



Document Number: DSP2049

Date: 2017-05-30

Version: 0.2.2b

OCP Hardware Management with Redfish

Information for Work-in-Progress version:

IMPORTANT: This document is not a standard. It does not necessarily reflect the views of the DMTF or its members. Because this document is a Work in Progress, this document may still change, perhaps profoundly and without notice. This document is available for public review and comment until superseded.

Provide any comments through the DMTF Feedback Portal:

<http://www.dmtf.org/standards/feedback>

Supersedes: None

Document Class: Informative

Document Status: Work in Progress

Document Language: en-US

12 Copyright Notice

13 Copyright © 2017 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

14 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
15 management and interoperability. Members and non-members may reproduce DMTF specifications and
16 documents, provided that correct attribution is given. As DMTF specifications may be revised from time to
17 time, the particular version and release date should always be noted.

18 Implementation of certain elements of this standard or proposed standard may be subject to third party
19 patent rights, including provisional patent rights (herein "patent rights"). DMTF makes no representations
20 to users of the standard as to the existence of such rights, and is not responsible to recognize, disclose,
21 or identify any or all such third party patent right, owners or claimants, nor for any incomplete or
22 inaccurate identification or disclosure of such rights, owners or claimants. DMTF shall have no liability to
23 any party, in any manner or circumstance, under any legal theory whatsoever, for failure to recognize,
24 disclose, or identify any such third party patent rights, or for such party's reliance on the standard or
25 incorporation thereof in its product, protocols or testing procedures. DMTF shall have no liability to any
26 party implementing such standard, whether such implementation is foreseeable or not, nor to any patent
27 owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is
28 withdrawn or modified after publication, and shall be indemnified and held harmless by any party
29 implementing the standard from any and all claims of infringement by a patent owner for such
30 implementations.

31 For information about patents held by third-parties which have notified the DMTF that, in their opinion,
32 such patent may relate to or impact implementations of DMTF standards, visit
33 <http://www.dmtf.org/about/policies/disclosures.php>.

34
35

36

37

38

39

40

41

Contents

43	1	Scope	5
44	2	Overview.....	5
45	3	Redfish Interface Features	5
46	3.1	Introduction	5
47	3.2	Reference Documents	5
48	3.2.1	Redfish.....	5
49	3.2.2	OCP Hardware Management	6
50	4	OCP Redfish Profile	6
51	4.1	Overview	6
52	4.2	Interface	7
53	4.3	Resources.....	7
54	4.4	Collection resources	7
55	4.5	Computer System resource	8
56	4.6	SEL (System Event Log) resource	9
57	4.7	Log Entry resource.....	10
58	4.8	Chassis resource	10
59	4.9	Power resource.....	11
60	4.10	Thermal resource.....	11
61	4.11	BMC resource	14
62	4.12	BMC Ethernet Interface resource	14
63	4.13	BMC Network Protocol resource	15
64	4.14	Account resource	16
65	4.15	Role resource.....	16
66	4.16	Session resource	17
67	5	Open Hardware Management v1.01	17
68	5.1	Mapping	17
69	5.2	Device ID	19
70	5.3	Boot Property Structure	19
71	5.4	Sensors.....	19
72	5.5	Channel Access	21
73	5.6	User Payload Access.....	21
74	5.7	Chassis Capabilities.....	21
75	5.8	Chassis Control.....	22
76		ANNEX A (informative) Change log.....	23
77			

78 Figures

79	Figure 1 – Chassis collection resource	7
80	Figure 2 - ComputerSystem resource	8
81	Figure 3 - SEL resource	9
82	Figure 4 – Log Entry resource.....	10
83	Figure 5 - Chassis resource	10
84	Figure 6 - Power resource.....	11
85	Figure 7 - Thermal resource.....	11
86	Figure 8 - BMC resource	14
87	Figure 9 - BMC Ethernet interface resource	14
88	Figure 10 - BMC network protocol resource	15
89	Figure 11 – Account resource	16
90	Figure 12 – Role resource.....	16

91 Figure 13 – Session resource 17

92 **Tables**

93 Table 1 - Redfish Base Model..... 7

94 Table 2- Redfish Base Model..... 17

95 Table 3 - Need a name here 20

96

97

OCP Hardware Management with Redfish

98 1 Scope

99 This document defines the Redfish model to remotely manage platforms and devices in Open Compute
100 Project.

101 2 Overview

102 Scalability in today's data center is increasingly achieved with horizontal, scale-out solutions, which often
103 include large quantities of simple servers. The usage model of scale-out hardware is drastically different
104 than that of traditional enterprise platforms, and requires a new approach to management.

105 Designed to meet the expectations of end users for simple and secure management of modern scalable
106 platform hardware, DMTF's Redfish® is an open industry standard specification and schema that
107 specifies a RESTful interface and utilizes JSON and OData to help customers integrate solutions within
108 their existing tool chains. An aggressive development schedule is quickly advancing Redfish toward its
109 goal of addressing all the components in the data center with a consistent API.

110 Redfish is composed of an interface specification and resource models, for different management
111 domains. The resource models are specified as schema, in OData CSDL (Common Schema Descriptor
112 Language) and json-schema format. Mockups of the resources have been found to be more easily
113 understood and are present in the document.

114 3 Redfish Interface Features

115 3.1 Introduction

116 This document specifies the Redfish model elements that support OCP Hardware Management. This
117 document describes the Redfish equivalent mechanism to support OCP Hardware Management.

118 The OCP Hardware Management working group has posted multiple document drafts specifying how the
119 OCP Hardware Manage can be supported with IPMI (Intelligent Platform Management Interface).

