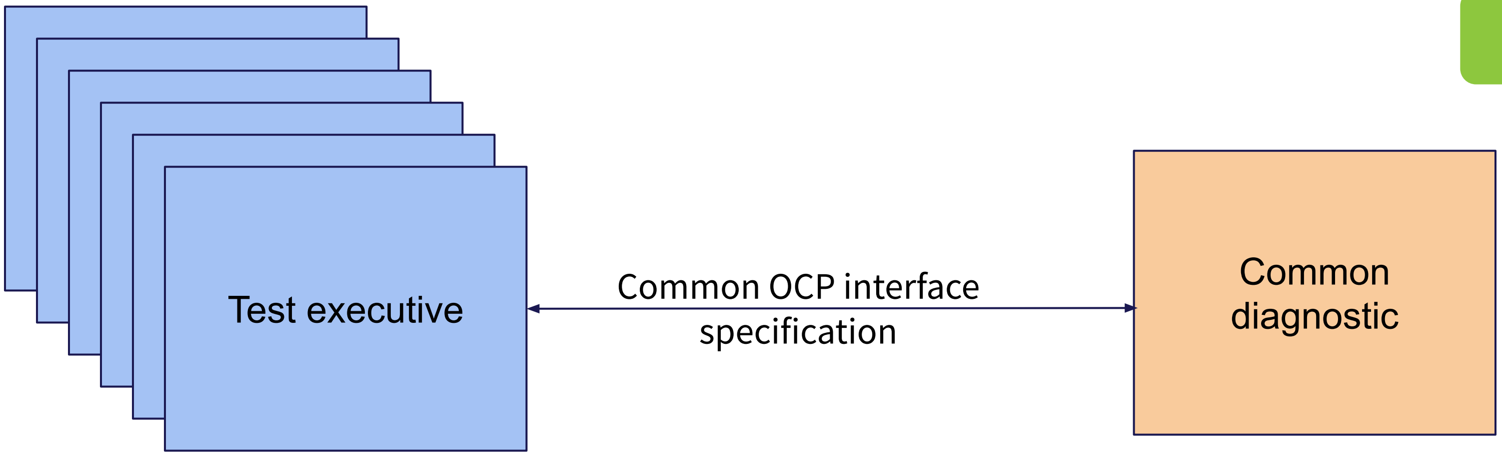
Refresher from Last Year

Before the OCP Diagnostic standard, diagnostics were tightly integrated with their test framework, leading to low portability:



OCP T&V Specification

TEST &
VALIDATION



Test executives from multiple
companies and use-cases

How does it fit in different environments?

TEST AND VALIDATION

Test Environment/Executive

- Provides sequencing for tests.
- Typically integrates with PLM and control systems.
- Records test results to some persistent storage (database, etc).
- Provides arguments to a diagnostic
- May control the lifecycle of a diagnostic.
- May be responsible for installing a diagnostic payload onto a machine under test.
- Typically has final determination of pass/fail or at least the ability to override that.
- May transform OCP diagnostic output to an internal/alternative representation.

OCP Diagnostic

- Parses input arguments.
- Performs actual testing either on or off the device under test.
- Provides a consistent output format.
- Provides pass/fail result which can be overridden by a test executive.

The OCP Diagnostic framework is NOT a test executive.

A test executive typically has dozens of integration points in an organization (i.e. ERP, MES, Data Collection, etc).

By contrast, the diagnostic or test typically only has two integration points, so portability is best achieved by interacting at this level.

What's included in the Core Library?

OCP Diagnostic Core Library

Data Model		DUT Communication Library	Hardware Abstraction Interface
Input Parameter Handling	Output Data Generation	SSH Backend	Redfish Backend

Objectives of OCP T&V Output Specification

1. **Standardized**
2. **Structured**
3. **Portable**
4. **Extensible**
5. **Streamable**

Why it's important?

