



1
2
3
4
5

Document Number: DSP0243

Date: 2012-09-20

Version: 2.0.0

Work Group Version: 0.9.9

6 **Open Virtualization Format Specification**

7 **Information for work in progress version:**

8 "IMPORTANT: This specification is not a standard. It does not necessarily reflect the views of the DMTF
9 or all of its members. Because this document is a Work in Progress, this specification may still change,
10 perhaps profoundly. This document is available for public review and comment until the stated expiration
11 date."

12 It expires on: **2012-12-15**

13 Target version for final status 2.0.0

14 Provide any comments through the DMTF Feedback Portal: <http://www.dmtf.org/standards/feedback/>

15 **Document Type: Specification**

16 **Document Status: Work in Progress**

17 **Document Language: E**

18 Copyright notice

19 Copyright © 2012 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

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

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34 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

CONTENTS

41	Foreword.....	6
42	Introduction	8
43	1 Scope.....	9
44	2 Normative References.....	9
45	3 Terms and Definitions	10
46	4 Symbols and Abbreviated Terms	12
47	5 OVF Packages.....	12
48	5.1 OVF Package Structure	12
49	5.2 Virtual Disk Formats.....	14
50	5.3 Distribution as a Single File.....	14
51	5.4 Distribution as a Set of Files	15
52	6 OVF Descriptor	15
53	7 Envelope Element	16
54	7.1 File References.....	17
55	7.2 Content Element	18
56	7.3 Extensibility	19
57	7.4 Conformance	20
58	8 Virtual Hardware Description.....	20
59	8.1 VirtualHardwareSection	20
60	8.2 Extensibility	23
61	8.3 Virtual Hardware Elements	23
62	8.4 Ranges on Elements.....	25
63	9 Core Metadata Sections in version 2.....	27
64	9.1 DiskSection	28
65	9.2 NetworkSection.....	29
66	9.3 ResourceAllocationSection	30
67	9.4 AnnotationSection.....	30
68	9.5 ProductSection.....	31
69	9.6 EulaSection.....	34
70	9.7 StartupSection	34
71	9.8 DeploymentOptionSection	35
72	9.9 OperatingSystemSection	37
73	9.10 InstallSection.....	38
74	9.11 EnvironmentFilesSection	38
75	9.12 BootDeviceSection.....	39
76	9.13 SharedDiskSection	40
77	9.14 ScaleOutSection	41
78	9.15 PlacementGroupSection and PlacementSection	42
79	9.16 Encryption Section	44
80	10 Internationalization	46
81	10.1 Internal Resource Bundles.....	46
82	10.2 External Resource Bundles.....	47
83	10.3 Message Content in External File	47
84	11 OVF Environment.....	48
85	11.1 Environment Document.....	48
86	11.2 Transport.....	49
87	ANNEX A (informative) Symbols and Conventions.....	51
88	ANNEX B (normative) OVF XSD	52
89	ANNEX C (informative) OVF Mime Type Registration Template	53
90	ANNEX D (informative) Network Port Profile Examples.....	55

91 D.1 Example 1 (OVF Descriptor for One Virtual System and One Network with an Inlined
 92 Network Port Profile)..... 55
 93 D.2 Example 2 (OVF Descriptor for One Virtual System and One Network with a Locally
 94 Referenced Network Port Profile) 57
 95 D.3 Example 3 (OVF Descriptor for One Virtual System and One Network with a Network Port
 96 Profile referenced by a URI)..... 58
 97 D.4 Example 4 (OVF Descriptor for Two Virtual Systems and One Network with Two Network
 98 Port Profiles referenced by URIs) 60
 99 D.5 Example 5 (networkportprofile1.xml)..... 63
 100 D.6 Example 6 (networkportprofile2.xml)..... 63
 101 ANNEX E (informative) Bibliography..... 64
 102 ANNEX F (informative) Change Log 65
 103

104 **Tables**

105 Table 1 – XML Namespace Prefixes..... 16

106 Table 2 – Actions for Child Elements with `ovf:required` Attribute..... 23

107 Table 3 – HostResource Element 24

108 Table 4 – Elements for Virtual Devices and Controllers..... 25

109 Table 5 – Core Metadata Sections in version 2 27

110 Table 6 – Property Types..... 33

111 Table 7 – Property Qualifiers 34

112 Table 8 – Core Sections..... 49

113

114

Foreword

115 The *Open Virtualization Format Specification* (DSP0243) was prepared by the System Virtualization,
116 Partitioning, and Clustering Working Group of the DMTF.

117 This specification has been developed as a result of joint work with many individuals and teams,
118 including:

119

120	Lawrence Lamers	VMware Inc. (Chair)
121	Hemal Shah	Broadcom Corporation (co-Editor)
122	Steffen Grarup	VMware Inc. (co-Editor)
123		
124	Vincent Kowalski	BMC Software
125	Hemal Shah	Broadcom Corporation
126	John Crandall	Brocade Communications Systems
127	Marvin Waschke	CA Technologies
128	Naveen Joy	Cisco
129	Steven Neely	Cisco
130	Shishir Pardikar	Citrix Systems Inc.
131	Thomas Root	Citrix Systems Inc.
132	Richard Landau	DMTF Fellow
133	Jacques Durand	Fujitsu
134	Derek Coleman	Hewlett-Packard Company
135	Robert Freund	Hitachi, Ltd.
136	Fred Maciel	Hitachi, Ltd.
137	Eric Wells	Hitachi, Ltd.
138	Abdellatif Touimi	Huawei
139	Jeff Wheeler	Huawei
140	HengLiang Zhang	Huawei
141	Oliver Benke	IBM
142	Ron Doyle	IBM
143	Michael Gering	IBM
144	Michael Johanssen	IBM
145	Andreas Maier	IBM
146	Marc-Arthur Pierre-Louis	IBM
147	John Leung	Intel Corporation
148	Nitin Bhat	Microsoft Corporation
149	Maurizio Carta	Microsoft Corporation
150	Monica Martin	Microsoft Corporation
151	John Parchem	Microsoft Corporation
152	Ed Reed	Microsoft Corporation
153	Nihar Shah	Microsoft Corporation
154	Cheng Wei	Microsoft Corporation
155	Narayan Venkat	NetApp
156	Tatyana Bagerman	Oracle
157	Srinivas Maturi	Oracle
158	Dr. Fermín Galán Márquez	Telefónica
159	Miguel Ángel Peñalvo	Telefónica
160	Dr. Fernando de la Iglesia	Telefónica
161	Álvaro Polo	Telefónica
162	Steffen Grarup	VMware Inc.
163	Lawrence Lamers	VMware Inc.
164	Rene Schmidt	VMware Inc.
165	Paul Ferdinand	WBEM Solutions

166	Junsheng Chu	ZTE Corporation
167	Bhumip Khasnabish	ZTE Corporation
168	Ghazanfar Ali	ZTE Corporation

169

Introduction

170 The *Open Virtualization Format (OVF) Specification* describes an open, secure, portable, efficient and
171 extensible format for the packaging and distribution of software to be run in virtual machines. The key
172 properties of the format are as follows:

173 • **Optimized for distribution**

174 OVF supports content verification and integrity checking based on industry-standard public key
175 infrastructure, and it provides a basic scheme for management of software licensing.

176 • **Optimized for a simple, automated user experience**

177 OVF supports validation of the entire package and each virtual machine or metadata
178 component of the OVF during the installation phases of the virtual machine (VM) lifecycle
179 management process. It also packages with the package relevant user-readable descriptive
180 information that a virtualization platform can use to streamline the installation experience.

181 • **Supports both single VM and multiple-VM configurations**

182 OVF supports both standard single VM packages and packages containing complex, multi-tier
183 services consisting of multiple interdependent VMs.

184 • **Portable VM packaging**

185 OVF is virtualization platform neutral, while also enabling platform-specific enhancements to be
186 captured. It supports the full range of virtual hard disk formats used for hypervisors today, and it
187 is extensible, which allow it to accommodate formats that may arise in the future. Virtual
188 machine properties are captured concisely and accurately.

189 • **Vendor and platform independent**

190 OVF does not rely on the use of a specific host platform, virtualization platform, or guest
191 operating system.

192 • **Extensible**

193 OVF is immediately useful — and extensible. It is designed to be extended as the industry
194 moves forward with virtual appliance technology. It also supports and permits the encoding of
195 vendor-specific metadata to support specific vertical markets.

196 • **Localizable**

197 OVF supports user-visible descriptions in multiple locales, and it supports localization of the
198 interactive processes during installation of an appliance. This capability allows a single
199 packaged appliance to serve multiple market opportunities.

200 • **Open standard**

201 OVF has arisen from the collaboration of key vendors in the industry, and it is developed in an
202 accepted industry forum as a future standard for portable virtual machines.

203 It is not an explicit goal for OVF to be an efficient execution format. A hypervisor is allowed but not
204 required to run software in virtual machines directly out of the Open Virtualization Format.

205

Open Virtualization Format Specification

206 1 Scope

207 The *Open Virtualization Format (OVF) Specification* describes an open, secure, portable, efficient and
208 extensible format for the packaging and distribution of software to be run in virtual machines.

209 This version of the specification (2.0) is intended to allow OVF 1.x tools to work with OVF 2.0 descriptors
210 in the following sense:

211

- 212 • Existing OVF 1.x tools should be able to parse OVF 2.0 descriptors.
- 213 • Existing OVF 1.x tools should be able to give warnings/errors if dependencies to 2.0 features are
214 required for correct operation.

215 2 Normative References

216 The following referenced documents are indispensable for the application of this document. For dated
217 references, only the edition cited applies. For undated references, the latest edition of the referenced
218 document (including any amendments) applies.

219 [ISO/IEC/IEEE 9945:2009](#): Information technology -- Portable Operating System Interface (POSIX®) Base
220 Specifications, Issue 7
221 http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50516

222 DMTF DSP0004, *Common Information Model (CIM) Infrastructure Specification 2.7*,
223 http://www.dmtf.org/standards/published_documents/DSP0004_2.7.pdf

224 DMTF DSP0230, *WS-CIM Mapping Specification 1.0*,
225 http://www.dmtf.org/standards/published_documents/DSP0230_1.0.pdf

226 DMTF DSP1041, *Resource Allocation Profile (RAP) 1.1*,
227 http://www.dmtf.org/standards/published_documents/DSP1041_1.1.pdf

228 DMTF DSP1043, *Allocation Capabilities Profile (ACP) 1.0*,
229 http://www.dmtf.org/standards/published_documents/DSP1043_1.0.pdf

230 DMTF DSP8023, *Open Virtualization Format (OVF) 2 XML Schema*,
231 <http://schemas.dmtf.org/ovf/envelope/2/dsp8023.xsd>

232 DMTF DSP8049, *Network Port Profile XML Schema*,
233 <http://schemas.dmtf.org/ovf/networkportprofile/1/dsp8049.xsd>

234

235 IETF RFC1738, T. Berners-Lee, *Uniform Resource Locators (URL)*, December 1994,
236 <http://tools.ietf.org/html/rfc1738>

237 IETF RFC1952, P. Deutsch, *GZIP file format specification version 4.3*, May 1996,
238 <http://tools.ietf.org/html/rfc1952>

239 IETF Standard 68, *Augmented BNF for Syntax Specifications: ABNF*,
240 <http://tools.ietf.org/html/rfc5234>

- 241 IETF RFC2616, R. Fielding et al, *Hypertext Transfer Protocol – HTTP/1.1*, June 1999,
242 <http://tools.ietf.org/html/rfc2616>
- 243 IETF Standard 66, *Uniform Resource Identifiers (URI): Generic Syntax*,
244 <http://tools.ietf.org/html/rfc3986>
- 245 ISO 9660, 1988 Information processing-Volume and file structure of CD-ROM for information interchange,
246 http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=17505
- 247 ISO, ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
248 <http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype>
- 249 W3C, *XML Schema Part 1: Structures Second Edition*, 28 October 2004. W3C Recommendation. URL:
250 <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/>
- 251 W3C, *XML Schema Part 2: Datatypes Second Edition*, 28 October 2004. W3C Recommendation. URL:
252 <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>
- 253 XML Encryption Syntax and Processing Version 1.1, 13 March 2012, W3C Candidate Recommendation
254 <http://www.w3.org/TR/2012/CR-xmlenc-core1-20120313/>
- 255 FIPS 180-2: Secure Hash Standard (SHS)
256 <http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf>

257 3 Terms and Definitions

258 For the purposes of this document, the following terms and definitions apply.

259 3.1

260 **can**

261 used for statements of possibility and capability, whether material, physical, or causal

262 3.2

263 **cannot**

264 used for statements of possibility and capability, whether material, physical, or causal

265 3.3

266 **conditional**

267 indicates requirements to be followed strictly to conform to the document when the specified conditions
268 are met

269 3.4

270 **mandatory**

271 indicates requirements to be followed strictly to conform to the document and from which no deviation is
272 permitted

273 3.5

274 **may**

275 indicates a course of action permissible within the limits of the document

276 3.6

277 **need not**

278 indicates a course of action permissible within the limits of the document

279 3.7

- 280 **optional**
281 indicates a course of action permissible within the limits of the document
- 282 **3.8**
283 **shall**
284 indicates requirements to be followed strictly to conform to the document and from which no deviation is
285 permitted
- 286 **3.9**
287 **shall not**
288 indicates requirements to be followed strictly to conform to the document and from which no deviation is
289 permitted
- 290 **3.10**
291 **should**
292 indicates that among several possibilities, one is recommended as particularly suitable, without
293 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 294 **3.11**
295 **should not**
296 indicates that a certain possibility or course of action is deprecated but not prohibited
- 297 **3.12**
298 **appliance**
299 see [virtual appliance](#)
- 300 **3.13**
301 **deployment platform**
302 the product that installs an OVF package
- 303 **3.14**
304 **guest software**
305 the software that runs inside a virtual machine
306 The guest is typically an operating system and some user-level applications and services.
- 307 **3.15**
308 **OVF package**
309 OVF XML descriptor file accompanied by zero or more files
- 310 **3.16**
311 **OVF descriptor**
312 OVF XML descriptor file
- 313 **3.17**
314 **platform**
315 see [deployment platform](#)
- 316 **3.18**
317 **virtual appliance**
318 a service delivered as a complete software stack installed on one or more virtual machines
319 A virtual appliance is typically expected to be delivered in an OVF package.

320 **3.19**

321 **virtual hardware**

322 the processor, memory and I/O resources of a virtual computer system

323 **3.20**

324 **virtual machine**

325 as defined in System Virtualization Profile

326 **3.21**

327 **virtual machine collection**

328 a collection comprised of a set of virtual machines. This service component can be a:

329 - simple set of one or more virtual machines, or

330 - a complex service component built out of a combination of virtual machines and other virtual
331 machine collections that enables nested complex service components.

332 **4 Symbols and Abbreviated Terms**

333 The following symbols and abbreviations are used in this document.

334 **4.1.1**

335 **CIM**

336 Common Information Model

337 **4.1.2**

338 **IP**

339 Internet Protocol

340 **4.1.3**

341 **OVF**

342 Open Virtualization Format

343 **4.1.4**

344 **VM**

345 Virtual Machine

346 **5 OVF Packages**

347 **5.1 OVF Package Structure**

348 An OVF package shall consist of the following files:

- 349 • one OVF descriptor with extension `.ovf`
- 350 • zero or one OVF manifest with extension `.mf`
- 351 • zero or one OVF certificate with extension `.cert`
- 352 • zero or more disk image files
- 353 • zero or more additional resource files, such as ISO images

354 The file extensions `.ovf`, `.mf` and `.cert` shall be used.

355 EXAMPLE 1: The following list of files is an example of an OVF package:

```
356 package.ovf
357 package.mf
358 de-DE-resources.xml
359 vmdisk1.vmdk
360 vmdisk2.vhd
361 resource.iso
```

362 An OVF package can be stored as either a single unit or a set of files, as described in 5.3 and 5.4. Both
363 modes shall be supported.