120 The "Open Hardware Management v1.01" draft describes the use cases and the IPMI-based commands
121 required to support remote machine management. Other IPMI-related documents are in various stages of
122 draft.

123 This document starts with the OCP Hardware Management v1.01 document and specifies the equivalent
124 Redfish mechanism. For the other IPMI-related drafts, this document attempts to specify the equivalent
125 Redfish mechanism, as much as possible.

126 3.2 Reference Documents

127 3.2.1 Redfish

128 The DMTF has various locations for learning and communicating about Redfish

- 129 • Dmtf.org/redfish: Releases of updates to the Redfish Specification and schema
- 130 • Redfish.dmtf.org: Redfish Developer Hub (training, open-source tools, etc.)
- 131 • Redfishforum.com: Public community forum (Redfish, Swordfish, Client SW, Service
132 Implementations, etc.)

133
134 Redfish Interface Specification v1.1 (DSP0266)
135 http://www.dmtf.org/sites/default/files/standards/documents/DSP0266_1.1.0.pdf

136 An interactive explorer of the OCP Redfish Profile discussed in the document is available at
137 <http://redfish.dmtf.org/redfish/v1/mockup/770>

138 **3.2.2 OCP Hardware Management**

139 The OCP Hardware Management specifications can be found on their Wiki.

140 http://www.opencompute.org/wiki/Hardware_Management/SpecsAndDesigns

- 141 • DRAFT - Open Hardware Management v1.01 (Feb 2014)

142 This specification specifies baseline set of commands, which references the DCMI specification.
143 Similarly, the DCMI specification reference commands in the IPMI specification.

- 144 • DRAFT - Requirements for Firmware Update, v0.2b (Aug 2014)

145 This document specifies the firmware update requirements for OCP compliant platforms and devices.

- 146 • DRAFT - OCP Hardware Management Specifications for IPMI

147 This section list four specifications in early draft, which specify the IPMI implementations on any device
148 using IPMI.

- 149 • Hardware Management - ICAP BASE, Version 0.02 (June 2014)
- 150 • Hardware Management - ICAP DRAM, Version 0.03 (June 2014)
- 151 • Hardware Management - ICAP Optical, Version 0.02 (June 2014)
- 152 • Hardware Management - SPEC ID, Version 0.04 (June 2014)

- 153 • Cloud Server Multi Node System Specification v0.7.5 (Aug 2015)

154 This document describes the requirement of a Cloud Server (server, enclosure, rack).

155 **4 OCP Redfish Profile**

156 **4.1 Overview**

157 The profile supports basic management features, as documented in OCP Hardware Management
158 Specification for IPMI v1.01

159 The profile supports a single monolithic server platform.

- 160 • One ComputerSystem
- 161 • One Chassis
- 162 • One Manager

163 The profile supports the following management features.

- 164 • Power-on/off/reset
- 165 • Boot to PXE, HDD, BIOS setup (boot override)
- 166 • 4 temp sensors per DCMI (CPU1, CPU2, Board, Inlet)
- 167 • Simple Power Reading, and DCMI Power Limiting
- 168 • Fan Monitoring w/ redundancy
- 169 • Set asset tag and Indicator LED

- 170 • Basic inventory (serial#, model, SKU, Vendor, BIOS ver...)
- 171 • User Management
- 172 • BMC management: get/set IP, version, enable/disable protocol

173 **4.2 Interface**

174 The management interface shall conform to the Redfish Specification v1.1 (DSP0266).

175 **4.3 Resources**

176 Table 1 specifies the Redfish resources that could be present in the interface. A resource is singleton
 177 resource, unless otherwise specified.

178 The management interface could support the following resources.

179 **Table 1 - Redfish Base Model**

Resource	URI
Service Root	/
Systems collection	/Systems
Computer System	/Systems/{id}
Log Services	/Systems/{id}/LogServices
SEL (System Event Log)	/Systems/{id}/LogServices/SEL
Entry in SEL	/Systems/{id}/LogServices/SEL/Entries/{id}
Chassis collection	/Chassis
Chassis	/Chassis/{id}
Power	/Chassis/{id}/Power
Thermal	/Chassis/{id}/Thermal
Managers collection	/Managers
BMC (baseboard mgmt controller)	/Managers/BMC
BMC's Ethernet interface	/Managers/BMC/EthernetInterfaces/eth0
BMC's network protocol	/Managers/BMC/NetworkProtocol
Account Service	/AccountService
Accounts collection	/AccountService/Accounts
Roles collection	/AccountService/Roles
Session Service	/SessionService
schema	/\$metadata
odata	/odata

180 **4.4 Collection resources**

181 In Redfish, the collection resources have the same JSON format. There is a Members property which
 182 contains a link to the members of the collection. Since this profile is for a single monolithic server
 183 platform, only one member should exist.

184 Figure 1 is an example of the JSON response to a GET request for a Chassis collection resource.

185 **Figure 1 – Chassis collection resource**

```

186 {
187     "@odata.type": "#ChassisCollection.ChassisCollection",
188     "Name": "Chassis Collection",
189     "Members@odata.count": 1,
190     "Members": [
    
```

```

191     {
192         "@odata.id": "/redfish/v1/Chassis/1"
193     }
194 ],
195 "@odata.context": "/redfish/v1/$metadata#ChassisCollection.ChassisCollection",
196 "@odata.id": "/redfish/v1/Chassis",
197 "@Redfish.Copyright": ". . ."
198 }

```

199 4.5 Computer System resource

200 Figure 2 is an example of the JSON response to a GET request for a ComputerSystem resource.

201 This profile could contain each of the properties shown in Figure 2.

202 The System resource supports a 'Reset' action. The reset is performed by sending a POST request to
 203 the path specified in the "target" property. The POST request could include a value for the ResetType
 204 property. The reset type values supported by the implement is contained in the
 205 ResetType@Redfish.AllowableValues property.