- Diagnostic reuse across multiple environments allows reproducibility, and lowers NRE development and sustaining costs.
- Rich data model provides the necessary information for repair prediction, anomaly detection, and statistical process control (SPC).
- Sharing component vendor data from the data center in a digestible format to accelerate RMA and root-cause analysis.
- "Off the Shelf" health tests for non-differentiated components accelerates time to market.

Sharing DC Information with Hardware Partners

In many cases, to find ***rare*** problems, we require a hyper-scale sized fleet on deployed machines.

In many cases, hardware vendors request large numbers of tests to be performed and shared which results in an "IT" project to get the results back to them in a format that they can digest.

The OCP diagnostics schema provides a common language for test result sharing and execution.

Examples of Hardware Data Sharing

- Safely sharing information from the Data Center to vendors to help improve their products
  - DIMM Failure and Test information "at hyperscale"
  - Storage "Mean Time to Failure" and "Mean time to data loss" statistics.
- OSFCI (https://osfci.tech)
  - offers OCP members and public a fully loaded CI based on real hardware execution
  - Sharing of hardware and testing frameworks between hyperscalers. E.g. with ConTest.
- Reproduction of Issues
  - NPI work between new servers and components with portable diagnostics for reproducing SW and HW issues.

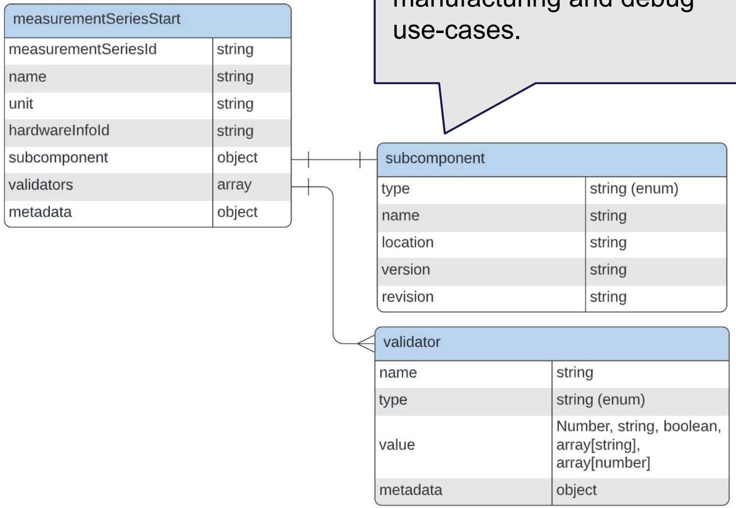
Understanding Test Coverage

The OCP Diagnostic format is aimed at associating diagnosis and parametrics to specific hardware components and ***sub-components***.

This format allows attribution of test coverage to a:

- Field Replaceable Unit
- Subcomponent or Chip
- Netlist, functional block, or Bus
- A logical function

Version 2.x of the OCP test and validation specification includes a cleaner model for "sub-FRU" coverage for manufacturing and debug use-cases.



Enabling SPC and Repair Prediction

The OCP Test and Validation specification provides rich data which has been designed to provide a viable format for powerful AI Root cause Analysis as well as traditional statistical process control capabilities

- Anomaly detection of quality and yield problems
- Prediction of Failures
- Predictions of Repairs
- Cross correlations between process and parameters
- Automatic Alerts

AI Powered Root Cause Analysis

- Real time monitoring of key performance indicators
- Product Quality Problems
- Cp/Cpk Analysis