364 An OVF package may have a manifest file containing the SHA digests of individual files in the package.
365 OVF packages authored according to this version of the specification shall use SHA256 digests; older
366 OVF packages are allowed to use SHA1. The manifest file shall have an extension `.mf` and the same
367 base name as the `.ovf` file and be a sibling of the `.ovf` file. If the manifest file is present, a consumer of
368 the OVF package shall verify the digests by computing the actual SHA digests and comparing them with
369 the digests listed in the `References` element of the OVF descriptor, see clause 7.1, and for no other files.
370

371 The syntax definitions below use ABNF with the exceptions listed in ANNEX A.

372 The format of the manifest file is as follows:

```
373 manifest_file = *( file_digest )
374 file_digest  = algorithm "(" file_name ")" "=" sp digest nl
375 algorithm    = "SHA1" | "SHA256"
376 digest      = *( hex-digit )
377 hex-digit   = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "a" |
378 "b" | "c" | "d" | "e" | "f"
379 sp          = %x20
380 nl         = %x0A
```

381 EXAMPLE 2: The following example show the partial contents of a manifest file:

```
382 SHA256(package.ovf)= 9902cc5ec4f4a00cabbbf7b60d039263587ab430d5fdbc5cd5e8707391c90a4
383 SHA256(vmdisk.vmdk)= aab66c4d70e17cec2236a651a3fc618cafc5ec6424122904dc0b9c286fce40c2
```

384 An OVF package may be signed by signing the manifest file. The digest of the manifest file is stored in a
385 certificate file with extension `.cert` file along with the base64-encoded X.509 certificate. The `.cert` file
386 shall have the same base name as the `.ovf` file and be a sibling of the `.ovf` file. A consumer of the OVF
387 package shall verify the signature and should validate the certificate. The format of the certificate file shall
388 be as follows:

```
389 certificate_file = manifest_digest certificate_part
390 manifest_digest = algorithm "(" file_name ")" "=" sp signed_digest nl
391 algorithm       = "SHA1" | "SHA256"
392 signed_digest   = *( hex-digit )
393 certificate_part = certificate_header certificate_body certificate_footer
394 certificate_header = "-----BEGIN CERTIFICATE-----" nl
395 certificate_footer = "-----END CERTIFICATE-----" nl
396 certificate_body  = base64-encoded-certificate nl
397                  ; base64-encoded-certificate is a base64-encoded X.509
398                  ; certificate, which may be split across multiple lines
399 hex-digit       = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "a"
400 | "b" | "c" | "d" | "e" | "f"
401 sp              = %x20
402 nl              = %x0A
```

403 EXAMPLE 3: The following list of files is an example of a signed OVF package:

```
404 package.ovf
405 package.mf
406 package.cert
407 de-DE-resources.xml
408 vmdisk1.vmdk
409 vmdisk2.vmdk
410 resource.iso
```

411 EXAMPLE 4: The following example shows the contents of a sample OVF certification file, where the SHA1 digest
412 of the manifest file has been signed with a 512 bit key:

```
413 SHA1(package.mf) = 7f4b8efb8fe20c06df1db68281a63f1b088e19dbf00e5af9db5e8e3e319de
414 7019db88a3bc699bab6ccd9e09171e21e88ee20b5255cec3fc28350613b2c529089
415 -----BEGIN CERTIFICATE-----
416 MIIBGjCCASwCAQQwDQYJKoZIhvcNAQEEBQAwoDELMakGA1UEBhMCQVUxDDAKBgNV
417 BAGTA1FMRDEbMBkGA1UEAxMSU1NMZWV5L3JzYSB0IENBMB4XDThkMTAwOTIz
418 MzIwNVowXDTk4MDcwNTIzMzIwNVowYDELMAkGA1UEBhMCQVUxDDAKBgNVBAGTA1FM
419 RDEZMBcGA1UEChMQTWluY29tIFB0eS4gTHRkLjJELMAkGA1UECjMxMxGzAZBgNV
420 BAMTElNTTGVheSBkZWlvIHNLcnZlcjBcMA0GCsGSIb3DQEBAQUAA0sAMEgCQQC3
421 LCXcScWua0PFLkHBLm2VejqPA1F4RQ8q0VjRiPafjx/Z/aWH3ipdMVvuJGa/wFXb
422 /nDFLDlFwP+oCPwhBtVPAGMBAAEwDQYJKoZIhvcNAQEEBQADQQAARNFsihWIjBzb0
423 DcsU0BvL2bvSwJrPEqFlkDq3F4M6EgutL9axEcANWgbbEdAvNJD1dmEmoWny27Pn
424 Ims6ZOZB
425 -----END CERTIFICATE-----
```

426 The manifest and certificate files, when present, shall not be included in the *References* section of the
427 OVF descriptor (see 7.1). This ensures that the OVF descriptor content does not depend on whether the
428 OVF package has a manifest or is signed, and the decision to add a manifest or certificate to a package
429 can be deferred to a later stage.

430 The file extensions `.mf` and `.cert` may be used for other files in an OVF package, as long as they do
431 not occupy the sibling URLs or path names where they would be interpreted as the package manifest or
432 certificate.

433 5.2 Virtual Disk Formats

434 OVF does not require any specific disk format to be used, but to comply with this specification the disk
435 format shall be given by a URI which identifies an unencumbered specification on how to interpret the
436 disk format. The specification need not be machine readable, but it shall be static and unique so that the
437 URI may be used as a key by software reading an OVF package to uniquely determine the format of the
438 disk. The specification shall provide sufficient information so that a skilled person can properly interpret
439 the disk format for both reading and writing of disk data. The URI should be resolvable.

440 5.3 Distribution as a Single File

441 An OVF package may be stored as a single file using the TAR format. The extension of that file shall be
442 `.ova` (open virtual appliance or application).

443 EXAMPLE: The following example shows a sample filename for an OVF package of this type:

```
444 D:\virtualappliances\myapp.ova
```

445 For OVF packages stored as single file, all file references in the OVF descriptor shall be relative-path
446 references and shall point to files included in the TAR archive. Relative directories inside the archive are
447 allowed, but relative-path references shall not contain `..` dot-segments.

448 Ordinarily, a TAR extraction tool would have to scan the whole archive, even if the file requested is found
449 at the beginning, because replacement files can be appended without modifying the rest of the archive.
450 Entries in an OVF TAR file shall exist only once.

451 In addition, the entries shall be in one of the following orders inside the archive:

452 1) OVF descriptor
453 2) The remaining files shall be in the same order as listed in the References section (see 7.1). Note
454 that any external string resource bundle files for internationalization shall be first in the
455 References section (see clause 10).

456 1) OVF descriptor
457 2) OVF manifest
458 3) OVF certificate
459 4) The remaining files shall be in the same order as listed in the References section (see 7.1).
460 Note that any external string resource bundle files for internationalization shall be first in the
461 References section (see clause 10).

462 1) OVF descriptor
463 2) The remaining files shall be in the same order as listed in the References section (see 7.1).
464 Note that any external string resource bundle files for internationalization shall be first in the
465 References section (see clause 10).
466 3) OVF manifest
467 4) OVF certificate

468 For deployment, the ordering restriction ensures that it is possible to extract the OVF descriptor from an
469 OVF TAR file without scanning the entire archive. For generation, the ordering restriction ensures that an
470 OVF TAR file can easily be generated on-the-fly. The restrictions do not prevent OVF TAR files from
471 being created using standard TAR packaging tools.

472 The TAR format used shall comply with the USTAR (Uniform Standard Tape Archive) format as defined
473 by the [ISO/IEC/IEEE 9945:2009](#).

474 5.4 Distribution as a Set of Files

475 An OVF package can be made available as a set of files, for example on a standard Web server.

476 EXAMPLE: An example of an OVF package as a set of files on Web server follows:

```
477 http://mywebsite/virtualappliances/package.ovf  
478 http://mywebsite/virtualappliances/vmdisk1.vmdk  
479 http://mywebsite/virtualappliances/vmdisk2.vmdk  
480 http://mywebsite/virtualappliances/resource.iso  
481 http://mywebsite/virtualappliances/de-DE-resources.xml
```

482 6 OVF Descriptor

483 The OVF descriptor contains the metadata about the package and its contents. This is an extensible
484 XML document for encoding information, such as product details, virtual hardware requirements, and
485 licensing.

486 The DMTF DSP8023 schema definition file for the OVF descriptor contains the elements and attributes.
487 The OVF descriptor shall validate with the DMTF DSP8023.

488 Clauses 7, 8, and 0, describe the semantics, structure, and extensibility framework of the OVF descriptor.
489 These clauses are not a replacement for reading the schema definitions, but they complement the
490 schema definitions.

491 The XML namespaces used in this specification are listed in Table 1. The choice of any namespace prefix
492 is arbitrary and not semantically significant.

493

Table 1 – XML Namespace Prefixes

Prefix	XML Namespace
ovf	http://schemas.dmtf.org/ovf/envelope/2
ovfenv	http://schemas.dmtf.org/ovf/environment/1
rasd	http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData
vssd	http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData
epasd	http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_EthernetPortAllocationSettingData
sasd	http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData
cim	http://schemas.dmtf.org/wbem/wscim/1/common

494 7 Envelope Element

495 The `Envelope` element describes all metadata for the virtual machines (including virtual hardware), as
 496 well as the structure of the OVF package itself.

497 The outermost level of the envelope consists of the following parts:

- 498 • A version indication, defined by the XML namespace URIs.
- 499 • A list of file references to all external files that are part of the OVF package, defined by the
 500 `References` element and its `File` child elements. These are typically virtual disk files, ISO
 501 images, and internationalization resources.
- 502 • A metadata part, defined by section elements, as defined in clause 0.
- 503 • A description of the content, either a single virtual machine (`VirtualSystem` element) or a
 504 collection of multiple virtual machines (`VirtualSystemCollection` element).
- 505 • A specification of message resource bundles for zero or more locales, defined by a `Strings`
 506 element for each locale.

507 **EXAMPLE:** An example of the structure of an OVF descriptor with the top-level `Envelope` element follows:

```

508 <?xml version="1.0" encoding="UTF-8"?>
509 <Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
510   xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-
511   schema/2/CIM_VirtualSystemSettingData"
512   xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
513   schema/2/CIM_ResourceAllocationSettingData"
514   xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2"
515   xmlns="http://schemas.dmtf.org/ovf/envelope/2"
516   xml:lang="en-US">
517   <References>
518     <File ovf:id="de-DE-resources.xml" ovf:size="15240"
519       ovf:href="http://mywebsite/virtualappliances/de-DE-resources.xml"/>
520     <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="180114671"/>
521     <File ovf:id="file2" ovf:href="vmdisk2.vmdk" ovf:size="4882023564"
522   ovf:chunkSize="2147483648"/>
523     <File ovf:id="file3" ovf:href="resource.iso" ovf:size="212148764"
524   ovf:compression="gzip"/>
525     <File ovf:id="icon" ovf:href="icon.png" ovf:size="1360"/>
526   </References>
527   <!-- Describes meta-information about all virtual disks in the package -->
528   <DiskSection>
529     <Info>Describes the set of virtual disks</Info>
  
```



```

530     <!-- Additional section content -->
531     </DiskSection>
532     <!-- Describes all networks used in the package -->
533     <NetworkSection>
534         <Info>List of logical networks used in the package</Info>
535         <!-- Additional section content -->
536     </NetworkSection>
537     <SomeSection ovf:required="false">
538         <Info>A plain-text description of the content</Info>
539         <!-- Additional section content -->
540     </SomeSection>
541     <!-- Additional sections can follow -->
542     <VirtualSystemCollection ovf:id="Some Product">
543         <!-- Additional sections including VirtualSystem or VirtualSystemCollection-->
544     </VirtualSystemCollection >
545     <Strings xml:lang="de-DE">
546         <!-- Specification of message resource bundles for de-DE locale -->
547     </Strings>
548 </Envelope>

```

549 The optional `xml:lang` attribute on the `Envelope` element shall specify the default locale for messages
550 in the descriptor. The optional `Strings` elements shall contain string resource bundles for different
551 locales. See clause 10 for more details on internationalization support.

552 7.1 File References

553 The file reference part defined by the `References` element allows a tool to easily determine the integrity
554 of an OVF package without having to parse or interpret the entire structure of the descriptor. Tools can
555 safely manipulate (for example, copy or archive) OVF packages with no risk of losing files.

556 External string resource bundle files for internationalization shall be placed first in the `References`
557 element, see clause 10 for details.

558 Each `File` element in the reference part shall be given an identifier using the `ovf:id` attribute. The
559 identifier shall be unique inside an OVF package. Each `File` element shall be specified using the
560 `ovf:href` attribute, which shall contain a URL. Relative-path references and the URL schemes "file",
561 "http", and "https" shall be supported, see [RFC1738](#) and [RFC3986](#). Other URL schemes should not
562 be used. If no URL scheme is specified, the value of the `ovf:href` attribute shall be interpreted as a
563 path name of the referenced file relative to the location of the OVF descriptor itself. The relative path
564 name shall use the syntax of relative-path references in [RFC3986](#). The referenced file shall exist. Two
565 different `File` elements shall not reference the same file with their `ovf:href` attributes.

566 The size of the referenced file may be specified using the `ovf:size` attribute. The unit of this attribute
567 shall be bytes. If present, the value of the `ovf:size` attribute should match the actual size of the
568 referenced file.

569 Each file referenced by a `File` element may be compressed using gzip (see [RFC1952](#)). When a `File`
570 element is compressed using gzip, the `ovf:compression` attribute shall be set to "gzip". Otherwise,
571 the `ovf:compression` attribute shall be set to "identity" or the entire attribute omitted. Alternatively,
572 if the href is an HTTP or HTTPS URL, then the compression may be specified by the HTTP server by
573 using the HTTP header `Content-Encoding: gzip` (see [RFC2616](#)). Using HTTP content encoding in
574 combination with the `ovf:compression` attribute is allowed, but in general does not improve the
575 compression ratio. When compression is used, the `ovf:size` attribute shall specify the size of the actual
576 compressed file.

577 Files referenced from the reference part may be split into chunks to accommodate file size restrictions on
578 certain file systems. Chunking shall be indicated by the presence of the `ovf:chunkSize` attribute; the
579 value of `ovf:chunkSize` shall be the size of each chunk, except the last chunk, which may be smaller.