206 **Figure 2 - ComputerSystem resource**

```

207 {
208     "@odata.type": "#ComputerSystem.v1_1_0.ComputerSystem",
209     "Id": "1",
210     "Name": "Catfish System",
211     "SystemType": "Physical",
212     "AssetTag": "CATFISHASSETTAG",
213     "Manufacturer": "CatfishManufacturer",
214     "Model": "YellowCat1000",
215     "SerialNumber": "2M220100SL",
216     "SKU": "",
217     "PartNumber": "",
218     "Description": "Catfish Implementation Recipe of simple scale-out monolithic
219 server",
220     "UUID": "00000000-0000-0000-0000-000000000000",
221     "HostName": "catfishHostname",
222     "PowerState": "On",
223     "BiosVersion": "X00.1.2.3.4(build-23)",
224     "Status": {
225         "State": "Enabled",
226         "Health": "OK"
227     },
228     "IndicatorLED": "Off",
229     "Boot": {
230         "BootSourceOverrideEnabled": "Once",
231         "BootSourceOverrideMode": "UEFI",
232         "UefiTargetBootSourceOverride": "uefiDevicePath",
233         "BootSourceOverrideTarget": "Pxe",
234         "BootSourceOverrideTarget@Redfish.AllowableValues": [
235             "None",
236             "Pxe",
237             "Usb",
238             "Hdd",
239             "BiosSetup",

```



```

240         "UefiTarget",
241         "UefiHttp"
242     ]
243 },
244 "LogServices": {
245     "@odata.id": "/redfish/v1/Systems/1/LogServices"
246 },
247 "Links": {
248     "Chassis": [ { "@odata.id": "/redfish/v1/Chassis/1" } ],
249     "ManagedBy": [ { "@odata.id": "/redfish/v1/Managers/bmc" } ]
250 },
251 "Actions": {
252     "#ComputerSystem.Reset": {
253         "target": "/redfish/v1/Systems/1/Actions/ComputerSystem.Reset",
254         "ResetType@Redfish.AllowableValues": [
255             "On",
256             "ForceOff",
257             "GracefulShutdown",
258             "ForceRestart",
259             "Nmi",
260             "GracefulRestart",
261             "ForceOn"
262         ]
263     }
264 },
265 "@odata.context": "/redfish/v1/$metadata#ComputerSystem.ComputerSystem",
266 "@odata.id": "/redfish/v1/Systems/1",
267 "@Redfish.Copyright": "...
268 }

```

269 4.6 SEL (System Event Log) resource

270 Figure 3 is an example of the JSON response to a GET request for the SEL singleton resource.

271 This profile could contain each of the properties shown in Figure 3.

272 Figure 3 - SEL resource

```

273 {
274     "@odata.type": "#LogService.v1_0_2.LogService",
275     "Id": "SEL",
276     "Name": "System Log Service",
277     "MaxNumberOfRecords": 1000,
278     "OverWritePolicy": "WrapsWhenFull",
279     "DateTime": "2015-03-13T04:14:33+06:00",
280     "DateTimeLocalOffset": "+06:00",
281     "ServiceEnabled": true,
282     "Status": {
283         "State": "Enabled",
284         "Health": "OK"
285     },
286     "Actions": {
287         "#LogService.ClearLog": {
288             "target": "/redfish/v1/Systems/1/LogServices/SEL/Actions/LogService.Reset"
289         }
290     },

```

```

291     "Entries": { "@odata.id": "/redfish/v1/Systems/1/LogServices/SEL/Entries" },
292
293     "@odata.context": "/redfish/v1/$metadata#LogService.LogService",
294     "@odata.id": "/redfish/v1/Systems/1/LogServices/SEL",
295     "@Redfish.Copyright": "..."
```

296 4.7 Log Entry resource

297 Figure 4 is an example of the JSON response to a GET request for the SEL event singleton resource.

298 This profile could contain each of the properties shown in Figure 4.

299 Figure 4 – Log Entry resource

```

300 {
301     "@odata.type": "#LogEntry.v1_0_2.LogEntry",
302     "Id": "1",
303     "Name": "Log Entry 1",
304     "EntryType": "SEL",
305     "OemRecordFormat": "CompanyX",
306     "Severity": "Critical",
307     "Created": "2012-03-07T14:44",
308     "EntryCode": "Assert",
309     "SensorType": "Temperature",
310     "SensorNumber": 1,
311     "Message": "Message for Event, Description for SEL, OEM depends",
312     "MessageId": "Event.1.0.TempAssert",
313     "MessageArgs": [ "ArrayOfMessageArgs" ],
314     "Links": {
315         "OriginOfCondition": { "@odata.id": "/redfish/v1/Chassis/1/Thermal" }
316     },
317
318     "@odata.context": "/redfish/v1/$metadata#LogEntry.LogEntry",
319     "@odata.id": "/redfish/v1/Systems/1/LogServices/SEL/Entries/1",
320     "@Redfish.Copyright": "..."
```