Statistical Process Control

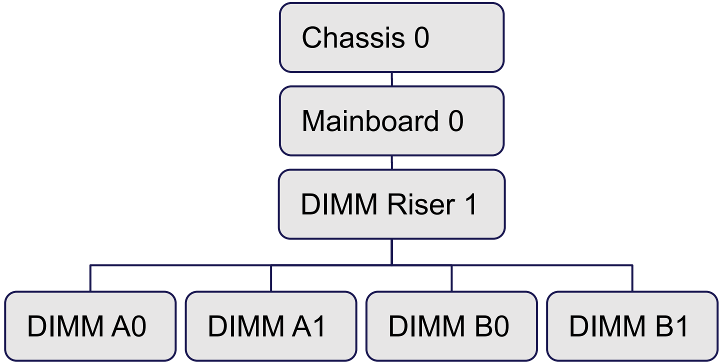
Redfish Device Connectivity

In order to factor in hardware topology, it's critical that master data exists that defines this topology. In some cases, this knowledge has been incorporated into the set of repair rules. For inference models though, this information needs to explicitly exist as ***master data***.

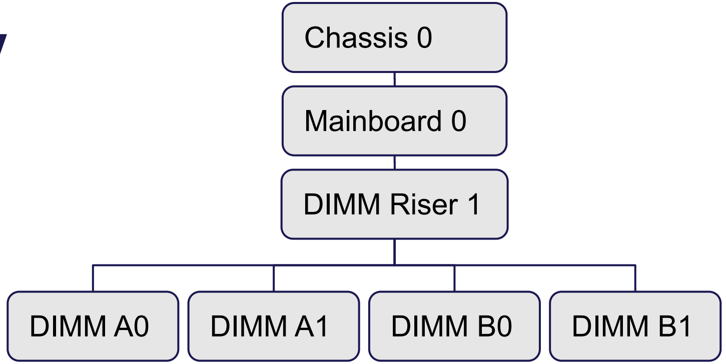
This satisfies two needs:

- It provides a consistent name for the hardware to reference for repairability strategies.

- It provides the topology information



Redfish Device Connectivity




```
{
  "@odata.type": "#Cable.v1_2_0.DIMM",
  "Links": {
      "UpstreamChassis": {
      "@odata.id": "/redfish/v1/Chassis/dimm_riser1"
    },
  },
  "Location": {
    "PartLocation": {
      "LocationType": "Slot",
      "ServiceLabel": "DIMM_A0"
    }
  }
}
```

ServiceLabel and Links in Redfish can be used to define a full location in OCP Diagnostics

```
"hardwareInfos": [
          {
            "hardwareInfoId": "1",
            "computerSystem": "primary_node",
            "manager": "bmc0",
            "name": "primary node",
            "location": "chassis0/MB0/dimm_riser1/DIMM_A0",
            "odataId": "/redfish/v1/Systems/437XR1138R2/Memory/1"",
            "partNumber": "P03052-091",
            "serialNumber": "HMA2022029281901",
            "manufacturer": "hynix",
            "manufacturerPartNumber": "HMA84GR7AFR4N-VK",
            "partType": "DIMM",
            "version": "1",
            "revision": "2"
          }
        ]
```

Unique Location in OCP Diagnostic HardwareInfo from service label and link relationships

https://github.com/google/ecclesia-machine-management/blob/master/ecclesia/lib/redfish/g3doc/topology.md

What’s New

The OCP Test and Validation track has released 2.0 of our diagnostic output specification aimed at enabling statistical process control, and repair prediction.

What's changed and why?

- Hardware Information is now closer aligned to DMTF Redfish specifications for machine management.
- Naming consistency improvements
- More flexible/extensible measurement validators.
- Simplified Measurement Schemas
- Improvements in File/Attachment handling
- Improvements in specifying relationships with data that is associated with a component below the FRU level.
- Versioning Information

JSON Output artifacts

TEST AND VALIDATION

- Output artifact
  - schemaVersionArtifact
  - testRunArtifact
    - testRunStart
    - testRunEnd
    - log
    - error
  - testStepArtifact
    - testStepStart
    - testStepEnd
    - measurement
    - measurementSeriesStart
    - measurementSeriesEnd
    - measurementSeriesElement
    - diagnosis
    - error file
    - log
    - extension