580 When `ovf:chunkSize` is specified, the `File` element shall reference a chunk file representing a chunk
 581 of the entire file. In this case, the value of the `ovf:href` attribute specifies only a part of the URL, and
 582 the syntax for the URL resolving to the chunk file shall be as follows.

```
583 chunk-url      = href-value "." chunk-number
584 chunk-number  = 9(decimal-digit)
585 decimal-digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

586 The syntax is defined in ABNF notation with the exceptions listed in ANNEX A. The `href-value` shall be
 587 the value of the `ovf:href` attribute. The `chunk-number` shall be the 0-based position of the chunk
 588 starting with the value 0 and increasing with increments of 1 for each chunk.

589 If chunking is combined with compression, the entire file shall be compressed before chunking and each
 590 chunk shall be an equal slice of the compressed file, except for the last chunk which may be smaller.

591 If the OVF package has a manifest file, the file name in the manifest entries shall match the value of the
 592 `ovf:href` attribute for the file, except if the file is split into multiple chunks, in which case the `chunk-`
 593 `url` shall be used, and the manifest file shall contain an entry for each individual chunk. If chunked files
 594 are used, the manifest file may contain an entry for the entire file; and if present this digest shall also be
 595 verified.

596 EXAMPLE 1: The following example shows different types of file references:

```
597 <File ovf:id="disk1" ovf:href="disk1.vmdk" />
598 <File ovf:id="disk2" ovf:href="disk2.vmdk" ovf:size="5368709120"
599         ovf:chunkSize="2147483648" />
600 <File ovf:id="iso1" ovf:href="resources/image1.iso" />
601 <File ovf:id="iso2" ovf:href="http://mywebsite/resources/image2.iso" />
```

602 EXAMPLE 2: The following example shows manifest entries corresponding to the file references above:

```
603 SHA1(disk1.vmdk)= 3e19644ec2e806f38951789c76f43e4a0ec7e233
604 SHA1(disk2.vmdk.000000000)= 4f7158731ff434380bf217da248d47a2478e79d8
605 SHA1(disk2.vmdk.000000001)= 12849daeeaf43e7a89550384d26bd437bb8defaf
606 SHA1(disk2.vmdk.000000002)= 4cdd21424bd9eeafa4c42112876217de2ee5556d
607 SHA1(resources/image1.iso)= 72b37ff3fdd09f2a93f1b8395654649b6d06b5b3
608 SHA1(http://mywebsite/resources/image2.iso)=
609 d3c2d179011c970615c5cf10b30957d1c4c968ad
```

610 7.2 Content Element

611 Virtual machine configurations in an OVF package are represented by a `VirtualSystem` or
 612 `VirtualSystemCollection` element. These elements shall be given an identifier using the `ovf:id`
 613 attribute. Direct child elements of a `VirtualSystemCollection` shall have unique identifiers.

614 In the OVF schema, the `VirtualSystem` and `VirtualSystemCollection` elements are part of a
 615 substitution group with the `Content` element as head of the substitution group. The `Content` element is
 616 abstract and cannot be used directly. The OVF descriptor shall have one or more `Content` elements.

617 The `VirtualSystem` element describes a single virtual machine and is simply a container of section
 618 elements. These section elements describe virtual hardware, resources, and product information and are
 619 described in detail in clauses 8 and 0.

620 An example of a `VirtualSystem` element structure is as follows:

```
621 <VirtualSystem ovf:id="simple-app">
622   <Info>A virtual machine</Info>
623   <Name>Simple Appliance</Name>
624   <SomeSection>
625     <!-- Additional section content -->
626   </SomeSection>
627   <!-- Additional sections can follow -->
```

628 `</VirtualSystem>`

629 The `VirtualSystemCollection` element is a container of multiple `VirtualSystem` or
 630 `VirtualSystemCollection` elements. Thus, arbitrary complex configurations can be described. The
 631 section elements at the `VirtualSystemCollection` level describe appliance information, properties,
 632 resource requirements, and so on, and are described in detail in clause 0.

633 An example of a `VirtualSystemCollection` element structure is as follows:

```
634 <VirtualSystemCollection ovf:id="multi-tier-app">
635   <Info>A collection of virtual machines</Info>
636   <Name>Multi-tiered Appliance</Name>
637   <SomeSection>
638     <!-- Additional section content -->
639   </SomeSection>
640   <!-- Additional sections can follow -->
641   <VirtualSystem ovf:id="...">
642     <!-- Additional sections -->
643   </VirtualSystem>
644   <!-- Additional VirtualSystem or VirtualSystemCollection elements can follow-->
645 </VirtualSystemCollection>
```

646 All elements in the `Content` substitution group contain an `Info` element and may contain a `Name`
 647 element. The `Info` element contains a human readable description of the meaning of this entity. The
 648 `Name` element is an optional localizable display name of the content. See clause 10 for details on how to
 649 localize the `Info` and `Name` element.

650 7.3 Extensibility

651 This specification allows custom meta-data to be added to OVF descriptors in several ways:

- 652 • New section elements may be defined as part of the `Section` substitution group, and used
 653 where the OVF schemas allow sections to be present. All subtypes of `Section` contain an `Info`
 654 element that contains a human readable description of the meaning of this entity. The values of
 655 `Info` elements can be used, for example, to give meaningful warnings to users when a section is
 656 being skipped, even if the parser does not know anything about the section. See clause 10 for
 657 details on how to localize the `Info` element.
- 658 • The OVF schemas use an open content model, where all existing types may be extended at the
 659 end with additional elements. Extension points are declared in the OVF schemas with `xs:any`
 660 declarations with `namespace="##other"`.
- 661 • The OVF schemas allow additional attributes on existing types.

662 Custom extensions shall not use XML namespaces defined in this specification. This applies to both
 663 custom elements and custom attributes.

664 On custom elements, a Boolean `ovf:required` attribute specifies whether the information in the
 665 element is required for correct behavior or optional. If not specified, the `ovf:required` attribute defaults
 666 to TRUE. A consumer of an OVF package that detects an extension that is required and that it does not
 667 understand shall fail.

668 For known `Section` elements, if additional child elements that are not understood are found and the
 669 value of their `ovf:required` attribute is TRUE, the consumer of the OVF package shall interpret the
 670 entire section as one it does not understand. The check is not recursive; it applies only to the direct
 671 children of the `Section` element. This behavior ensures that older parsers reject newer OVF
 672 specifications, unless explicitly instructed not to do so.

673 On custom attributes, the information in the attribute shall not be required for correct behavior.

674 EXAMPLE 1:

```
675 <!-- Optional custom section example -->
676 <otherns:IncidentTrackingSection ovf:required="false">
677   <Info>Specifies information useful for incident tracking purposes</Info>
678   <BuildSystem>Acme Corporation Official Build System</BuildSystem>
679   <BuildNumber>102876</BuildNumber>
680   <BuildDate>10-10-2008</BuildDate>
681 </otherns:IncidentTrackingSection>
```

682 EXAMPLE 2:

```
683 <!-- Open content example (extension of existing type) -->
684 <AnnotationSection>
685   <Info>Specifies an annotation for this virtual machine</Info>
686   <Annotation>This is an example of how a future element (Author) can still be
687     parsed by older clients</Annotation>
688   <!-- AnnotationSection extended with Author element -->
689   <otherns:Author ovf:required="false">John Smith</otherns:Author>
690 </AnnotationSection>
```

691 EXAMPLE 3:

```
692 <!-- Optional custom attribute example -->
693 <Network ovf:name="VM network" otherns:desiredCapacity="1 Gbit/s">
694   <Description>The main network for VMs</Description>
695 </Network>
```

696 7.4 Conformance

697 This specification defines three conformance levels for OVF descriptors, with 1 being the highest level of
698 conformance:

- 699 • OVF descriptor uses only sections and elements and attributes that are defined in this
700 specification.
701 Conformance Level: 1.
- 702 • OVF descriptor uses custom sections or elements or attributes that are not defined in this
703 specification, and all such extensions are optional as defined in 7.3.
704 Conformance Level: 2.
- 705 • OVF descriptor uses custom sections or elements that are not defined in this specification and at
706 least one such extension is required as defined in 7.3. The definition of all required extensions
707 shall be publicly available in an open and unencumbered XML Schema. The complete
708 specification may be inclusive in the XML schema or available as a separate document.
709 Conformance Level: 3.

710 The use of conformance level 3 limits portability and should be avoided if at all possible.

711 The conformance level is not specified directly in the OVF descriptor but shall be determined by the
712 above rules.

713 8 Virtual Hardware Description

714 8.1 VirtualHardwareSection

715 Each VirtualSystem element may contain one or more VirtualHardwareSection elements, each of which
716 describes the virtual hardware required by the virtual system. The virtual hardware required by a virtual
717 machine is specified in VirtualHardwareSection elements. This specification supports abstract or
718 incomplete hardware descriptions in which only the major devices are described. The virtualization

719 platform may create additional virtual hardware controllers and devices, as long as the required devices
720 listed in the descriptor are realized.

721
722 This virtual hardware description is based on the CIM classes CIM_VirtualSystemSettingData,
723 CIM_ResourceAllocationSettingData, CIM_EthernetPortAllocationSettingData, and
724 CIM_StorageAllocationSettingData. The XML representation of the CIM model is based on the
725 WS-CIM mapping (DSP0230). Note: This means that the XML elements that belong to the class
726 complex type should be ordered by Unicode code point (binary) order of their CIM property name
727 identifiers.

728 EXAMPLE: Example of VirtualHardwareSection:

```

729 <VirtualHardwareSection>
730   <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
731   <Item>
732     <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
733     <rasd:Description>Virtual CPU</rasd:Description>
734     <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
735     <rasd:InstanceID>1</rasd:InstanceID>
736     <rasd:Reservation>1</rasd:Reservation>
737     <rasd:ResourceType>3</rasd:ResourceType>
738     <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
739     <rasd:VirtualQuantityUnit>Count</ rasd:VirtualQuantityUnit>
740   </Item>
741   <Item>
742     <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
743     <rasd:Description>Memory</rasd:Description>
744     <rasd:ElementName>1 GByte of memory</rasd:ElementName>
745     <rasd:InstanceID>2</rasd:InstanceID>
746     <rasd:Limit>4</rasd:Limit>
747     <rasd:Reservation>4</rasd:Reservation>
748     <rasd:ResourceType>4</rasd:ResourceType>
749   </Item>
750   <EthernetPortItem>
751     <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
752     <rasd:AllocationUnits>bit / second *2^30 </rasd:AllocationUnits> VERIFY
753 the PUnit for Gbits per second
754     <epasd:Connection>VM Network</epasd:Connection>
755     <epasd:Description>Virtual NIC</epasd:Description>
756
757     <epasd:ElementName>Ethernet Port</epasd:ElementName>
758     <epasd:InstanceID>3</epasd:InstanceID>
759     <epasd:NetworkPortProfileID>1</epasd:NetworkPortProfileID>
760     <epasd:NetworkPortProfileIDType>4</epasd:NetworkPortProfileIDType>
761     <epasd:ResourceType>10</epasd:ResourceType>
762     <epasd:VirtualQuantity>1</epasd:VirtualQuantity>
763     <epasd:VirtualQuantityUnits>Count</epasd:VirtualQuantityUnits>
764   </EthernetPortItem>
765   <StorageItem>
766     <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
767     <sasd:Description>Virtual Disk</sasd:Description>
768     <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
769     <sasd:InstanceID>4</sasd:InstanceID>
770     <sasd:Reservation>100</sasd:Reservation>
771     <sasd:ResourceType>31</sasd:ResourceType>
772     <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
773     <sasd:VirtualQuantityUnit>Count</sasd:VirtualQuantityUnit>
774   </StorageItem>
775 </VirtualHardwareSection>

```

776 A `VirtualSystem` element shall have a `VirtualHardwareSection` direct child element.
777 `VirtualHardwareSection` shall not be a direct child element of a `VirtualSystemCollection`
778 element and of an `Envelope` element.

779 Multiple `VirtualHardwareSection` element occurrences are allowed within a single `VirtualSystem`
780 element. The consumer of the OVF package should select the most appropriate virtual hardware
781 description for the particular virtualization platform. A `VirtualHardwareSection` element may contain
782 an `ovf:id` attribute which can be used to identify the element. If present the attribute value must be
783 unique within the `VirtualSystem`.

784 The `ovf:transport` attribute specifies the types of transport mechanisms by which properties are
785 passed to the virtual machine in an OVF environment document. This attribute supports a pluggable and
786 extensible architecture for providing guest/platform communication mechanisms. Several transport types
787 may be specified separated by single space character. See 9.5 for a description of properties and clause
788 11 for a description of transport types and OVF environments.

789 A `VirtualHardwareSection` element contains sub elements that describe virtual system and virtual
790 hardware resources (CPU, memory, network, and storage).

791 A `VirtualHardwareSection` element shall have zero or one `System` direct child element, followed by
792 zero or more `Item` direct child elements, zero or more `EthernetPortItem` direct child elements, and
793 zero or more `StorageItem` direct child elements.

794 The `System` element is an XML representation of the values of one or more properties of the CIM class
795 `CIM_VirtualSystemSettingData`. The `vssd:VirtualSystemType`, a direct child element of
796 `System` element, specifies a virtual system type identifier, which is an implementation defined string that
797 uniquely identifies the type of the virtual system. For example, a virtual system type identifier could be
798 `vmx-4` for VMware's fourth-generation virtual hardware or `xen-3` for Xen's third-generation virtual
799 hardware. Zero or more virtual system type identifiers may be specified separated by single space
800 character. In order for the OVF virtual system to be deployable on a target platform, the virtual machine
801 on the target platform should support at least one of the virtual system types identified in the
802 `vssd:VirtualSystemType` elements. The virtual system type identifiers specified in
803 `vssd:VirtualSystemType` elements are expected to be matched against the values of property
804 `VirtualSystemTypesSupported` of CIM class `CIM_VirtualSystemManagementCapabilities`.

805 The virtual hardware characteristics are described as a sequence of `Item` elements. The `Item` element
806 is an XML representation of an instance of the CIM class `CIM_ResourceAllocationSettingData`.
807 The element can describe all memory and CPU requirements as well as virtual hardware devices.

808 Multiple device subtypes may be specified in an `Item` element, separated by a single space character.

809 EXAMPLE:

810

```
<rasd:ResourceSubType>buslogic lsilogic</rasd:ResourceSubType>
```

811 The network hardware characteristics are described as a sequence of `EthernetPortItem` elements.
812 The `EthernetPortItem` element is an XML representation of the values of one or more properties of
813 the CIM class `CIM_EthernetPortAllocationSettingData`.

814 The storage hardware characteristics are described as a sequence of `StorageItem` elements. The
815 `StorageItem` element is an XML representation of the values of one or more properties of the CIM class
816 `CIM_StorageAllocationSettingData`.

817 **8.2 Extensibility**

818 The optional `ovf:required` attribute on the `Item`, `EthernetPortItem`, or `StorageItem`
 819 element specifies whether the realization of the element (for example, a CD-ROM or USB controller) is
 820 required for correct behavior of the guest software. If not specified, `ovf:required` defaults to TRUE.

821 On child elements of the `Item`, `EthernetPortItem`, or `StorageItem` element, the optional
 822 Boolean attribute `ovf:required` shall be interpreted, even though these elements are in a different
 823 RASD WS-CIM namespace. A tool parsing an `Item` element should act according to Table 2.

824 **Table 2 – Actions for Child Elements with `ovf:required` Attribute**

Child Element	ovf:required Attribute Value	Action
Known	TRUE or not specified	Shall interpret <code>Item</code> , <code>EthernetPortItem</code> , or <code>StorageItem</code>
Known	FALSE	Shall interpret <code>Item</code> , <code>EthernetPortItem</code> , or <code>StorageItem</code>
Unknown	TRUE or not specified	Shall fail <code>Item</code> , <code>EthernetPortItem</code> , or <code>StorageItem</code>
Unknown	FALSE	Shall ignore Child Element

825 **8.3 Virtual Hardware Elements**

826 The element type of the `Item` element in a `VirtualHardwareSection` element is
 827 `CIM_ResourceAllocationSettingData_Type` as defined in [http://schemas.dmtf.org/wbem/wscim/1/cim-](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData.xsd)
 828 [schema/2/CIM_ResourceAllocationSettingData.xsd](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData.xsd).

829 The child elements of `Item` represent the values of one or more properties exposed by the
 830 `CIM_ResourceAllocationSettingData` class. They have the semantics of defined settings as
 831 defined in [DSP1041](#), any profiles derived from [DSP1041](#) for specific resource types, and this document.

832 EXAMPLE: The following example shows a description of memory size:

```
833 <Item>
834   <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
835   <rasd:Description>Memory Size</rasd:Description>
836   <rasd:ElementName>256 MB of memory</rasd:ElementName>
837   <rasd:InstanceID>2</rasd:InstanceID>
838   <rasd:ResourceType>4</rasd:ResourceType>
839   <rasd:VirtualQuantity>256</rasd:VirtualQuantity>
840 </Item>
```

841 The element type of the `EthernetPortItem` element in a `VirtualHardwareSection` element is
 842 `CIM_EthernetPortAllocationSettingData_Type` as defined in [http://schemas.dmtf.org/wbem/wscim/1/cim-](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_EthernetPortAllocationSettingData.xsd)
 843 [schema/2/CIM_EthernetPortAllocationSettingData.xsd](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_EthernetPortAllocationSettingData.xsd).

844 The child elements represent the values of one or more properties exposed by the
 845 `CIM_EthernetPortAllocationSettingData` class. They have the semantics of defined settings as
 846 defined in [DSP1050](#), any profiles derived from [DSP1050](#) for specific Ethernet port resource types, and
 847 this document.

848 EXAMPLE: The following example shows a description of a virtual Ethernet adapter:

```
849 <EthernetPortItem>
850   <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
851   <epasd:Connection>VM Network</epasd:Connection>
852   <epasd:Description>Virtual NIC</epasd:Description>
```

```

853     <epasd:ElementName>Ethernet Port 1</epasd:ElementName>
854     <epasd:InstanceID>3</epasd:InstanceID>
855     <epasd:NetworkPortProfileID>1</epasd:NetworkPortProfileID>
856     <epasd:NetworkPortProfileIDType>4</epasd:NetworkPortProfileIDType>
857     <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
858 </EthernetPortItem>

```

859 The element type of the `StorageItem` element in a `VirtualHardwareSection` element is
 860 `CIM_StorageAllocationSettingData_Type` as defined in [http://schemas.dmtf.org/wbem/wscim/1/cim-](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData.xsd)
 861 [schema/2/CIM_StorageAllocationSettingData.xsd](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData.xsd).

862 The child elements represent the values of one or more properties exposed by the
 863 `CIM_StorageAllocationSettingData` class. They have the semantics of defined settings as defined
 864 in DSP10xx, any profiles derived from DSP10xx for specific storage resource types, and this document.

865 EXAMPLE: The following example shows a description of a virtual storage:

```

866 <StorageItem>
867   <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
868   <sasd:Description>Virtual Disk</sasd:Description>
869   <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
870   <sasd:InstanceID>4</sasd:InstanceID>
871   <sasd:Reservation>100</sasd:Reservation>
872   <sasd:ResourceType>31</sasd:ResourceType>
873   <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
874 </StorageItem>

```

875 The `Description` element is used to provide additional metadata about the `Item`, `EthernetPortItem`, or
 876 `StorageItem` element itself. This element enables a consumer of the OVF package to provide descriptive
 877 information about all items, including items that were unknown at the time the application was written.

878 The `Caption`, `Description` and `ElementName` elements are localizable using the `ovf:msgid`
 879 attribute from the OVF envelope namespace. See clause 10 for more details on internationalization
 880 support.

881 The optional `ovf:configuration` attribute contains a list of configuration names. See 9.8 on
 882 deployment options for semantics of this attribute. The optional `ovf:bound` attribute is used to specify
 883 ranges; see 8.4.

884 Devices such as disks, CD-ROMs, and networks need a backing from the deployment platform. The
 885 requirements on a backing are either specified using the `HostResource` or the `Connection` element.

886 For an Ethernet adapter, a logical network name is specified in the `Connection` element. Ethernet
 887 adapters that refer to the same logical network name within an OVF package shall be deployed on the
 888 same network.

889 The `HostResource` element is used to refer to resources included in the OVF descriptor as well as
 890 logical devices on the deployment platform. Values for `HostResource` elements referring to resources
 891 included in the OVF descriptor are formatted as URIs as specified in Table 3.

892 **Table 3 – HostResource Element**

Content	Description
<code>ovf:/file/<id></code>	A reference to a file in the OVF, as specified in the References section. <code><id></code> shall be the value of the <code>ovf:id</code> attribute of the <code>File</code> element being referenced.
<code>ovf:/disk/<id></code>	A reference to a virtual disk, as specified in the <code>DiskSection</code> or <code>SharedDiskSection</code> . <code><id></code> shall be the value of the <code>ovf:diskId</code> attribute of the <code>Disk</code> element being referenced.

893 If no backing is specified for a device that requires a backing, the deployment platform shall make an
 894 appropriate choice, for example, by prompting the user. More than one backing for a device shall not be
 895 specified.

896 Table 4 gives a brief overview on how elements from rasd, epasd, and sasd namespaces are used to
 897 describe virtual devices and controllers.

898 **Table 4 – Elements for Virtual Devices and Controllers**

Element	Usage
Description	A human-readable description of the meaning of the information. For example, "Specifies the memory size of the virtual machine".
ElementName	A human-readable description of the content. For example, "256MB memory".
InstanceID	A unique instance ID of the element within the section.
HostResource	Abstractly specifies how a device shall connect to a resource on the deployment platform. Not all devices need a backing. See Table 3.
ResourceType OtherResourceType ResourceSubtype	Specifies the kind of device that is being described.
AutomaticAllocation	For devices that are connectable, such as floppies, CD-ROMs, and Ethernet adaptors, this element specifies whether the device should be connected at power on.
Parent	The InstanceID of the parent controller (if any).
Connection	For an Ethernet adapter, this specifies the abstract network connection name for the virtual machine. All Ethernet adapters that specify the same abstract network connection name within an OVF package shall be deployed on the same network. The abstract network connection name shall be listed in the NetworkSection at the outermost envelope level.
Address	Device specific. For an Ethernet adapter, this specifies the MAC address.
AddressOnParent	For a device, this specifies its location on the controller.
AllocationUnits	Specifies the unit of allocation used. For example, "byte * 2^20".
VirtualQuantity	Specifies the quantity of resources presented. For example, "256".
Reservation	Specifies the minimum quantity of resources guaranteed to be available.
Limit	Specifies the maximum quantity of resources that are granted.
Weight	Specifies a relative priority for this allocation in relation to other allocations.

899 Only fields directly related to describing devices are mentioned. Refer to the [CIM MOF](#) for a complete
 900 description of all fields, each field corresponds to the identically named property in the
 901 CIM_ResourceAllocationSettingData class or a class derived from it.

902 8.4 Ranges on Elements

903 The optional `ovf:bound` attribute may be used to specify ranges for the `Item` elements. A range has a
 904 minimum, normal, and maximum value, denoted by `min`, `normal`, and `max`, where `min <= normal <=`
 905 `max`. The default values for `min` and `max` are those specified for `normal`.

906 A platform deploying an OVF package should start with the normal value and adjust the value within the
 907 range for ongoing performance tuning and validation.

908 For the `Item`, `EthernetPortItem`, and `StorageItem` elements in `VirtualHardwareSection`
 909 and `ResourceAllocationSection` elements, the following additional semantics are defined:

- 910 • Each `Item`, `EthernetPortItem`, or `StorageItem` element has an optional `ovf:bound`
 911 attribute. This value may be specified as `min`, `max`, or `normal`. The value defaults to `normal`. If
 912 the attribute is not specified or is specified as `normal`, then the item shall be interpreted as
 913 being part of the regular virtual hardware or resource allocation description.

- 914 • If the `ovf:bound` value is specified as either `min` or `max`, the item is used to specify the upper
 915 or lower bound for one or more values for a given `InstanceID`. Such an item is called a range
 916 marker.

917 The semantics of range markers are as follows:

- 918 • `InstanceID` and `ResourceType` shall be specified, and the `ResourceType` shall match
 919 other `Item` elements with the same `InstanceID`.
- 920 • More than one `min` range marker nor more than one `max` range marker for a given `RASD`,
 921 `EPASD`, or `SASD` (identified with `InstanceID`) shall not be specified..
- 922 • An `Item`, `EthernetPortItem`, or `StorageItem` element with a range marker shall have
 923 a corresponding `Item`, `EthernetPortItem`, or `StorageItem` element without a range
 924 marker, that is, an `Item`, `EthernetPortItem`, and `StorageItem` element with no
 925 `ovf:bound` attribute or `ovf:bound` attribute with value `normal`. This corresponding item
 926 specifies the default value.
- 927 • For an `Item`, `EthernetPortItem`, and `StorageItem` element where only a `min` range
 928 marker is specified, the `max` value is unbounded upwards within the set of valid values for the
 929 property.
- 930 • For an `Item`, `EthernetPortItem`, and `StorageItem` where only a `max` range marker is
 931 specified, the `min` value is unbounded downwards within the set of valid values for the property.
- 932 • The default value shall be inside the range.
- 933 • Non-integer elements shall not be used in the range markers for `RASD`, `EPASD`, or `SASD`.

934 EXAMPLE: The following example shows the use of range markers:

```

935 <VirtualHardwareSection>
936   <Info>...</Info>
937   <Item>
938     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
939     <rasd:ElementName>512 MB memory size</rasd:ElementName>
940     <rasd:InstanceID>0</rasd:InstanceID>
941     <rasd:ResourceType>4</rasd:ResourceType>
942     <rasd:VirtualQuantity>512</rasd:VirtualQuantity>
943   </Item>
944   <Item ovf:bound="min">
945     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
946     <rasd:ElementName>384 MB minimum memory size</rasd:ElementName>
947     <rasd:InstanceID>0</rasd:InstanceID>
948     <rasd:Reservation>384</rasd:Reservation>
949     <rasd:ResourceType>4</rasd:ResourceType>
950   </Item>
951   <Item ovf:bound="max">
952     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
953     <rasd:ElementName>1024 MB maximum memory size</rasd:ElementName>
954     <rasd:InstanceID>0</rasd:InstanceID>
955     <rasd:Reservation>1024</rasd:Reservation>
956     <rasd:ResourceType>4</rasd:ResourceType>
957   </Item>
958 </VirtualHardwareSection>
  
```

959

960 **9 Core Metadata Sections in version 2**961 Table 5 shows the core metadata sections that are defined in the `ovf` namespace.962 **Table 5 – Core Metadata Sections in version 2**

Section	Locations	Multiplicity
DiskSection Describes meta-information about all virtual disks in the package	Envelope	Zero or one
NetworkSection Describes logical networks used in the package	Envelope	Zero or one
ResourceAllocationSection Specifies reservations, limits, and shares on a given resource, such as memory or CPU for a virtual machine collection	VirtualSystemCollection	Zero or one
AnnotationSection Specifies a free-form annotation on an entity	VirtualSystem VirtualSystemCollection	Zero or one
ProductSection Specifies product-information for a package, such as product name and version, along with a set of properties that can be configured	VirtualSystem VirtualSystemCollection	Zero or more
EulaSection Specifies a license agreement for the software in the package	VirtualSystem VirtualSystemCollection	Zero or more
StartupSection Specifies how a virtual machine collection is powered on	VirtualSystemCollection	Zero or one
DeploymentOptionSection Specifies a discrete set of intended resource requirements	Envelope	Zero or one
OperatingSystemSection Specifies the installed guest operating system of a virtual machine	VirtualSystem	Zero or one
InstallSection Specifies that the virtual machine needs to be initially booted to install and configure the software	VirtualSystem	Zero or one
EnvironmentFilesSection Specifies additional files from an OVF package to be included in the OVF environment	VirtualSystem	Zero or one
BootDeviceSection Specifies boot device order to be used by a virtual machine	VirtualSystem	Zero or more
SharedDiskSection Specifies virtual disks shared by more than one VirtualSystems at runtime	Envelope	Zero or one
ScaleOutSection Specifies that a VirtualSystemCollection contain a set of children that are homogeneous with respect to a prototype	VirtualSystemCollection	Zero or more
PlacementGroupSection Specifies a placement policy for a group of VirtualSystems or VirtualSystemCollections	Envelope	Zero or more
PlacementSection Specifies membership of a particular placement policy group	VirtualSystem VirtualSystemCollection	Zero or one
EncryptionSection Specifies encryption scheme for encrypting parts of an OVF descriptor or files that it refers to.	Envelope	Zero or one

963 The following subclauses describe the semantics of the core sections and provide some examples. The
 964 sections are used in several places of an OVF envelope; the description of each section defines where it
 965 may be used. See the OVF schema for a detailed specification of all attributes and elements.

966 In the OVF schema, all sections are part of a substitution group with the `Section` element as head of the
 967 substitution group. The `Section` element is abstract and cannot be used directly.

968 **9.1 DiskSection**

969 A `DiskSection` describes meta-information about virtual disks in the OVF package. Virtual disks and
970 their metadata are described outside the virtual hardware to facilitate sharing between virtual machines
971 within an OVF package. Virtual disks in `DiskSection` can be referenced by multiple virtual machines,
972 but seen from the guest software each virtual machine get individual private disks. Any level of sharing
973 done at runtime is deployment platform specific and not visible to the guest software. See clause 9.13 for
974 details on how to configure sharing of virtual disk at runtime with concurrent access.

975 **EXAMPLE:** The following example shows a description of virtual disks:

```
976 <DiskSection>  
977   <Info>Describes the set of virtual disks</Info>  
978   <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="8589934592"  
979     ovf:populatedSize="3549324972"  
980     ovf:format=  
981       "http://www.vmware.com/interfaces/specifications/vmdk.html#sparse">  
982   </Disk>  
983   <Disk ovf:diskId="vmdisk2" ovf:capacity="536870912"  
984   </Disk>  
985   <Disk ovf:diskId="vmdisk3" ovf:capacity="{disk.size}"  
986     ovf:capacityAllocationUnits="byte * 2^30"  
987   </Disk>  
988 </DiskSection>
```

989 `DiskSection` is a valid section at the outermost envelope level only.

990 Each virtual disk represented by a `Disk` element shall be given an identifier using the `ovf:diskId`
991 attribute; the identifier shall be unique within the `DiskSection`.

992 The capacity of a virtual disk shall be specified by the `ovf:capacity` attribute with an `xs:long` integer
993 value. The default unit of allocation shall be bytes. The optional string attribute
994 `ovf:capacityAllocationUnits` may be used to specify a particular unit of allocation. Values for
995 `ovf:capacityAllocationUnits` shall match the format for programmatic units defined in [DSP0004](#)
996 with the restriction that the base unit shall be "byte".

997 The `ovf:fileRef` attribute denotes the virtual disk content by identifying an existing `File` element in
998 the `References` element, the `File` element is identified by matching its `ovf:id` attribute value with the
999 `ovf:fileRef` attribute value. Omitting the `ovf:fileRef` attribute shall indicate an empty disk. In this
1000 case, the disk shall be created and the entire disk content zeroed at installation time. The guest software
1001 will typically format empty disks in some file system format.

1002 The format URI (see 5.2) of a non-empty virtual disk shall be specified by the `ovf:format` attribute.

1003 Different `Disk` elements shall not contain `ovf:fileRef` attributes with identical values. `Disk` elements
1004 shall be ordered such that they identify any `File` elements in the same order as these are defined in the
1005 `References` element.

1006 For empty disks, rather than specifying a fixed virtual disk capacity, the capacity for an empty disk may be
1007 given using an OVF property, for example `ovf:capacity="{disk.size}"`. The OVF property shall
1008 resolve to an `xs:long` integer value. See 9.5 for a description of OVF properties. The
1009 `ovf:capacityAllocationUnits` attribute is useful when using OVF properties because a user may
1010 be prompted and can then enter disk sizing information in ,for example, gigabytes.

1011 For non-empty disks, the actual used size of the disk may optionally be specified using the
1012 `ovf:populatedSize` attribute. The unit of this attribute shall be bytes. The `ovf:populatedSize`
1013 attribute may be an estimate of used disk size but shall not be larger than `ovf:capacity`.

1014 In `VirtualHardwareSection`, virtual disk devices may have a `rasd:HostResource` element
 1015 referring to a `Disk` element in `DiskSection`; see 8.3. The virtual disk capacity shall be defined by the
 1016 `ovf:capacity` attribute on the `Disk` element. If a `rasd:VirtualQuantity` element is specified along
 1017 with the `rasd:HostResource` element, the virtual quantity value shall not be considered and may have
 1018 any value.

1019 OVF allows a disk image to be represented as a set of modified blocks in comparison to a parent image.
 1020 The use of parent disks can often significantly reduce the size of an OVF package if it contains multiple
 1021 disks with similar content, such as a common base operating system. Actual sharing of disk blocks at
 1022 runtime is optional and deployment platform specific and shall not be visible to the guest software.

1023 For the `Disk` element, a parent disk may optionally be specified using the `ovf:parentRef` attribute,
 1024 which shall contain a valid `ovf:diskId` reference to a different `Disk` element. If a disk block does not
 1025 exist locally, lookup for that disk block then occurs in the parent disk. In `DiskSection`, parent `Disk`
 1026 elements shall occur before child `Disk` elements that refer to them. Similarly, in `References` element,
 1027 the `File` elements referred from these `Disk` elements shall respect the same ordering. The ordering
 1028 restriction ensures that in an OVA archive, parent disks always occur before child disks, making it
 1029 possible for a tool to consume the archive in a streaming mode, see also clause 5.3.

1030 9.2 NetworkSection

1031 The `NetworkSection` element shall list all logical networks used in the OVF package.

```

1032 <NetworkSection>
1033   <Info>List of logical networks used in the package</Info>
1034   <Network ovf:name="VM Network">
1035     <Description>The network that the service will be available on</Description>
1036     <NetworkPortProfile>
1037       <Item>
1038         <epasd:AllocationUnits>GigaBits per Second</epasd:AllocationUnits>
1039         <epasd:ElementName>Network Port Profile 1</epasd:ElementName>
1040         <epasd:InstanceID>1</epasd:InstanceID>
1041         <epasd:NetworkPortProfileID>1</epasd:NetworkPortProfileID>
1042         <epasd:NetworkPortProfileIDType>4</epasd:NetworkPortProfileIDType>
1043         <epasd:Reservation>1</epasd:Reservation>
1044       </Item>
1045     </NetworkPortProfile>
1046   </Network>
1047 </NetworkSection>

```

1048 `NetworkSection` is a valid element at the outermost envelope level. A `Network` element is a child
 1049 element of `NetworkSection`. Each `Network` element in the `NetworkSection` shall be given a unique
 1050 name using the `ovf:name` attribute. The name shall be unique within an `ovf` envelope.

1051 All networks referred to from `Connection` elements in all `VirtualHardwareSection` elements shall
 1052 be defined in the `NetworkSection`.

1053 Starting with version 2.0 of this specification, each logical network may contain a set of networking
 1054 attributes that should be applied when mapping the logical network at deployment time to a physical or
 1055 virtual network. Networking attributes are specified by embedding or referencing zero or more instances
 1056 of network port profile as specified by `NetworkPortProfile` or `NetworkPortProfileURI` child
 1057 element of the `Network` element.

1058 The `NetworkPortProfile` child element of the `Network` element defines the contents of a network
 1059 port profile. The `NetworkPortProfileURI` child element of the `Network` element defines the
 1060 reference to a network port profile.

1061 Examples of using the DSP8049 and EPASD are in ANNEX D.

1062 9.3 ResourceAllocationSection

1063 The ResourceAllocationSection element describes all resource allocation requirements of a
 1064 VirtualSystemCollection entity. These resource allocations shall be performed when deploying the
 1065 OVF package.

```

1066 <ResourceAllocationSection>
1067   <Info>Defines reservations for CPU and memory for the collection of VMs</Info>
1068   <Item>
1069     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
1070     <rasd:ElementName>300 MB reservation</rasd:ElementName>
1071     <rasd:InstanceID>0</rasd:InstanceID>
1072     <rasd:Reservation>300</rasd:Reservation>
1073     <rasd:ResourceType>4</rasd:ResourceType>
1074   </Item>
1075   <Item ovf:configuration="..." ovf:bound="...">
1076     <rasd:AllocationUnits>hertz * 10^6</rasd:AllocationUnits>
1077     <rasd:ElementName>500 MHz reservation</rasd:ElementName>
1078     <rasd:InstanceID>0</rasd:InstanceID>
1079     <rasd:Reservation>500</rasd:Reservation>
1080     <rasd:ResourceType>3</rasd:ResourceType>
1081   </Item>
1082   <EthernetPortItem>
1083     <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
1084     <epasd:Connection>VM Network</epasd:Connection>
1085     <epasd:Description>Virtual NIC</epasd:Description>
1086     <epasd:ElementName>Ethernet Port 1</epasd:ElementName>
1087     <epasd:InstanceID>3</epasd:InstanceID>
1088     <epasd:NetworkPortProfileID>1</epasd:NetworkPortProfileID>
1089     <epasd:NetworkPortProfileIDType>4</epasd:NetworkPortProfileIDType>
1090     <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
1091   </EthernetPortItem>
1092   <StorageItem>
1093     <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
1094     <sasd:Description>Virtual Disk</sasd:Description>
1095     <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
1096     <sasd:InstanceID>4</sasd:InstanceID>
1097     <sasd:Reservation>100</sasd:Reservation>
1098     <sasd:ResourceType>31</sasd:ResourceType>
1099     <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
1100   </StorageItem>
1101 </ResourceAllocationSection>

```

1102 ResourceAllocationSection is a valid element for a VirtualSystemCollection entity.

1103 The optional ovf:configuration attribute contains a list of configuration names. See 9.8 on
 1104 deployment options for semantics of this attribute.

1105 The optional ovf:bound attribute contains a value of min, max, or normal. See 8.4 for semantics of this
 1106 attribute.

1107 9.4 AnnotationSection

1108 The AnnotationSection element is a user-defined annotation on an entity. Such annotations may be
 1109 displayed when deploying the OVF package.

```

1110 <AnnotationSection>
1111   <Info>An annotation on this service. It can be ignored</Info>
1112   <Annotation>Contact customer support if you have any problems</Annotation>
1113 </AnnotationSection >

```

1114 AnnotationSection is a valid element for a VirtualSystem and a VirtualSystemCollection
1115 entity.

1116 See clause 10 for details on how to localize the Annotation element.

1117 9.5 ProductSection

1118 The ProductSection element specifies product-information for an appliance, such as product name,
1119 version, and vendor.

```

1120 <ProductSection ovf:class="com.mycrm.myservice" ovf:instance="1">
1121   <Info>Describes product information for the service</Info>
1122   <Product>MyCRM Enterprise</Product>
1123   <Vendor>MyCRM Corporation</Vendor>
1124   <Version>4.5</Version>
1125   <FullVersion>4.5-b4523</FullVersion>
1126   <ProductUrl>http://www.mycrm.com/enterprise</ProductUrl>
1127   <VendorUrl>http://www.mycrm.com</VendorUrl>
1128   <Icon ovf:height="32" ovf:width="32" ovf:mimeType="image/png" ovf:fileRef="icon">
1129   <Category>Email properties</Category>
1130   <Property ovf:key="admin.email" ovf:type="string" ovf:userConfigurable="true">
1131     <Label>Admin email</Label>
1132     <Description>Email address of administrator</Description>
1133   </Property>
1134   <Category>Admin properties</Category>
1135   <Property ovf:key="app_log" ovf:type="string" ovf:value="low"
1136   ovf:userConfigurable="true">
1137     <Description>Loglevel for the service</Description>
1138   </Property>
1139   <Property ovf:key="app_isSecondary" ovf:value="false" ovf:type="boolean">
1140     <Description>Cluster setup for application server</Description>
1141   </Property>
1142   <Property ovf:key="app_ip" ovf:type="string" ovf:value="{appserver-vm}">
1143     <Description>IP address of the application server VM</Description>
1144   </Property>
1145 </ProductSection>

```

1146 The optional Product element specifies the name of the product, while the optional Vendor element
1147 specifies the name of the product vendor. The optional Version element specifies the product version in
1148 short form, while the optional FullVersion element describes the product version in long form. The
1149 optional ProductUrl element specifies a URL which shall resolve to a human readable description of
1150 the product, while the optional VendorUrl specifies a URL which shall resolve to a human readable
1151 description of the vendor.

1152 The optional AppUrl element specifies a URL resolving to the deployed product instance. The optional
1153 Icon element specifies display icons for the product.

1154 The Property elements specify application-level customization parameters and are particularly relevant
1155 to appliances that need to be customized during deployment with specific settings such as network
1156 identity, the IP addresses of DNS servers, gateways, and others.

1157 The ProductSection is a valid section for a VirtualSystem and a VirtualSystemCollection entity.

1158 The Property elements may be grouped by using Category elements. The set of Property elements
1159 grouped by a Category element is the sequence of Property elements following the Category
1160 element, until but not including an element that is not a Property element. For OVF packages
1161 containing a large number of Property elements, this may provide a simpler installation experience.
1162 Similarly, each Property element may have a short label defined by its Label child element in addition

- 1163 to a description defined by its `Description` child element. See clause 10 for details on how to localize
1164 the `Category` element and the `Description` and `Label` child elements of the `Property` element.
- 1165 Each `Property` element in a `ProductSection` shall be given an identifier that is unique within the
1166 `ProductSection` using the `ovf:key` attribute.
- 1167 Each `Property` element in a `ProductSection` shall be given a type using the `ovf:type` attribute and
1168 optionally type qualifiers using the `ovf:qualifiers` attribute. Valid types are listed in Table 6, and valid
1169 qualifiers are listed in Table 7.
- 1170 The optional attribute `ovf:value` is used to provide a default value for a property. One or more optional
1171 `Value` elements may be used to define alternative default values for different configurations, as defined
1172 in 9.8.
- 1173 The optional attribute `ovf:userConfigurable` determines whether the property value is configurable
1174 during the installation phase. If `ovf:userConfigurable` is `FALSE` or omitted, the `ovf:value` attribute
1175 specifies the value to be used for that customization parameter during installation. If
1176 `ovf:userConfigurable` is `TRUE`, the `ovf:value` attribute specifies a default value for that
1177 customization parameter, which may be changed during installation.
- 1178 A simple OVF implementation such as a command-line installer typically uses default values for
1179 properties and does not prompt even though `ovf:userConfigurable` is set to `TRUE`. To force
1180 prompting at startup time, omitting the `ovf:value` attribute is sufficient for integer types, because the
1181 empty string is not a valid integer value. For string types, prompting may be forced by adding a qualifier
1182 requiring a non-empty string, see Table 7.
- 1183 The optional Boolean attribute `ovf:password` indicates that the property value may contain sensitive
1184 information. The default value is `FALSE`. OVF implementations prompting for property values are advised
1185 to obscure these values when `ovf:password` is set to `TRUE`. This is similar to HTML text input of type
1186 `password`. Note that this mechanism affords limited security protection only. Although sensitive values
1187 are masked from casual observers, default values in the OVF descriptor and assigned values in the OVF
1188 environment are still passed in clear text.
- 1189 Zero or more `ProductSections` may be specified within a `VirtualSystem` or
1190 `VirtualSystemCollection`. Typically, a `ProductSection` corresponds to a particular software
1191 product that is installed. Each product section at the same entity level shall have a unique `ovf:class`
1192 and `ovf:instance` attribute pair. For the common case where only a single `ProductSection` is used,
1193 the `ovf:class` and `ovf:instance` attributes are optional and default to the empty string. The
1194 `ovf:class` property should be used to uniquely identify the software product using the reverse domain
1195 name convention. Examples of values are `com.vmware.tools` and `org.apache.tomcat`. If multiple
1196 instances of the same product are installed, the `ovf:instance` attribute shall be used to identify the
1197 different instances.
- 1198 Property elements are exposed to the guest software through the OVF environment, as described in
1199 clause 11. The value of the `ovfenv:key` attribute of a `Property` element exposed in the OVF
1200 environment shall be constructed from the value of the `ovf:key` attribute of the corresponding
1201 `Property` element defined in a `ProductSection` entity of an OVF descriptor as follows:
- 1202

```
key-value-env = [class-value "."] key-value-prod ["." instance-value]
```
- 1203 The syntax definition above use ABNF with the exceptions listed in ANNEX A, where:
- 1204 • `class-value` is the value of the `ovf:class` attribute of the `Property` element defined in the
1205 `ProductSection` entity. The production `[class-value "."]` shall be present if and only if
1206 `class-value` is not the empty string.

- 1207 • `key-value-prod` is the value of the `ovf:key` attribute of the `Property` element defined in the
- 1208 `ProductSection` entity.
- 1209 • `instance-value` is the value of the `ovf:instance` attribute of the `Property` element defined in
- 1210 the `ProductSection` entity. The production `["." instance-value]` shall be present if and only
- 1211 if `instance-value` is not the empty string.

1212 EXAMPLE: The following OVF environment example shows how properties can be propagated to the guest

1213 software:

```

1214 <Property ovf:key="com.vmware.tools.logLevel" ovf:value="none" />
1215 <Property ovf:key="org.apache.tomcat.logLevel.1" ovf:value="debug" />
1216 <Property ovf:key="org.apache.tomcat.logLevel.2" ovf:value="normal" />
    
```

1217 The consumer of an OVF package should prompt for properties where `ovf:userConfigurable` is

1218 TRUE. These properties may be defined in multiple `ProductSections` as well as in sub-entities in the

1219 OVF package.

1220 If a `ProductSection` exists, then the first `ProductSection` entity defined in the top-level `Content`

1221 element of a package shall define summary information that describes the entire package. After

1222 installation, a consumer of the OVF package could choose to make this information available as an

1223 instance of the `CIM_Product` class.

1224 `Property` elements specified on a `VirtualSystemCollection` are also seen by its immediate

1225 children (see clause 11). Children may refer to the properties of a parent `VirtualSystemCollection`

1226 using macros on the form `#{name}` as value for `ovf:value` attributes.

1227 Table 6 lists the valid types for properties. These are a subset of CIM intrinsic types defined in [DSP0004](#),

1228 which also define the value space and format for each intrinsic type. Each `Property` element shall

1229 specify a type using the `ovf:type` attribute.

1230 **Table 6 – Property Types**

Type	Description
uint8	Unsigned 8-bit integer
sint8	Signed 8-bit integer
uint16	Unsigned 16-bit integer
sint16	Signed 16-bit integer
uint32	Unsigned 32-bit integer
sint32	Signed 32-bit integer
uint64	Unsigned 64-bit integer
sint64	Signed 64-bit integer
String	String
Boolean	Boolean
real32	IEEE 4-byte floating point
real64	IEEE 8-byte floating point

1231 Table 7 lists the supported CIM type qualifiers as defined in [DSP0004](#). Each `Property` element may

1232 optionally specify type qualifiers using the `ovf:qualifiers` attribute with multiple qualifiers separated

1233 by commas; see production `qualifierList` in ANNEX A “MOF Syntax Grammar Description” in

1234 [DSP0004](#).

1235

Table 7 – Property Qualifiers

Type	Description
String	MinLen(min) MaxLen(max) ValueMap{...}
uint8 sint8 uint16 sint16 uint32 sint32 uint64 sint64	ValueMap{...}

1236 9.6 EulaSection

1237 A `EulaSection` contains the legal terms for using its parent `Content` element. This license shall be
 1238 shown and accepted during deployment of an OVF package. Multiple `EulaSections` may be present in
 1239 an OVF. If unattended installations are allowed, all embedded license sections are implicitly accepted.