322 4.8 Chassis resource

323 Figure 5 is an example of the JSON response to a GET request on a Chassis singleton resource.

324 This profile could contain each of the properties shown in Figure 5.

325 Figure 5 - Chassis resource

```

326 {
327     "@odata.type": "#Chassis.v1_2_0.Chassis",
328     "Id": "1",
329     "Name": "Catfish System Chassis",
330     "ChassisType": "RackMount",
331     "Manufacturer": "CatfishManufacturer",
332     "Model": "YellowCat1000",
333     "SerialNumber": "2M220100SL",
334     "SKU": "",
335     "PartNumber": "",
336     "AssetTag": "CATFISHASSETTAG",
337     "IndicatorLED": "Lit",
338     "PowerState": "On",
339     "Status": {
340         "State": "Enabled",
341         "Health": "OK"
342     },
343     "Thermal": { "@odata.id": "/redfish/v1/Chassis/1/Thermal" },
344     "Power": { "@odata.id": "/redfish/v1/Chassis/1/Power" },
345     "Links": {
346         "ComputerSystems": [ { "@odata.id": "/redfish/v1/Systems/1" } ],
347         "ManagedBy": [ { "@odata.id": "/redfish/v1/Managers/bmc" } ],
348         "ManagersInChassis": [ { "@odata.id": "/redfish/v1/Managers/bmc" } ]
```

```

349     },
350     "@odata.context": "/redfish/v1/$metadata#Chassis.Chassis",
351     "@odata.id": "/redfish/v1/Chassis/1",
352     "@Redfish.Copyright": "...",
353 }

```

354 4.9 Power resource

355 Figure 6 is an example of the JSON response to a GET request on a Power singleton resource.

356 This profile could contain each of the properties shown in Figure 6.

357 Figure 6 - Power resource

```

358 {
359     "@odata.type": "#Power.v1_1_0.Power",
360     "Id": "Power",
361     "Name": "Power",
362     "PowerControl": [
363         {
364             "@odata.id": "/redfish/v1/Chassis/1/Power#/PowerControl/0",
365             "MemberId": "0",
366             "Name": "System Power Control",
367             "PowerConsumedWatts": 224,
368             "PowerCapacityWatts": 600,
369             "PowerLimit": {
370                 "LimitInWatts": 450,
371                 "LimitException": "LogEventOnly",
372                 "CorrectionInMs": 1000
373             },
374             "Status": {
375                 "State": "Enabled",
376                 "Health": "OK"
377             }
378         }
379     ],
380     "@odata.context": "/redfish/v1/$metadata#Power.Power",
381     "@odata.id": "/redfish/v1/Chassis/1/Power",
382     "@Redfish.Copyright": "...",
383 }

```

384 4.10 Thermal resource

385 Figure 7 is an example of the JSON response to a GET request on a Thermal singleton resource.

386 The Temperatures property could contain an entry for:

- 387 • Inlet Temperature
- 388 • Board Temperature
- 389 • CPU Temperature for each of the processors

390 The Fans property could contain an entry for each fan present on the platform. If the fans are redundant, the Redundancy property could be present.

392 The resource could contain each of the remaining properties shown in Figure 7.

393 Figure 7 - Thermal resource

```

394 {
395     "@odata.type": "#Thermal.v1_1_0.Thermal",
396     "Id": "Thermal",
397     "Name": "Thermal",
398     "Temperatures": [
399         {
400             "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Temperatures/0",

```

```

401     "MemberId": "0",
402     "Name": "Inlet Temp",
403     "SensorNumber": 42,
404     "Status": {
405         "State": "Enabled",
406         "Health": "OK"
407     },
408     "ReadingCelsius": 25,
409     "UpperThresholdNonCritical": 35,
410     "UpperThresholdCritical": 40,
411     "UpperThresholdFatal": 50,
412     "MinReadingRange": 0,
413     "MaxReadingRange": 200,
414     "PhysicalContext": "Intake"
415 },
416 {
417     "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Temperatures/1",
418     "MemberId": "1",
419     "Name": "Board Temp",
420     "SensorNumber": 43,
421     "Status": {
422         "State": "Enabled",
423         "Health": "OK"
424     },
425     "ReadingCelsius": 35,
426     "UpperThresholdNonCritical": 30,
427     "UpperThresholdCritical": 40,
428     "UpperThresholdFatal": 50,
429     "MinReadingRange": 0,
430     "MaxReadingRange": 200,
431     "PhysicalContext": "SystemBoard"
432 },
433 {
434     "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Temperatures/2",
435     "MemberId": "2",
436     "Name": "CPU1 Temp",
437     "SensorNumber": 44,
438     "Status": {
439         "State": "Enabled",
440         "Health": "OK"
441     },
442     "ReadingCelsius": 45,
443     "UpperThresholdNonCritical": 60,
444     "UpperThresholdCritical": 82,
445     "MinReadingRange": 0,
446     "MaxReadingRange": 200,
447     "PhysicalContext": "CPU"
448 },
449 {
450     "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Temperatures/3",
451     "MemberId": "3",
452     "Name": "CPU2 Temp",
453     "SensorNumber": 45,
454     "Status": {
455         "State": "Enabled",
456         "Health": "OK"
457     },
458     "ReadingCelsius": 46,
459     "UpperThresholdNonCritical": 60,
460     "UpperThresholdCritical": 82,
461     "MinReadingRange": 0,
462     "MaxReadingRange": 200,
463     "PhysicalContext": "CPU"
464 }
465 ],
466 "Fans": [
467     {
468         "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Fans/0",
469         "MemberId": "0",
470         "Name": "BaseBoard System Fan 1",
471         "PhysicalContext": "Backplane",