- Standard JSON types:
  - Array, Boolean, Null, Number, Object, String
- Date & time as per ISO-8601

Example - testRunStart


```
{
  "testRunArtifact": {
    "testRunStart": {
      "name": "mlc_test",
      "version": "1.0",
      "commandLine": "mlc/mlc --use_default_thresholds=true --data_collection_mode=true",
      "parameters": {
        "max_bandwidth": 7200.0,
        "mode": "fast_mode",
        "data_collection_mode": true,
        "min_bandwidth": 700.0,
        "use_default_thresholds": true
      },
      "dutInfo": {
        "id": "1",
        "hostname": "ocp_lab_0222",
        "platformInfos": [
          {
            "info": "memory_optimized"
          }
        ],
        "softwareInfos": [
          {
            "softwareInfoId": "1",
            "computerSystem": "primary_node",
            "softwareType": "FIRMWARE",
            "name": "bmc_firmware",
            "version": "10",
            "revision": "11"
          }
        ],
        "hardwareInfos": [
          {
            "hardwareInfoId": "1",
            "computerSystem": "primary_node",
            "manager": "bmc0",
            "name": "primary node",
            "location": "MB/DIMM_A1",
            "odataId":
"/redfish/v1/Systems/System.Embedded.1/Memory/DIMMSLOTA1",
            "partNumber": "P03052-091",
            "serialNumber": "HMA2022029281901",
            "manufacturer": "hynix",
            "manufacturerPartNumber": "HMA84GR7AFR4N-VK",
            "partType": "DIMM",
            "version": "1",
            "revision": "2"
          }
        ]
      }
    }
  },
  "sequenceNumber": 1,
  "timestamp": "2022-07-25T04:49:25.262947Z"
}
```


TEST AND VALIDATION

Example -Measurement


```
{
  "testStepArtifact": {
    "measurement": {
      "name": "measured-fan-speed-100",
      "unit": "RPM",
      "hardwareInfoId": "5",
      "subcomponent": {
        "name": "FAN1",
        "location": "F0_1",
        "version": "1",
        "revision": "1",
        "type": "UNSPECIFIED"
      },
      "validators": [
        {
          "name": "80mm_fan_upper_limit",
          "type": "LESS_THAN_OR_EQUAL",
          "value": 11000.0
        },
        {
          "name": "80mm_fan_lower_limit",
          "type": "GREATER_THAN_OR_EQUAL",
          "value": 8000.0
        }
      ],
      "value": 100221.0,
      "metadata": {
        "@type": "type.googleapis.com/experimental.users.dframe.ocp.ExampleMetadata",
        "exampleField1": "example string",
        "exampleField2": "42"
      }
    },
    "testStepId": "4123"
  },
  "sequenceNumber": 800,
  "timestamp": "2022-08-27T00:03:29.545404Z"
}
```