```

1240 <EulaSection>
1241   <Info>Licensing agreement</Info>
1242   <License>
1243   Lorem ipsum dolor sit amet, ligula suspendisse nulla pretium, rhoncus tempor placerat
1244   fermentum, enim integer ad vestibulum volutpat. Nisl rhoncus turpis est, vel elit,
1245   congue wisi enim nunc ultricies sit, magna tincidunt. Maecenas aliquam maecenas ligula
1246   nostra, accumsan taciti. Sociis mauris in integer, a dolor netus non dui aliquet,
1247   sagittis felis sodales, dolor sociis mauris, vel eu libero cras. Interdum at. Eget
1248   habitasse elementum est, ipsum purus pede porttitor class, ut adipiscing, aliquet sed
1249   auctor, imperdiet arcu per diam dapibus libero duis. Enim eros in vel, volutpat nec
1250   pellentesque leo, scelerisque.
1251   </License>
1252 </EulaSection>
  
```

1253 The `EulaSection` is a valid section for a `VirtualSystem` and a `VirtualSystemCollection` entity.

1254 See clause 10 for details on how to localize the `License` element.

1255 See also clause 10 for description of storing EULA license contents in an external file without any XML
 1256 header or footer. This allows inclusion of standard license or copyright text files in unaltered form.

1257 9.7 StartupSection

1258 The `StartupSection` specifies how a virtual machine collection is powered on and off.

```

1259 <StartupSection>
1260   <Item ovf:id="vm1" ovf:order="0" ovf:startDelay="30" ovf:stopDelay="0"
1261   ovf:startAction="powerOn" ovf:waitingForGuest="true"
1262   ovf:stopAction="powerOff"/>
1263   <Item ovf:id="teamA" ovf:order="0"/>
1264   <Item ovf:id="vm2" ovf:order="1" ovf:startDelay="0" ovf:stopDelay="20"
1265   ovf:startAction="powerOn" ovf:stopAction="guestShutdown"/>
1266 </StartupSection>
  
```

1267 Each `Content` element that is a direct child of a `VirtualSystemCollection` may have a
 1268 corresponding `Item` element in the `StartupSection` entity of the `VirtualSystemCollection` entity.
 1269 Note that `Item` elements may correspond to both `VirtualSystem` and `VirtualSystemCollection`

1270 entities. When a start or stop action is performed on a `VirtualSystemCollection` entity, the
 1271 respective actions on the `Item` elements of its `StartupSection` entity are invoked in the specified
 1272 order. Whenever an `Item` element corresponds to a (nested) `VirtualSystemCollection` entity, the
 1273 actions on the `Item` elements of its `StartupSection` entity shall be invoked before the action on the
 1274 `Item` element corresponding to that `VirtualSystemCollection` entity is invoked (i.e., depth-first
 1275 traversal).

1276 The following required attributes on `Item` are supported for a `VirtualSystem` and
 1277 `VirtualSystemCollection`:

- 1278 • `ovf:id` shall match the value of the `ovf:id` attribute of a `Content` element which is a direct
 1279 child of this `VirtualSystemCollection`. That `Content` element describes the virtual
 1280 machine or virtual machine collection to which the actions defined in the `Item` element apply.
- 1281 • `ovf:order` specifies the startup order using non-negative integer values. If the `ovf:order`
 1282 = "0" then the order is not specified. If the `ovf:order` is non-zero then the of execution of the
 1283 start action shall be the numerical ascending order of the values. The `Items` with same order
 1284 identifier may be started concurrently.

1285 The order of execution of the stop action should be the numerical descending order of the
 1286 values. In implementation specific scenarios the order of execution of the stop action may be
 1287 non-descending.

1288 The following optional attributes on `Item` are supported for a `VirtualSystem`.

- 1289 • `ovf:startDelay` specifies a delay in seconds to wait until proceeding to the next order in the
 1290 start sequence. The default value is 0.
- 1291 • `ovf:waitingForGuest` enables the platform to resume the startup sequence after the guest
 1292 software has reported it is ready. The interpretation of this is deployment platform specific. The
 1293 default value is `FALSE`.
- 1294 • `ovf:startAction` specifies the start action to use. Valid values are `powerOn` and `none`. The
 1295 default value is `powerOn`.
- 1296 • `ovf:stopDelay` specifies a delay in seconds to wait until proceeding to the previous order in
 1297 the stop sequence. The default value is 0.
- 1298 • `ovf:stopAction` specifies the stop action to use. Valid values are `powerOff`,
 1299 `guestShutdown`, and `none`. The interpretation of `guestShutdown` is deployment platform
 1300 specific. The default value is `powerOff`.

1301 If the `StartupSection` is not specified then an `ovf:order="0"` is implied.

1302 9.8 DeploymentOptionSection

1303 The `DeploymentOptionSection` specifies a discrete set of intended resource configurations. The
 1304 author of an OVF package can include sizing metadata for different configurations. A consumer of the
 1305 OVF shall select a configuration, for example, by prompting the user. The selected configuration shall be
 1306 available in the OVF environment file, enabling the guest software to adapt to the selected configuration.
 1307 See clause 11.

1308 The `DeploymentOptionSection` specifies an ID, label, and description for each configuration.

```

1309 <DeploymentOptionSection>
1310   <Configuration ovf:id="minimal">
1311     <Label>Minimal</Label>
1312     <Description>Some description</Description>
1313   </Configuration>
1314   <Configuration ovf:id="normal" ovf:default="true">
1315     <Label>Typical</Label>
1316     <Description>Some description</Description>
  
```

```

1317     </Configuration>
1318     <!-- Additional configurations -->
1319 </DeploymentOptionSection>

```

1320 The DeploymentOptionSection has the following semantics:

- 1321 • If present, the DeploymentOptionSection is valid only at the envelope level, and only one
- 1322 section shall be specified in an OVF descriptor.
- 1323 • The discrete set of configurations is described with Configuration elements, which shall
- 1324 have identifiers specified by the ovf:id attribute that are unique in the package.
- 1325 • A default Configuration element may be specified with the optional ovf:default attribute.
- 1326 If no default is specified, the first element in the list is the default. Specifying more than one
- 1327 element as the default is invalid.
- 1328 • The Label and Description elements are localizable using the ovf:msgid attribute. See
- 1329 clause 10 for more details on internationalization support.

1330 Configurations may be used to control resources for virtual hardware and for virtual machine collections.

1331 Item, EthernetPortItem, and StorageItem elements in VirtualHardwareSection elements

1332 describe resources for VirtualSystem entities, while Item, EthernetPortItem, and StorageItem

1333 elements in ResourceAllocationSection elements describe resources for virtual machine

1334 collections. For these two Item, EthernetPortItem, or StorageItem types, the following

1335 additional semantics are defined:

- 1336 • Each Item EthernetPortItem, and StorageItem has an optional
- 1337 ovf:configuration attribute, containing a list of configurations separated by a single space
- 1338 character. If not specified, the item shall be selected for any configuration. If specified, the item
- 1339 shall be selected only if the chosen configuration ID is in the list. A configuration attribute shall
- 1340 not contain an ID that is not specified in the DeploymentOptionSection.
- 1341 • Within a single VirtualHardwareSection or ResourceAllocationSection, multiple
- 1342 Item, EthernetPortItem, and StorageItem elements are allowed to refer to the same
- 1343 InstanceID. A single combined Item, EthernetPortItem, or StorageItem for the
- 1344 given InstanceID shall be constructed by picking up the child elements of each Item,
- 1345 EthernetPortItem, or StorageItem element, with child elements of a former Item,
- 1346 EthernetPortItem, or StorageItem element in the OVF descriptor not being picked up
- 1347 if there is a like-named child element in a latter Item, EthernetPortItem, or
- 1348 StorageItem element. Any attributes specified on child elements of Item,
- 1349 EthernetPortItem, or StorageItem elements that are not picked up that way, are not
- 1350 part of the combined Item, EthernetPortItem, or StorageItem element.
- 1351 • All Item, EthernetPortItem, StorageItem elements shall specify ResourceType, and
- 1352 Item, EthernetPortItem, and StorageItem elements with the same InstanceID shall
- 1353 agree on ResourceType.

1354 EXAMPLE 1: The following example shows a VirtualHardwareSection:

```

1355 <VirtualHardwareSection>
1356   <Info>...</Info>
1357   <Item>
1358     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
1359     <rasd:ElementName>512 MB memory size and 256 MB
1360 reservation</rasd:ElementName>
1361     <rasd:InstanceID>0</rasd:InstanceID>
1362     <rasd:Reservation>256</rasd:Reservation>
1363     <rasd:ResourceType>4</rasd:ResourceType>
1364     <rasd:VirtualQuantity>512</rasd:VirtualQuantity>
1365   </Item>
1366   ...
1367   <Item ovf:configuration="big">
1368     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>

```

```

1369         <rasd:ElementName>1024 MB memory size and 512 MB
1370 reservation</rasd:ElementName>
1371         <rasd:InstanceID>0</rasd:InstanceID>
1372         <rasd:Reservation>512</rasd:Reservation>
1373         <rasd:ResourceType>4</rasd:ResourceType>
1374         <rasd:VirtualQuantity>1024</rasd:VirtualQuantity>
1375     </Item>
1376 </VirtualHardwareSection>

```

1377 Note that the attributes `ovf:configuration` and `ovf:bound` on `Item` may be used in combination to
 1378 provide very flexible configuration options.

1379 Configurations can further be used to control default values for properties and whether properties are
 1380 user configurable. For `Property` elements inside a `ProductSection`, the following additional semantic
 1381 is defined:

- 1382 • It is possible to specify alternative default property values for different configurations in a
 1383 `DeploymentOptionSection`. In addition to a `Label` and `Description` element, each
 1384 `Property` element may optionally contain `Value` elements. The `Value` element shall have
 1385 an `ovf:value` attribute specifying the alternative default and an `ovf:configuration`
 1386 attribute specifying the configuration in which this new default value should be used. Multiple
 1387 `Value` elements shall not refer to the same configuration.
- 1388 • Starting with version 2.0 of this specification, a `Property` element may optionally have an
 1389 `ovf:configuration` attribute specifying the configuration in which this property should be
 1390 user configurable. The value of `ovf:userConfigurable` is implicitly set to `FALSE` for all
 1391 other configurations, in which case the default value of the property may not be changed
 1392 during installation.

1393 **EXAMPLE 2:** The following shows an example `ProductSection`:

```

1394 <ProductSection>
1395   <Property ovf:key="app.adminEmail" ovf:type="string" ovf:userConfigurable="true"
1396     ovf:configuration="standard">
1397     <Label>Admin email</Label>
1398     <Description>Email address of service administrator</Description>
1399   </Property>
1400   <Property ovf:key="app.log" ovf:type="string" ovf:value="low"
1401     ovf:userConfigurable="true">
1402     <Label>Loglevel</Label>
1403     <Description>Loglevel for the service</Description>
1404     <Value ovf:value="none" ovf:configuration="minimal">
1405   </Property>
1406 </ProductSection>

```

1407 In the example above, the `app.adminEmail` property is only user configurable in the `standard`
 1408 configuration, while the default value for the `app.log` property is changed from `low` to `none` in the
 1409 `minimal` configuration.

1410 9.9 OperatingSystemSection

1411 An `OperatingSystemSection` specifies the operating system installed on a virtual machine.

```

1412 <OperatingSystemSection ovf:id="76">
1413   <Info>Specifies the operating system installed</Info>
1414   <Description>Microsoft Windows Server 2008</Description>
1415 </OperatingSystemSection>

```

1416 The values for `ovf:id` should be taken from the `ValueMap` of the `CIM_OperatingSystem.OsType`
1417 property. The description should be taken from the corresponding `Values` of the
1418 `CIM_OperatingSystem.OsType` property.

1419 The `OperatingSystemSection` is a valid section for a `VirtualSystem` entity only.

1420 9.10 InstallSection

1421 The `InstallSection`, if specified, indicates that the virtual machine needs to be booted once in order
1422 to install and/or configure the guest software. The guest software is expected to access the OVF
1423 environment during that boot, and to shut down after having completed the installation and/or
1424 configuration of the software, powering off the guest.

1425 If the `InstallSection` is not specified, this indicates that the virtual machine does not need to be
1426 powered on to complete installation of guest software.