```

```

472     "Status": {
473         "State": "Enabled",
474         "Health": "OK"
475     },
476     "Reading": 2100,
477     "ReadingUnits": "RPM",
478     "UpperThresholdNonCritical": 42,
479     "UpperThresholdCritical": 4200,
480     "UpperThresholdFatal": 42,
481     "LowerThresholdNonCritical": 42,
482     "LowerThresholdCritical": 5,
483     "LowerThresholdFatal": 42,
484     "MinReadingRange": 0,
485     "MaxReadingRange": 5000,
486     "Redundancy": [
487         {
488             "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Redundancy/0"
489         }
490     ]
491 },
492 {
493     "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Fans/1",
494     "MemberId": "1",
495     "Name": "BaseBoard System Fan 2",
496     "PhysicalContext": "Backplane",
497     "Status": {
498         "State": "Enabled",
499         "Health": "OK"
500     },
501     "Reading": 2100,
502     "ReadingUnits": "RPM",
503     "UpperThresholdNonCritical": 42,
504     "UpperThresholdCritical": 4200,
505     "UpperThresholdFatal": 42,
506     "LowerThresholdNonCritical": 42,
507     "LowerThresholdCritical": 5,
508     "LowerThresholdFatal": 42,
509     "MinReadingRange": 0,
510     "MaxReadingRange": 5000,
511     "Redundancy": [
512         {
513             "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Redundancy/0"
514         }
515     ]
516 }
517 ],
518 "Redundancy": [
519     {
520         "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Redundancy/0",
521         "MemberId": "0",
522         "Name": "BaseBoard System Fans",
523         "RedundancySet": [
524             {
525                 "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Fans/0"
526             },
527             {
528                 "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Fans/1"
529             }
530         ],
531         "Mode": "N+m",
532         "Status": {
533             "State": "Enabled",
534             "Health": "OK"
535         },
536         "MinNumNeeded": 1,
537         "MaxNumSupported": 2
538     }
539 ],
540 "@odata.context": "/redfish/v1/$metadata#Thermal.Thermal",
541 "@odata.id": "/redfish/v1/Chassis/1/Thermal",

```

```
542     "@Redfish.Copyright": "..."  
543 }
```

544 4.11 BMC resource

545 Figure 8 is an example of the JSON response to a GET request on a BMC resource.

546 The resource could contain each of the properties shown in Figure 8.

547 The resource could support the Reset action.

548 Figure 8 - BMC resource

```
549 {  
550     "@odata.type": "#Manager.v1_1_0.Manager",  
551     "Id": "bmc",  
552     "Name": "Manager",  
553     "ManagerType": "BMC",  
554     "Description": "BMC",  
555     "ServiceEntryPointUUID": "92384634-2938-2342-8820-489239905423",  
556     "UUID": "00000000-0000-0000-0000-000000000000",  
557     "Model": "CatfishBMC",  
558     "DateTime": "2015-03-13T04:14:33+06:00",  
559     "DateTimeLocalOffset": "+06:00",  
560     "Status": {  
561         "State": "Enabled",  
562         "Health": "OK"  
563     },  
564     "FirmwareVersion": "1.00",  
565     "NetworkProtocol": { "@odata.id": "/redfish/v1/Managers/bmc/NetworkProtocol" },  
566     "EthernetInterfaces": { "@odata.id": "/redfish/v1/Managers/bmc/EthernetInterfaces" },  
567     "Links": {  
568         "ManagerForServers": [{ "@odata.id": "/redfish/v1/Systems/1" }],  
569         "ManagerForChassis": [{ "@odata.id": "/redfish/v1/Chassis/1" }],  
570         "ManagerInChassis": { "@odata.id": "/redfish/v1/Chassis/1" }  
571     },  
572     "Actions": {  
573         "#Manager.Reset": {  
574             "target": "/redfish/v1/Managers/bmc/Actions/Manager.Reset",  
575             "ResetType@Redfish.AllowableValues": [  
576                 "ForceRestart",  
577                 "GracefulRestart"  
578             ]  
579         }  
580     },  
581     "@odata.context": "/redfish/v1/$metadata#Manager.Manager",  
582     "@odata.id": "/redfish/v1/Managers/bmc",  
583     "@Redfish.Copyright": "..."  
584 }
```

586 4.12 BMC Ethernet Interface resource

587 Figure 9 is an example of the JSON response to a GET request on a BMC Ethernet interface resource.

588 The resource could contain each of the properties shown in Figure 9.

589 Figure 9 - BMC Ethernet interface resource

```
590 {  
591     "@odata.type": "#EthernetInterface.v1_0_2.EthernetInterface",  
592     "Id": "eth0",  
593     "Name": "Manager Ethernet Interface",  
594     "Description": "Management Network Interface",  
595     "Status": {  
596         "State": "Enabled",  
597         "Health": "OK"  
598     }
```

```

598 },
599 "InterfaceEnabled": true,
600 "PermanentMACAddress": "AA:BB:CC:DD:EE:FF",
601 "MACAddress": "AA:BB:CC:DD:EE:FF",
602 "SpeedMbps": 100,
603 "AutoNeg": true,
604 "FullDuplex": true,
605 "MTUSize": 1500,
606 "HostName": "MyHostName",
607 "FQDN": "MyHostName.MyDomainName.com",
608 "MaxIPv6StaticAddresses": 1,
609 "VLAN": {
610     "VLANEnable": true,
611     "VLANId": 101
612 },
613 "IPv4Addresses": [
614     {
615         "Address": "192.168.0.10",
616         "SubnetMask": "255.255.252.0",
617         "AddressOrigin": "DHCP",
618         "Gateway": "192.168.0.1"
619     }
620 ],
621 "IPv6AddressPolicyTable": [
622     {
623         "Prefix": "::1/128",
624         "Precedence": 50,
625         "Label": 0
626     }
627 ],
628 "IPv6StaticAddresses": [
629     {
630         "Address": "fe80::1ec1:deff:fe6f:1e24",
631         "PrefixLength": 16
632     }
633 ],
634 "IPv6DefaultGateway": "fe80::1ec1:deff:fe6f:1e24",
635 "IPv6Addresses": [
636     {
637         "Address": "fe80::1ec1:deff:fe6f:1e24",
638         "PrefixLength": 64,
639         "AddressOrigin": "SLAAC",
640         "AddressState": "Preferred"
641     }
642 ],
643
644 "@odata.context": "/redfish/v1/$metadata#EthernetInterface.EthernetInterface",
645 "@odata.id": "/redfish/v1/Managers/bmc/EthernetInterfaces/eth0",
646 "@Redfish.Copyright": "..."