TEST AND VALIDATION

Example -Diagnosis

TEST AND VALIDATION

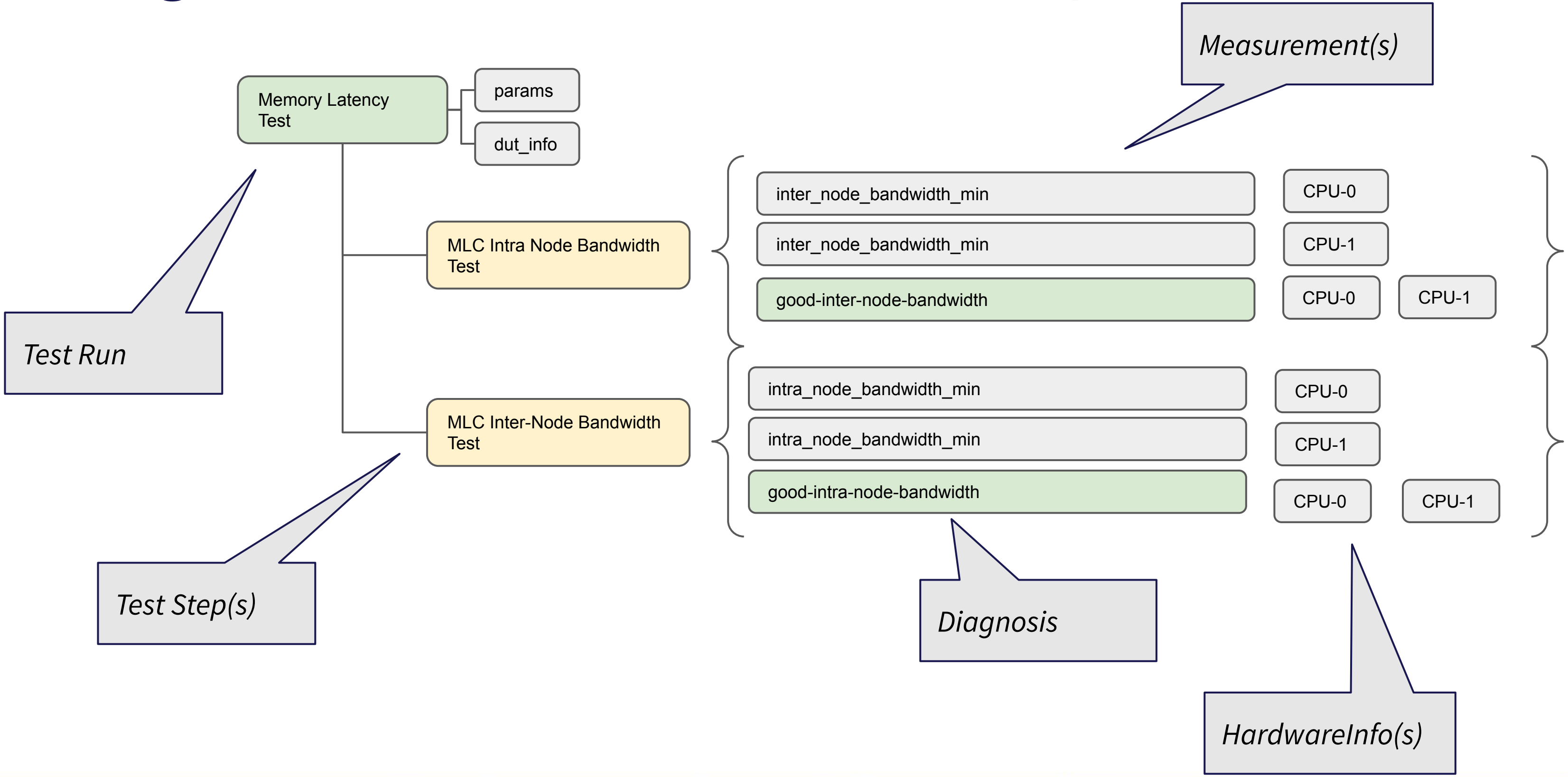
```
{
  "testStepArtifact": {
    "diagnosis": {
      "verdict": "mlc-intranode-bandwidth-pass",
      "type": "PASS",
      "message": "intranode bandwidth within threshold.",
      "hardwareInfoId": "1",
      "subcomponent": {
        "type": "BUS",
        "name": "QPI1",
        "location": "CPU-3-2-3",
        "version": "1",
        "revision": "0"
      }
    },
    "testStepId": "1"
  },
  "sequenceNumber": 1,
  "timestamp": "2022-08-27T00:03:29.545404Z"
}
```

Example - testRunEnd

```
{
  "testRunArtifact": {
    "testRunEnd": {
      "status": "COMPLETE",
      "result": "PASS"
    }
  },
  "sequenceNumber": 100,
  "timestamp": "2022-07-25T04:49:25.262947Z"
}
```

TEST AND VALIDATION

OCP Diagnostics - Result Model Example



What’s Next

- Diagnostics to be available:
    - CPU Diagnostics
    - Storage Performance Diagnostics
    - Peripheral Device Protocol Diagnostics
    - Memory Diagnostics

Call to Action

- Do you provide hardware and tests for your data center partners?
- Do you develop diagnostics and tests for L6/L10 testing?
- Do you work on product design, and qualification?
- Do you develop software, analytics, and dashboards used for product quality?
- Are you interested in data analytics on test data in manufacturing and data center operations?