```
1427 <InstallSection ovf:initialBootStopDelay="300">  
1428   <Info>Specifies that the virtual machine needs to be booted once after having  
1429   created the guest software in order to install and/or configure the software  
1430   </Info>  
1431 </InstallSection>
```

1432 `InstallSection` is a valid section for a `VirtualSystem` entity only.

1433 The optional `ovf:initialBootStopDelay` attribute specifies a delay in seconds to wait for the virtual
1434 machine to power off. If not set, the implementation shall wait for the virtual machine to power off by itself.
1435 If the delay expires and the virtual machine has not powered off, the consumer of the OVF package shall
1436 indicate a failure.

1437 Note that the guest software in the virtual machine can do multiple reboots before powering off.

1438 Several VMs in a virtual machine collection may have an `InstallSection` defined, in which case the
1439 above step is done for each VM, potentially concurrently.

1440 9.11 EnvironmentFilesSection

1441 Clause 11 describes how the OVF environment file is used to deliver runtime customization parameters to
1442 the guest operating system. In version 1 of this specification, the OVF environment file is the only file
1443 delivered to the guest operating system outside of the virtual disk structure. In order to provide additional
1444 deployment time customizations, the `EnvironmentFilesSection` enables the OVF package authors
1445 to specify additional files in the OVF package, outside of the virtual disks, that also is provided to the
1446 guest operating system at runtime via a transport.

1447 This enables increased flexibility in image customization outside of virtual disk capture, allowing OVF
1448 package authors to customize solutions by combining existing virtual disks without modifying them.

1449 For each additional file provided to the guest, the `EnvironmentFilesSection` shall contain a `File`
1450 element with required attributes `ovf:fileRef` and `ovf:path`. The `ovf:fileRef` attribute shall denote
1451 the actual content by identifying an existing `File` element in the `References` element, the `File`
1452 element is identified by matching its `ovf:id` attribute value with the `ovf:fileRef` attribute value. The
1453 `ovf:path` attribute denotes the relative location on the transport where this file will be placed, using the
1454 syntax of relative-path references in [RFC3986](#).

1455 The referenced `File` element in the `References` element identify the content using one of the URL
1456 schemes "file", "http", or "https". For the "file" scheme, the content is static and included in
1457 the OVF package. For the "http" and "https" schemes, the content shall be downloaded prior to the
1458 initial boot of the virtual system.

1459 The `iso` transport shall support this mechanism, see clause 11.2 for details. For this transport, the root
 1460 location relative to `ovf:path` values shall be directory `ovffiles` contained in the root directory of the
 1461 ISO image. The guest software can access the information using standard guest operating system tools.

1462 Other custom transport may support this mechanism. Custom transports will need to specify how to
 1463 access multiple data sources from a root location.

1464 EXAMPLE:

```

1465 <Envelope>
1466   <References>
1467     ...
1468     <File ovf:id="config" ovf:href="config.xml" ovf:size="4332"/>
1469     <File ovf:id="resources" ovf:href="http://mywebsite/resources/resources.zip"/>
1470   </References>
1471   ...
1472   <VirtualSystem ovf:id="...">
1473     ...
1474     <ovf:EnvironmentFilesSection ovf:required="false" ovf:transport="iso">
1475       <Info>Config files to be included in OVF environment</Info>
1476       <ovf:File ovf:fileRef="config" ovf:path="setup/cfg.xml"/>
1477       <ovf:File ovf:fileRef="resources" ovf:path="setup/resources.zip"/>
1478     </ovf:EnvironmentFilesSection>
1479     ...
1480   </VirtualSystem>
1481   ...
1482 </Envelope>

```

1483 In the example above, the file `config.xml` in the OVF package will be copied to the OVF environment
 1484 ISO image and be accessible to the guest software in location `/ovffiles/setup/cfg.xml`, while the
 1485 file `resources.zip` will be accessible in location `/ovffiles/setup/resources.zip`.

1486 9.12 BootDeviceSection

1487 Individual virtual machines will generally use the default device boot order provided by the virtualization
 1488 platform's virtual BIOS. `BootDeviceSection` allows the OVF package author to specify particular boot
 1489 configurations and boot order settings. This enables booting from non-default devices such as a NIC
 1490 using PXE, a USB device or a secondary disk. Moreover there could be multiple boot configurations with
 1491 different boot orders. For example, a virtual disk may be need to be patched before it is bootable and a
 1492 patch ISO image could be included in the OVF package.

1493 The Common Information Model (CIM) defines artifacts to deal with boot order use cases prevalent in the
 1494 industry for BIOSes found in desktops and servers. The boot configuration is defined by the class
 1495 `CIM_BootConfigSetting` which in turn contains one or more `CIM_BootSourceSetting` classes as
 1496 defined in the WS-CIM schema. Each class representing the boot source in turn has either the specific
 1497 device or a "device type" such as disk or CD/DVD as a boot source.

1498 In the context of this specification, the `InstanceID` field of `CIM_BootSourceSetting` is used for
 1499 identifying a specific device as the boot source. The `InstanceID` field of the device as specified in the
 1500 `Item` description of the device in the `VirtualHardwareSection` is used to specify the device as a
 1501 boot source. In case the source is desired to be a device type, the `StructuredBootString` field is
 1502 used to denote the type of device with values defined by the CIM boot control profile. When a boot source
 1503 is a device type, the deployment platform should try all the devices of the specified type.

1504 In the example below, the Pre-Install configuration specifies the boot source as a specific device
 1505 (network), while the Post-Install configuration specifies a device type (hard disk).

1506 EXAMPLE:

```

1507 <Envelope>
1508 ...
1509 <VirtualSystem ovf:id="...">
1510 ...
1511 <ovf:BootDeviceSection>
1512 <Info>Boot device order specification</Info>
1513 <bootc:CIM_BootConfigSetting>
1514 <bootc:Caption>Pre-Install</bootc:Caption>
1515 <bootc:Description>Boot Sequence for fixup of disk</bootc:Description>
1516 <boots:CIM_BootSourceSetting>
1517 <boots:Caption>Fix-up DVD on the network</boots:Caption>
1518 <boots:InstanceID>3</boots:InstanceID> <!-- Network device-->
1519 </boots:CIM_BootSourceSetting>
1520 <boots:CIM_BootSourceSetting>
1521 <boots:Caption>Boot virtual disk</boots:Caption>
1522 <boots:StructuredBootString>CIM:Hard-Disk</boots:StructuredBootString>
1523 </boots:CIM_BootSourceSetting>
1524 </bootc:CIM_BootConfigSetting>
1525 </ovf:BootDeviceSection>
1526 ...
1527 </VirtualSystem>
1528 </Envelope>

```

1529 9.13 SharedDiskSection

1530 The existing `DiskSection` in clause 9.1 describes virtual disks in the OVF package. Virtual disks in the
 1531 `DiskSection` can be referenced by multiple virtual machines, but seen from the guest software each
 1532 virtual machine gets individual private disks. Any level of sharing done at runtime is deployment platform
 1533 specific and not visible to the guest software.

1534 Certain applications such as clustered databases rely on multiple virtual machines sharing the same
 1535 virtual disk at runtime. `SharedDiskSection` allows the OVF package author to specify `Disk` elements
 1536 shared by more than one `VirtualSystem` at runtime, these could be virtual disks backing by an external
 1537 `File` reference, or empty virtual disks without backing. It is recommended that the guest software use
 1538 cluster-aware file system technology to be able to handle concurrent access.

1539 EXAMPLE:

```

1540 <ovf:SharedDiskSection>
1541 <Info>Describes the set of virtual disks shared between VMs</Info>
1542 <ovf:SharedDisk ovf:diskId="datadisk" ovf:fileRef="data"
1543 <ovf:capacity="8589934592" ovf:populatedSize="3549324972"
1544 <ovf:format=
1545 "http://www.vmware.com/interfaces/specifications/vmdk.html#sparse"/>
1546 <ovf:SharedDisk ovf:diskId="transientdisk" ovf:capacity="536870912"/>
1547 </ovf:SharedDiskSection>

```

1548 `SharedDiskSection` is a valid section at the outermost envelope level only.

1549 Each virtual disk is represented by a `SharedDisk` element that shall be given an identifier using the
 1550 `ovf:diskId` attribute; the identifier shall be unique within the combined content of `DiskSection` and
 1551 `SharedDiskSection`. The `SharedDisk` element has the same structure as the `Disk` element in
 1552 `DiskSection`, with the addition of an optional boolean attribute `ovf:readOnly` stating whether shared
 1553 disk access is read-write or read-only.

1554 Shared virtual disks are referenced from virtual hardware using the using the `HostResource` element as
 1555 described in clause 8.3.

1556 It is optional for the virtualization platform to support `SharedDiskSection`. The platform should give an
 1557 appropriate error message based on the value of the `ovf:required` attribute on the
 1558 `SharedDiskSection` element.

1559 9.14 ScaleOutSection

1560 The number of `VirtualSystems` and `VirtualSystemCollections` contained in an OVF package is generally
 1561 fixed and determined by the structure inside the `Envelope` element. The `ScaleOutSection` allows a
 1562 `VirtualSystemCollection` to contain a set of children that are homogeneous with respect to a prototypical
 1563 `VirtualSystem` or `VirtualSystemCollection`. The `ScaleOutSection` shall cause the deployment platform
 1564 to replicate the prototype a number of times, thus allowing the number of instantiated virtual machines to
 1565 be configured dynamically at deployment time.

1566 EXAMPLE:

```
1567 <VirtualSystemCollection ovf:id="web-tier">
1568   ...
1569   <ovf:ScaleOutSection ovf:id="web-server">
1570     <Info>Web tier</Info>
1571     <ovf:Description>Number of web server instances in web tier</ovf:Description>
1572     <ovf:InstanceCount ovf:default="4" ovf:minimum="2" ovf:maximum="8"/>
1573   </ovf:ScaleOutSection>
1574   ...
1575   <VirtualSystem ovf:id="web-server">
1576     <Info>Prototype web server</Info>
1577     ...
1578   </VirtualSystem>
1579 </VirtualSystemCollection>
```

1580 In the example above, the deployment platform creates a web tier containing between two and eight web
 1581 server virtual machine instances, with a default instance count of four. The deployment platform makes
 1582 an appropriate choice (e.g., by prompting the user). Assuming three replicas were created, the OVF
 1583 environment available to the guest software in the first replica has the following content structure:

1584 EXAMPLE:

```
1585 <Environment ... ovfenv:id="web-server-1">
1586   ...
1587   <Entity ovfenv:id="web-server-2">
1588     ...
1589   </Entity>
1590   <Entity ovfenv:id="web-server-3">
1591     ...
1592   </Entity>
1593 </Environment>
```

1594 This mechanism enables dynamic scaling of virtual machine instances at deployment time. Scaling at
 1595 runtime is not within the scope of this specification.

1596 The `ScaleOutSection` is a valid section inside `VirtualSystemCollection` only.

1597 The `ovf:id` attribute on `ScaleOutSection` identifies the `VirtualSystem` or `VirtualSystemCollection`
 1598 prototype to be replicated.

1599 For the `InstanceCount` element, the `ovf:minimum` and `ovf:maximum` attribute values shall be non-
 1600 negative integers and `ovf:minimum` shall be less than or equal to the value of `ovf:maximum`. The
 1601 `ovf:minimum` value may be zero in which case the `VirtualSystemCollection` may contain zero instances
 1602 of the prototype. If the `ovf:minimum` attribute is not present, it shall be assumed to have a value of one.
 1603 If the `ovf:maximum` attribute is not present, it shall be assumed to have a value of unbounded. The
 1604 `ovf:default` attribute is required and shall contain a value within the range defined by `ovf:minimum`
 1605 and `ovf:maximum`.

1606 Each replicated instance shall be assigned a unique `ovf:id` value within the `VirtualSystemCollection`.
 1607 The unique `ovf:id` value shall be constructed from the prototype `ovf:id` value with a sequence
 1608 number appended as follows:

```
1609 replica-ovf-id = prototype-ovf-id "-" decimal-number
1610 decimal-number = decimal-digit | (decimal-digit decimal-number)
1611 decimal-digit  = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

1612 The syntax definitions above use ABNF with the exceptions listed in ANNEX A. The first replica shall
 1613 have sequence number one and following sequence numbers shall be incremented by one for each
 1614 replica. Note that after deployment, no `VirtualSystem` will have the prototype `ovf:id` value itself.

1615 If the prototype being replicated has a starting order in the `StartupSection`, all replicas shall share this
 1616 value. It is not possible to specify a particular starting sequence among replicas.

1617 Property values for Property elements in the prototype are prompted for once per replica created. If the
 1618 OVF package author requires a property value to be shared among instances, that Property may be
 1619 declared at the containing `VirtualSystemCollection` level and referenced by replicas as described in
 1620 clause 9.5.

1621 Configurations from the `DeploymentOptionSection` may be used to control values for `InstanceCount`. The
 1622 `InstanceCount` element may have an `ovf:configuration` attribute specifying the configuration in
 1623 which this element should be used. Multiple elements shall not refer to the same configuration, and a
 1624 configuration attribute shall not contain an `ovf:id` value that is not specified in the
 1625 `DeploymentOptionSection`.

1626 EXAMPLE:

```
1627 <VirtualSystemCollection ovf:id="web-tier">
1628   ...
1629   <DeploymentOptionSection>
1630     <Info>Deployment size options</Info>
1631     <Configuration ovf:id="minimal">
1632       <Label>Minimal</Label>
1633       <Description>Minimal deployment scenario</Description>
1634     </Configuration>
1635     <Configuration ovf:id="common" ovf:default="true">
1636       <Label>Typical</Label>
1637       <Description>Common deployment scenario</Description>
1638     </Configuration>
1639     ...
1640   </DeploymentOptionSection>
1641   ...
1642   <ovf:ScaleOutSection ovf:id="web-server">
1643     <Info>Web tier</Info>
1644     <ovf:Description>Number of web server instances in web tier</ovf:Description>
1645     <ovf:InstanceCount ovf:default="4"/>
1646     <ovf:InstanceCount ovf:default="1" ovf:configuration="minimal"/>
1647   </ovf:ScaleOutSection>
1648   ...
1649 </VirtualSystemCollection>
```

1650 In the example above, the default replica count is four, unless the minimal deployment scenario is
 1651 chosen, in which case the default is one.

1652 9.15 PlacementGroupSection and PlacementSection

1653 Certain types of applications require the ability to specify that two or more `VirtualSystems` should be
 1654 deployed closely together since they rely on very fast communication or a common hardware dependency
 1655 such as a reliable communication link. Other types of applications require the ability to specify that two or

1656 more VirtualSystems should be deployed apart due to high-availability or disaster recovery
1657 considerations.

1658 PlacementGroupSection allow an OVF package author to define a placement policy for a group of
1659 VirtualSystems, while PlacementSection allow the author to annotate elements with membership of a
1660 particular placement policy group.

1661 Zero or more PlacementGroupSections may be declared at the Envelope level, while
1662 PlacementSection may be declared at the VirtualSystem or VirtualSystemCollection level. Declaring a
1663 VirtualSystemCollection member of a placement policy group applies transitively to all child VirtualSystem
1664 and child Virtual System Collections elements. The ovf:id attribute on PlacementGroupSection is
1665 used to identify the particular placement policy; the attribute value shall be unique within the OVF
1666 package. Placement policy group membership is specified using the ovf:group attribute on
1667 PlacementSection; the attribute value shall match the value of an ovf:id attribute on a
1668 PlacementGroupSection.

1669 This version of the specification defines the placement policies "affinity" and "availability",
1670 specified with the required ovf:policy attribute on PlacementGroupSection.

1671 Placement policy "affinity" describe that VirtualSystems should be placed as closely together as
1672 possible. The deployment platform should attempt to keep these virtual machines located as adjacently
1673 as possible, typically on the same physical host or with fast network connectivity between hosts.

1674 Placement policy "availability" describe that VirtualSystems should be placed separately. The
1675 deployment platform should attempt to keep these virtual machines located apart, typically on the
1676 different physical hosts.

1677 EXAMPLE:

```
1678 <Envelope ...>
1679   ...
1680   <ovf:PlacementGroupSection ovf:id="web" ovf:policy="availability">
1681     <Info>Placement policy for group of VMs</Info>
1682     <ovf:Description>Placement policy for web tier</ovf:Description>
1683   </ovf:PlacementGroupSection>
1684   ...
1685   <VirtualSystemCollection ovf:id="web-tier">
1686     ...
1687     <ovf:ScaleOutSection ovf:id="web-node">
1688       <Info>Web tier</Info>
1689       ...
1690     </ovf:ScaleOutSection>
1691     ...
1692     <VirtualSystem ovf:id="web-node">
1693       <Info>Web server</Info>
1694       ...
1695       <ovf2:PlacementSection ovf:group="web">
1696         <Info>Placement policy group reference</Info>
1697       </ovf2:PlacementSection>
1698       ...
1699     </VirtualSystem>
1700   </VirtualSystemCollection>
1701 </Envelope>
```

1702 In the example above, all virtual machines in the compute tier should be placed separately for high
1703 availability. This example also use the ScaleOutSection defined in clause 9.14, in which case each
1704 replica get the policy assigned.

1705 9.16 Encryption Section

1706 For licensing and other reasons it is desirable to have an encryption scheme enabling free exchange of
 1707 OVF appliances while ensuring that only the intended parties can use them. The XML Encryption Syntax
 1708 and Processing standard is utilized to encrypt either the files in the reference section or any parts of the
 1709 XML markup of an OVF document.