```

647 4.13 BMC Network Protocol resource

648 Figure 10 is an example of the JSON response to a GET request on a BMC network protocol resource.

649 The resource could contain each of the properties shown in Figure 10.

650 **Figure 10 - BMC network protocol resource**

```

651 {
652     "@odata.type": "#ManagerNetworkProtocol.v1_0_2.ManagerNetworkProtocol",
653     "Id": "NetworkProtocol",
654     "Name": "Manager Network Protocol",
655     "Description": "Manager Network Service Status",
656     "Status": {
657         "State": "Enabled",
658         "Health": "OK"
659     },
660     "HostName": "myBmcHostname",
661     "FQDN": "mymanager.mydomain.com",

```

```

662 "HTTP": { "ProtocolEnabled": true, "Port": 80 },
663 "HTTPS": { "ProtocolEnabled": true, "Port": 443 },
664 "IPMI": { "ProtocolEnabled": true, "Port": 623 },
665 "SSH": { "ProtocolEnabled": true, "Port": 22 },
666 "SNMP": { "ProtocolEnabled": true, "Port": 161 },
667 "Telnet": { "ProtocolEnabled": true, "Port": 23 },
668 "SSDP": {
669     "ProtocolEnabled": true,
670     "Port": 1900,
671     "NotifyMulticastIntervalSeconds": 600,
672     "NotifyTTL": 5,
673     "NotifyIPv6Scope": "Site"
674 },
675
676 "@odata.context": "/redfish/v1/$metadata#ManagerNetworkProtocol.ManagerNetworkProtocol",
677 "@odata.id": "/redfish/v1/Managers/bmc/NetworkProtocol",
678 "@Redfish.Copyright": "...
679 }

```

680 4.14 Account resource

681 Figure 11 is an example of the JSON response to a GET request on an Account resource.

682 The resource could contain each of the properties shown in Figure 11.

683 Figure 11 – Account resource

```

684 {
685     "@odata.type": "#ManagerAccount.v1_0_2.ManagerAccount",
686     "Id": "root",
687     "Name": "UserAccount",
688     "Description": "User Account",
689     "Enabled": true,
690     "Password": null,
691     "UserName": "root",
692     "RoleId": "Administrator",
693     "Locked": false,
694     "Links": {
695         "Role": { "@odata.id": "/redfish/v1/AccountService/Roles/Admin" }
696     },
697     "@odata.context": "/redfish/v1/$metadata#ManagerAccount.ManagerAccount",
698     "@odata.id": "/redfish/v1/AccountService/Accounts/root",
699     "@Redfish.Copyright": "...
700 }

```

701 4.15 Role resource

702 Figure 12 is an example of the JSON response to a GET request on a Role resource.

703 The resource could contain each of the properties shown in Figure 12.

704 Figure 12 – Role resource

```

705 {
706     "@odata.type": "#Role.v1_0_2.Role",
707     "Id": "Administrator",
708     "Name": "User Role",
709     "Description": "Admin User Role",
710     "IsPredefined": true,
711     "AssignedPrivileges": [
712         "Login",
713         "ConfigureManager",
714         "ConfigureUsers",
715         "ConfigureSelf",
716         "ConfigureComponents"
717     ],
718     "@odata.context": "/redfish/v1/$metadata#Role.Role",
719     "@odata.id": "/redfish/v1/AccountService/Roles/Admin",

```



```
720     "@Redfish.Copyright": "..."  
721 }
```

722 **4.16 Session resource**

723 Figure 13 is an example of the JSON response to a GET request on a Session resource.

724 The resource could contain each of the properties shown in Figure 13.

725 **Figure 13 – Session resource**

```
726 {  
727     "@odata.type": "#Session.v1_0_2.Session",  
728     "Id": "1234567890ABCDEF",  
729     "Name": "User Session",  
730     "Description": "Manager User Session",  
731     "UserName": "root",  
732  
733     "@odata.context": "/redfish/v1/$metadata#Session.Session",  
734     "@odata.id": "/redfish/v1/SessionService/Sessions/1234567890ABCDEF",  
735     "@Redfish.Copyright": "..."  
736 }
```

737 **5 Open Hardware Management v1.01**

738 **5.1 Mapping**

739 The mapping below is organized according to the IPMI-based documents. The IPMI column contains the
740 referenced IPMI-based command.

741 The Redfish equivalent column contains the mechanism that would be used to perform the same task.

742 In some cases, the task can be described simply. In those cases, the column contains the HTTP request,
743 the generalized path and the action that the client would perform on the JSON included in the HTTP
744 response.

745 In this description text, whether the resource and/or property currently exists in the Redfish model is
746 indicated by the color of the text:

- 747 • Black text – resource or property exists in the Redfish model
- 748 • Blue text – resource or property does not exist in the Redfish model. The text is a proposal.