Consider adopting OCP Diagnostic Core Libraries!

Thank You for Your Contribution

- Adrian Enache (Meta)
- Arun Darlie Koshy (HPE)
- Brennen Direnzo (Keysight)
- Dan Frame (Google)
- Daniel Alvarez Wise (Meta)
- Dylan Hawkes (Google)
- Giovanni Colapinto (Meta)
- Karen Murphy (Meta)
- Marcos Pearson (Google)
- Paul Ng (Meta)
- Thomas Minor (Google)
- Vincent Matossian (Meta)
- Yuanlin Wen (Google)

Join us **TODAY**!!!

- **OCP T&V Output Specification 2.0:**
  - **OCP Test and Validation Output Specification 2.0**
- **Weekly Meeting:**

  Every Wednesday 8AM Pacific Time
- **Project Wiki with latest specification:**

  https://www.opencompute.org/wiki/OCP_Test_and_Validation_Enablement_Initiative

Thank you!

EMPOWERING OPEN.

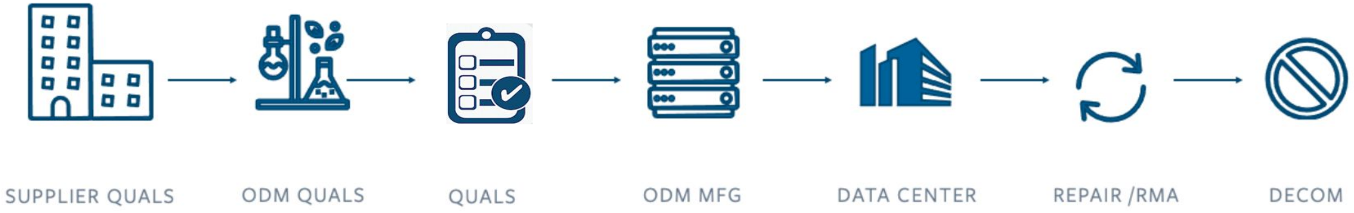
OCP GLOBAL SUMMIT

OCTOBER 18-20, 2022
SAN JOSE, CA

Portable Hardware Diagnostics

The OCP Test and Validation Diagnostic Framework allows support for:

- Many Different Test Executives and Sequencers used for different testing scenarios
- Different Security Requirements
- Different Operating Systems
- Different Data Storage Systems



What's included in the Core Library?

OCP Diagnostic Core Library

Data Model	*"Provides a consistent output model with necessary fidelity to do effective statistical process control, and repair prediction using heuristics or inference modeling."*
DUT Communication Library	*"Handles ensuring that diagnostics can run consistently in labs, manufacturing, and production operations with different security and connectivity requirements."*
Hardware Abstraction Interface	*"Allows diagnostics to be used across platforms with different machine management stacks and implementations."*

Hardware Info/DMTF Redfish

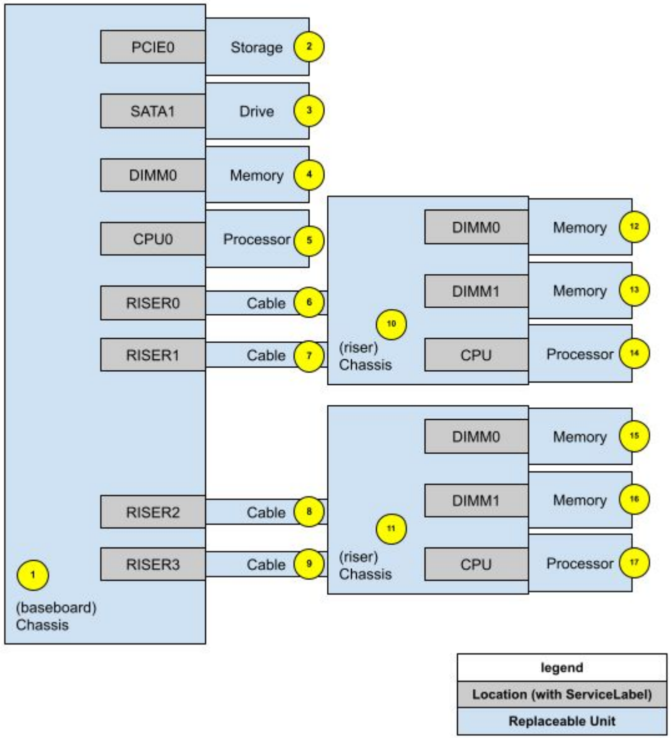
OCP Diagnostic results are correlated with one or more pieces hardware when at all possible.

This hardware must have an identifier with the following desirable properties:

- Identifiers that are stable over the life of a machine, even in the face of component failure.
- Identifiers which are consistent across telemetry producers to allow data correlation from different producers.
- Identifiers which can be understood unambiguously by humans without needing additional tools.

These FRU level identifiers should correlate with the *HardwareInfo* name and location fields. It is strongly recommended that these FRU level identifiers map to [Redfish DSP2046 v2022.1] attributes:

OCP Diagnostics Output Property	Redfish Model Property	Description
HardwareInfo.Location	Location.PartLocation.ServiceLabel	The label of the part location, such as a silk-screened name or printed label.
HardwareInfo.odataid	@odata.id	Unique identifier for a component in the Redfish hierarchy



What's Changed in Device Metadata?

HardwareInfo details

- Aligned specification closer to DMTF models for management concepts.
- Better handling of component level indictments below a FRU for manufacturing testing.
- Cleaned up separation of Location of the hardware component vs. its other physical properties.

SoftwareInfo details

- Added additional fields to handle software on "multi-node" machines.
- Fixed naming inconsistencies.
- Added enumeration of software types (firmware, drivers, etc)
- Added extensible metadata properties.

DUTInfo details

- Fixed naming inconsistencies
- Added extensible metadata properties
- Added durable property to separate the unique identifier of the DUT from mutable hostname for equivalence with hardware information.

What's Changed in Test Information?

daniel

TestRun details

- Added additional information about test parameters used to invoke the test.
- Fixed inconsistencies in status enum naming.
- Simplified parameters to make it easier to implement for people building diagnostics compatible with the specification.

Measurement details

- Flattened time series measurement data to eliminate redundancy.
- Added better timestamp information.
- More closely correlated measurement data to subcomponents below a FRU when necessary.
- Added extensible metadata properties.
- Added more expressive syntax for communicating parametric limits and thresholds.

Logs and Files details

- Added richer metadata on file attachments to help parsing and storing this data in data collection systems.
- Included optional MIME-Types
- Improved naming consistency
- Added extensible metadata properties.

Example - schemaVersionArtifact

The schemaVersion is a new message to allow OCP diagnostics clients to be compatible with newer versions of the specification as it continues to grow and evolve to adapt to more use-cases.

Major Changes are infrequent, and may break backwards compatibility with existing consumers.

Minor Changes are more frequent, and will not break backwards compatibility with existing consumers. A minor change must be compatible with the existing versions JSON schema validation.


```
{
  "schemaVersion": {
    "major": 2,
    "minor": 0
  },
  "timestamp":
"2022-07-25T01:33:46.953314Z",
  "sequenceNumber": 0
}
```