749 Table 2 contains the Redfish model elements to support hardware management. Redfish supports a
750 collection of Managers and each manager is a singleton resource (./Managers/{id}). For a platform with
751 only one manager, a BMC (baseboard management controller), then {id} = BMC.

752 **Table 2- Redfish Base Model**

IPMI Command	Redfish Equivalent	In OCP mockup
Get DCMI Capabilities Info	GET ./Profiles.... (is this needed)???	
Set & Get DCMI Configuration Parameters	TBD	N
Get Management Controller Identifier String	GET ./Managers/{id} Extract value of Name property	Y
Set Management Controller Identifier	PATCH ./Managers/{id}	Y

IPMI Command	Redfish Equivalent	In OCP mockup
String	Request contains value of Name property	
Get Asset Tag	GET ./Systems/{id} Extract AssetTag	Y
Set Asset Tag	PATCH ./Systems/{id} Request contains AssetTag value	Y
Get Device ID	Not Supported. See section 5.2	N
Get System GUID	GET ./Systems/{id}/UUID	Y
Set & Get System Boot Options	Supported. See section 5.3	Y
Get DCMI Sensor Info	GET ./TelemetryService/MetricDefinitions/{id}, where ID is the name of the sensor. Supported. See section 5.4	Y
Get Sensor Reading	GET <path to resource containing the sensor reading> Extract the value of the sensor reading See section 5.4	Y
Get SEL info	GET ./Systems/{id}/LogServices/SEL	Y
Get SEL Entry	GET ./Systems/{id}/LogServices/SEL/Entries/{id}	Y
Clear SEL	POST ./Managers/{id}/LogServices/SEL/Actions/LogService.Reset	Y
Get Power Reading	GET ./Chassis/{id}/Power Extract PowerControl/PowerConsumedWatts	Y
Get Temperature Readings	GET ./Chassis/{id}/Thermal Extract Temperatures	Y
Get LAN Configuration Parameters	GET ./Managers/{id}/EthernetInterfaces/{id}	Y
Set LAN Configuration Parameters	POST ./Managers/{id}/EthernetInterfaces/{id}/SD	Y
Set & Get Channel Access	Not supported. See section 5.5	N
Get User Access	GET ./AccountService/Accounts/{id} Extract the RoleId property	Y
Set User Access	PATCH ./AccountService/Accounts/{id} Request contains value of RoleId property	Y
Get User Name	GET ./AccountService/Accounts/{id} Extract the UserName property	Y
Set User Name	GET ./AccountService/Accounts/{id} Request contains value of UserName property	Y
Set User Password	GET ./AccountService/{id} Request contains the value of the Password property	Y
Set & Get User Payload Access	Not supported. See section 5.6	N
Get Chassis Capabilities	Not supported. See section 5.7	N
Get Chassis Status	GET ./Chassis/{id}/State/Status	Y
Chassis Control	POST ./Systems/1/Actions/ComputerSystem.Reset Supported. See section 5.8	Y

IPMI Command	Redfish Equivalent	In OCP mockup
Chassis Identify	GET ./Chassis/{id}#/AssetTag	Y
Get ACPI Power State	GET ./Chassis/{id}#/ACPIPowerState	N

753 **5.2 Device ID**

754 IPMI supports a device ID to uniquely specify each device.

755 Redfish does not support the notion of Device ID.

756 **5.3 Boot Property Structure**

757 The System resource contains a boot property contains properties to control the booting of the system.
 758 The BootSourceOverrideTarget is a Redfish annotation which may be present to provide the client with
 759 the values that the implementation supports.

```

760 "Boot": {
761   "BootSourceOverrideEnabled": "Once",
762   "BootSourceOverrideMode": "UEFI",
763   "BootSourceOverrideTarget": "Pxe",
764   "UefiTargetBootSourceOverride": "uefi device path"
765   "BootSourceOverrideTarget@Redfish.AllowableValues": [
766     "None",
767     "Pxe",
768     "Floppy",
769     "Cd",
770     "Usb",
771     "Hdd",
772     "BiosSetup",
773     "Utilities",
774     "Diags",
775     "UefiTarget",
776     "SDCard",
777     "UefiHttp"
778   ]
779 }
```

780 **5.4 Sensors**

781 Regarding sensors, the OCP Specification refers to the DCMI. The DCMI spec specifies three sensors:

- 782 • Inlet Temperature
- 783 • Baseboard Temperature
- 784 • CPU (Processor) Temperature

785 In Redfish, these temperature metrics are contained in the Thermal resource. Within the resource, the
 786 Temperatures JSON object contains the temperature metrics associated with the Chassis named "Ch_1".
 787 Each temperature is identified by a "Name" property.

788 The following values of Name could be supported:

- 789 • "Inlet Temp", in which PhysicalContext="Intake"
- 790 • "Board Temp", in which PhysicalContext="SystemBoard"
- 791 • "CPU{n} Temp", where "{n}" is a unique integer value, in which PhysicalContext="CPU"

```

792 "@odata.id": "/redfish/v1/Chassis/Ch_1/Thermal",
793
794 "Temperatures": [
795   {
```

```

796     "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Temperatures/0",
797     "MemberId": "0",
798     "Name": "Inlet Temp",
799     "Status": { "State": "Enabled", "Health": "OK" },
800     "ReadingCelsius": 25,
801     "PhysicalContext": "Intake",
802     . . .
803   },
804   {
805     "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Temperatures/1",
806     "MemberId": "1",
807     "Name": "Board Temp",
808     "Status": { "State": "Enabled", "Health": "OK" },
809     "ReadingCelsius": 35,
810     "PhysicalContext": "SystemBoard"
811     . . .
812   },
813   {
814     "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Temperatures/2",
815     "MemberId": "2",
816     "Name": "CPU1 Temp",
817     "Status": { "State": "Enabled", "Health": "OK" },
818     "ReadingCelsius": 45,
819     "PhysicalContext": "CPU",
820     . . .
821   },
822   {
823     "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Temperatures/3",
824     "MemberId": "3",
825     "Name": "CPU2 Temp",
826     "Status": { "State": "Enabled", "Health": "OK" },
827     "ReadingCelsius": 46,
828     "PhysicalContext": "CPU",
829     . . .
830   }

```

831

832

Table 3 - Need a name here

DCMI Command	Redfish Equivalent	In OCP Profile
Get Inlet Temperature	GET ./Chassis/{id}/Thermal In the Temperature array property, find the Temperature, whose Name property is "Inlet Temp". From the Temperature so found, extract the value of ReadingCelsius	Y
Get Baseboard Temperature	GET ./Chassis/{id}/Thermal In the Temperature array property, find the Temperature, whose Name property is "Board Temp". From the Temperature so found, extract the value of ReadingCelsius	Y
Get CPU (Processor) Temperature	GET ./Chassis/{id}/Thermal In the Temperature array property, find the Temperature, whose Name property is "CPU{n} Temp", where "{n}" is a unique integer value From the Temperature so found, extract the value of ReadingCelsius	Y

833 The sensor section of DCMI specification also lists three event sensors.

- 834 • A power off event
- 835 • A power threshold event
- 836 • A thermal limit event

837 In Redfish, these power metrics are contained in the Power resource. Within the resource, the
 838 PowerControl JSON object contains the properties associated with the Chassis named "Ch_1". Each
 839 power control is identified by a "Name" property.

840 The following values of Name could be supported:

- 841 • "System Power Control", in which PhysicalContext="SystemBoard"

```

842 "@odata.id": "/redfish/v1/Chassis/Ch_1/Power",
843 "PowerControl": [
844   {
845     "@odata.id": "/redfish/v1/Chassis/1/Power#/PowerControl/0",
846     "MemberId": "0",
847     "Name": "System Power Control",
848     "PowerConsumedWatts": 224,
849     "PowerCapacityWatts": 600,
850     "PowerLimit": {
851       "LimitInWatts": 450,
852       "LimitException": "LogEventOnly",
853       "CorrectionInMs": 1000
854     },
855     "Status": { "State": "Enabled", "Health": "OK" },
856   }
857 ]
  
```

858 Redfish supports a subscribe/publish event model.

- 859 • For power off events
- 860 • The client would subscribes to a lifecycle events (EventType=" StatusChange")
- 861 • Upon receiving an event, inspect the Event Message and determine whether the MessageID property
- 862 has the value of TBD
- 863 • For power threshold events, the client would TBD
- 864 • For thermal limit events, the client would TBD

865 5.5 Channel Access

866 IPMI supports a channel mechanism for routing messages to IPMB, and routing messages to different
 867 platform internal media. Channels are a specific capability of IPMI.

868 Redfish does not support channels.

869 5.6 User Payload Access

870 IPMI supports a payload mechanism to carry data besides IPMI payloads and Serial-over-LAN payloads
 871 over the IPMI protocol.

872 Redfish specifies HTTP/HTTPS. Redfish does not specify a special payload mechanisms. For Serial-
 873 over-LAN, the SSH or Telnet protocol could be supported.

874 5.7 Chassis Capabilities

875 IPMI supports chassis capabilities to return information about the chassis management which are present
 876 on the IPMB and how to access those functions. IPMB is a specific capability of IPMI.

877 Redfish does not support chassis capabilities.

878 5.8 Chassis Control

879 Redfish defines Chassis as the 'sheet metal' container with power and cooling domain. The generic
880 definition of Chassis allows chassis to define any container.

881 Redfish encapsulates the other state actions besides power state actions in the ComputerSystem
882 resource. The ComputerSystem resource defines the Action property.

883 The fragment defines an action names ComputerSystem.Reset, which the Redfish client can perform by
884 POST'ing to the URI specified in the "target" property. The POST request should include a JSON file with
885 the parameter(s) specified in the ResetActionInfo resource.

```
886     "Actions": {  
887         "#ComputerSystem.Reset": {  
888             "target": "/redfish/v1/Systems/1/Actions/ComputerSystem.Reset",  
889             "@Redfish.ActionInfo": "/redfish/v1/Systems/1/ResetActionInfo"  
890         }  
891     }
```

892 Below is the contents of the ResetActionInfo resource. There is only one parameter specified which is
893 required. The property name is "ResetType" and its value is a string. The structure also shows that
894 allowable values that can be used as values.

```
895 {  
896     "@Redfish.Copyright": "...",  
897     "@odata.context": "/redfish/v1/$metadata#ActionInfo.ActionInfo",  
898     "@odata.id": "/redfish/v1/Systems/1/ResetActionInfo",  
899     "@odata.type": "#ActionInfo.v1_0_0.ActionInfo",  
900     "Parameters": [{  
901         "Name": "ResetType",  
902         "Required": true,  
903         "DataType": "String",  
904         "AllowableValues": [  
905             "On",  
906             "ForceOff",  
907             "GracefulShutdown",  
908             "GracefulRestart",  
909             "ForceRestart",  
910             "Nmi",  
911             "ForceOn",  
912             "PushPowerButton"  
913         ]  
914     }]  
915 }
```

916

917
918
919
920
921

ANNEX A
(informative)

Change log

Version	Date	Description
0.2.2b	2017-05-30	Work in Progress

922