1710 The various aspects of OVF encryption are as shown below:

- 1711 1. block encryption
 1712 The OVF document author shall utilize block encryption algorithms as specified in the XML
 1713 encryption 1.1 documents (ref) for this purpose.
- 1714 2. key derivation
 1715 The OVF author may use the appropriate key for this purpose. If the key is derived using a
 1716 passphrase then the author shall use one of the key derivations specified in the XML Encryption
 1717 1.1 standard.
- 1718 3. key transport.
 1719 If the encryption key is embedded in the OVF document, the specified key transport mechanisms
 1720 shall be used.

1721 This specification defines a new section called the EncryptionSection as a focal point for the encryption
 1722 functionality. This new section provides a single location for placing the encryption algorithm related
 1723 markup and the corresponding reference list to point to the OVF content that has been encrypted.

1724 Note that depending on which parts of the OVF markup has been encrypted, an OVF descriptor may not
 1725 validate against the OVF schemas until decrypted.

1726 Below is an example of an OVF encryption section with encryption methods utilized in the OVF
 1727 document, and the corresponding reference list pointing to the items that have been encrypted.

1728 **EXAMPLE:**

```

1729 <ovf:EncryptionSection>
1730 <!-- This section contains two different methods of encryption and the corresponding
1731 backpointers to the data that is encrypted ->
1732 <!-- Method#1: Pass phrase based Key derivation ->
1733 <!-- The following derived key block defines PBKDF2 and the corresponding back
1734 pointers to the encrypted data elements -->
1735 <!-- Use a salt value "ovfpassword" and iteration count of 4096 --->
1736 <xenc11:DerivedKey>
1737 <xenc11:KeyDerivationMethod
1738 Algorithm="http://www.rsasecurity.com/rsalabs/pkcs/schemas/pkcs-5#pbkdf2"/>
1739 <pkcs-5:PBKDF2-params>
1740 <Salt>
1741 <Specified>ovfpassword</Specified>
1742 </Salt>
1743 <IterationCount>4096</IterationCount>
1744 <KeyLength>16</KeyLength>
1745 <PRF Algorithm="http://www.w3.org/2001/04/xmldsig-more#hmac-sha256"/>
1746 </pkcs-5:PBKDF2-params>
1747 ...
1748 <!-- The ReferenceList element below contains references to the file Ref-109.vhd via
1749 the URI syntax which is specified by XML Encryption.
1750 --->
1751 <xenc:ReferenceList>
1752 <xenc:DataReference URI="#first.vhd" />
1753 <xenc:DataReference URI=... />
1754 <xenc:DataReference URI=... />
1755 </xenc:ReferenceList>
1756 </xenc11:DerivedKey>
1757 <!-- Method#2: The following example illustrates use of a symmetric key
1758 transported using the public key within a certificate ->

```

```

1759 <xenc:EncryptedKey>
1760   <xenc:EncryptionMethod   Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-
1761   1_5" />
1762     <ds:KeyInfo xmlns:ds='http://www.w3.org/2000/09/xmldsig#'
1763       <ds:X509Data>
1764         <ds:X509Certificate> ... </ds:X509Certificate>
1765       </ds:X509Data>
1766     </ds:KeyInfo>
1767     <xenc:CipherData>
1768       <xenc:CipherValue> ... </xenc:CipherValue>
1769     </xenc:CipherData>
1770 <!-- The ReferenceList element below contains reference #second-xml-fragment" to the
1771 XML fragment that has been encrypted using the above method --->
1772 <xenc:ReferenceList>
1773   <xenc:DataReference URI='#second-xml-fragment' />
1774   <xenc:DataReference URI='...' />
1775   <xenc:DataReference URI='...' />
1776 </xenc:ReferenceList>
1777 </xenc:EncryptedKey>
1778 </ovf:EncryptionSection>

```

1779 Below is an example of the encrypted file which is referenced in the EncryptionSection above using
 1780 URI='Ref-109.vhd' syntax.

1781 **EXAMPLE:**

```

1782 <ovf:References>
1783 <ovf:File ovf:id="Xen:9cb10691-4012-4aeb-970c-3d47a906bfff/0b13bdba-3761-8622-22fc-
1784 2e252ed9ce14" ovf:href="Ref-109.vhd">
1785 <!-- the encrypted file referenced by the package is enclosed by an EncryptedData with
1786 a CipherReference to the actual encrypted file. The EncryptionSection in this example
1787 has a back pointer to it under the PBKDF2 algorithm via Id="first.vhd". This tells the
1788 decrypter how to decrypt the file -->
1789 <xenc:EncryptedData Id="first.vhd" Type='http://www.w3.org/2001/04/xmlenc#Element' >
1790   <xenc:EncryptionMethod
1791 Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc" />
1792   <xenc:CipherData>
1793     <xenc:CipherReference URI='Ref-109.vhd' />
1794   </xenc:CipherData>
1795 </xenc:EncryptedData>
1796 </ovf:File>
1797 </ovf:References>

```

1798 Below is an example of the encrypted OVF markup which is referenced in the EncryptionSection above
 1799 using URI='#second-xml-fragment' syntax.

1800 **EXAMPLE:**

```

1801 <!-- the EncryptedData element below encompasses encrypted xml from the original
1802 document. It is provided with the Id "first-xml-fragment" which allows it to be
1803 referenced from the EncryptionSection. -->
1804 <xenc:EncryptedData Type=http://www.w3.org/2001/04/xmlenc#Element Id="second-xml-
1805 fragment">
1806 <!-- Each EncryptedData specifies its own encryption method. -->
1807   <xenc:EncryptionMethod Algorithm=http://www.w3.org/2001/04/xmlenc#aes128-cbc/>
1808   <xenc:CipherData>
1809     <!-- Encrypted content --->
1810     <xenc:CipherValue>DEADBEEF</xenc:CipherValue>
1811   </xenc:CipherData>
1812 </xenc:EncryptedData>

```

1813 10 Internationalization

1814 The following elements support localizable messages using the optional `ovf:msgid` attribute:

- 1815 • Info element on Content
- 1816 • Name element on Content
- 1817 • Info element on Section
- 1818 • Annotation element on AnnotationSection
- 1819 • License element on EulaSection
- 1820 • Description element on NetworkSection
- 1821 • Description element on OperatingSystemSection
- 1822 • Description, Product, Vendor, Label, and Category elements on ProductSection
- 1823 • Description and Label elements on Property
- 1824 • Description and Label elements on DeploymentOptionSection
- 1825 • ElementName, Caption and Description subelements on the System element in
- 1826 VirtualHardwareSection
- 1827 • ElementName, Caption and Description subelements on Item elements in
- 1828 VirtualHardwareSection
- 1829 • ElementName, Caption and Description subelements on Item elements in
- 1830 ResourceAllocationSection

1831 The `ovf:msgid` attribute contains an identifier that refers to a message that may have different values in
1832 different locales.

1833 EXAMPLE 1:

```
1834 <Info ovf:msgid="info.text">Default info.text value if no locale is set or no locale
1835 match</Info>
1836 <License ovf:msgid="license.tomcat-6_0"/> <!-- No default message -->
```

1837 The `xml:lang` attribute on the `Envelope` element shall specify the default locale for messages in the
1838 descriptor. The attribute is optional with a default value of "en-US".

1839 10.1 Internal Resource Bundles

1840 Message resource bundles can be internal or external to the OVF descriptor. Internal resource bundles
1841 are represented as `Strings` elements at the end of the `Envelope` element.

1842 EXAMPLE 2:

```
1843 <ovf:Envelope xml:lang="en-US">
1844   ...
1845   ... sections and content here ...
1846   ...
1847   <Info msgid="info.os">Operating System</Info>
1848   ...
1849   <Strings xml:lang="da-DA">
1850     <Msg ovf:msgid="info.os">Operativsystem</Msg>
1851     ...
1852   </Strings>
1853   <Strings xml:lang="de-DE">
1854     <Msg ovf:msgid="info.os">Betriebssystem</Msg>
1855     ...
1856   </Strings>
1857 </ovf:Envelope>
```

1858 10.2 External Resource Bundles

1859 External resource bundles shall be listed first in the `References` section and referred to from `Strings`
 1860 elements. An external message bundle follows the same schema as the embedded one. Exactly one
 1861 `Strings` element shall be present in an external message bundle, and that `Strings` element may not
 1862 have an `ovf:fileRef` attribute specified.

1863 EXAMPLE 3:

```
1864 <ovf:Envelope xml:lang="en-US">
1865   <References>
1866     ...
1867     <File ovf:id="it-it-resources" ovf:href="resources/it-it-bundle.msg"/>
1868   </References>
1869   ... sections and content here ...
1870   ...
1871   <Strings xml:lang="it-IT" ovf:fileRef="it-it-resources"/>
1872   ...
1873 </ovf:Envelope>
```

1874 EXAMPLE 4: Example content of external resources/it-it-bundle.msg file, which is referenced in previous example:

```
1875 <Strings
1876   xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/1"
1877   xmlns="http://schemas.dmtf.org/ovf/envelope/1"
1878   xml:lang="it-IT">
1879   <Msg ovf:msgid="info.os">Sistema operativo</Msg>
1880   ...
1881 </Strings>
```

1882 The embedded and external `Strings` elements may be interleaved, but they shall be placed at the end
 1883 of the `Envelope` element. If multiple occurrences of a `msgid` attribute with a given locale occur, a latter
 1884 value overwrites a former.

1885 10.3 Message Content in External File

1886 Starting with version 2.0 of this specification, the content of all localizable messages may be stored in an
 1887 external file using the optional `ovf:fileRef` attribute on the `Msg` element. For the `License` element on
 1888 `EulaSection` in particular, this allows inclusion of a standard license text file in unaltered form without
 1889 any XML header or footer.

1890 The `ovf:fileRef` attribute denotes the message content by identifying an existing `File` element in the
 1891 `References` element, the `File` element is identified by matching its `ovf:id` attribute value with the
 1892 `ovf:fileRef` attribute value. The content of an external file referenced using `ovf:fileRef` shall be
 1893 interpreted as plain text in UTF-8 Unicode.

1894 If the referenced file is not found, the embedded content of the `Msg` element shall be used.

1895 The optional `ovf:fileRef` attribute may appear on `Msg` elements in both internal and external `Strings`
 1896 resource bundles.

1897 EXAMPLE 5:

```
1898 <Envelope xml:lang="en-US">
1899   <References>
1900     <File ovf:id="license-en-US" ovf:href="license-en-US.txt"/>
1901     <File ovf:id="license-de-DE" ovf:href="license-de-DE.txt"/>
1902   </References>
1903   ...
1904   <VirtualSystem ovf:id="...">
1905     <EulaSection>
1906       <Info>Licensing agreement</Info>
1907       <License ovf:msgid="license">Unused</License>
```

```

1908     </EulaSection>
1909     ...
1910 </VirtualSystem>
1911     ...
1912 <Strings xml:lang="en-US">
1913     <Msg ovf:msgid="license" ovf:fileRef="license-en-US">Invalid license</Msg>
1914 </Strings>
1915 <Strings xml:lang="de-DE">
1916     <Msg ovf:msgid="license" ovf:fileRef="license-de-DE">Ihre Lizenz ist nicht
1917 gültig</Msg>
1918 </Strings>
1919 </Envelope>

```

1920 In the example above, the default license agreement is stored in plain text file `license-en-US.txt`,
 1921 while the license agreement for the `de-DE` locale is stored in file `license-de-DE.txt`.

1922 Note that the above mechanism works for all localizable elements and not just `License`.

1923 11 OVF Environment

1924 The OVF environment defines how the guest software and the deployment platform interact. This
 1925 environment allows the guest software to access information about the deployment platform, such as the
 1926 user-specified values for the properties defined in the OVF descriptor.

1927 The environment specification is split into a *protocol* part and a *transport* part. The *protocol* part defines
 1928 the format and semantics of an XML document that can be made accessible to the guest software. The
 1929 *transport* part defines how the information is communicated between the deployment platform and the
 1930 guest software.

1931 The `dsp8027_1.1.0.xsd` XML schema definition file for the OVF environment contains the elements
 1932 and attributes.

1933 11.1 Environment Document

1934 The environment document is an extensible XML document that is provided to the guest software about
 1935 the environment in which it is being executed. The way that the document is obtained depends on the
 1936 transport type.

1937 EXAMPLE: An example of the structure of the OVF environment document follows:

```

1938 <?xml version="1.0" encoding="UTF-8"?>
1939 <Environment xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1940     xmlns:ovfenv="http://schemas.dmtf.org/ovf/environment/1"
1941     xmlns="http://schemas.dmtf.org/ovf/environment/1"
1942     ovfenv:id="identification of VM from OVF descriptor">
1943     <!-- Information about virtualization platform -->
1944     <PlatformSection>
1945         <Kind>Type of virtualization platform</Kind>
1946         <Version>Version of virtualization platform</Version>
1947         <Vendor>Vendor of virtualization platform</Vendor>
1948         <Locale>Language and country code</Locale>
1949         <TimeZone>Current timezone offset in minutes from UTC</TimeZone>
1950     </PlatformSection>
1951     <!-- Properties defined for this virtual machine -->
1952     <PropertySection>
1953         <Property ovfenv:key="key" ovfenv:value="value">
1954             <!-- More properties -->
1955         </PropertySection>
1956     <Entity ovfenv:id="id of sibling virtual system or virtual system collection">
1957         <PropertySection>
1958             <!-- Properties from sibling -->

```



```

1959     </PropertySection>
1960     </Entity>
1961 </Environment>
    
```

1962 The value of the `ovfenv:id` attribute of the `Environment` element shall match the value of the `ovf:id`
 1963 attribute of the `VirtualSystem` entity describing this virtual machine.

1964 The `PlatformSection` element contains optional information provided by the deployment platform.
 1965 Elements `Kind`, `Version`, and `Vendor` describe deployment platform vendor details; these elements are
 1966 experimental. Elements `Locale` and `TimeZone` describe the current locale and time zone; these
 1967 elements are experimental.

1968 The `PropertySection` element contains `Property` elements with key/value pairs corresponding to all
 1969 properties specified in the OVF descriptor for the current virtual machine, as well as properties specified
 1970 for the immediate parent `VirtualSystemCollection`, if one exists. The environment presents
 1971 properties as a simple list to make it easy for applications to parse. Furthermore, the single list format
 1972 supports the override semantics where a property on a `VirtualSystem` may override one defined on a
 1973 parent `VirtualSystemCollection`. The overridden property shall not be in the list. Overriding may
 1974 occur if a property in the current virtual machine and a property in the parent
 1975 `VirtualSystemCollection` has identical `ovf:key`, `ovf:class`, and `ovf:instance` attribute
 1976 values; see 9.5. In this case, the value of an overridden parent property may be obtained by adding a
 1977 differently named child property referencing the parent property with a macro; see 9.5.

1978 An `Entity` element shall exist for each sibling `VirtualSystem` and `VirtualSystemCollection`, if
 1979 any are present. The value of the `ovfenv:id` attribute of the `Entity` element shall match the value of
 1980 the `ovf:id` attribute of the sibling entity. The `Entity` elements contain the property key/value pairs in
 1981 the sibling's OVF environment documents, so the content of an `Entity` element for a particular sibling
 1982 shall contain the exact `PropertySection` seen by that sibling. This information can be used, for
 1983 example, to make configuration information such as IP addresses available to `VirtualSystems` being
 1984 part of a multi-tiered application.

1985 Table 8 shows the core sections that are defined.

1986 **Table 8 – Core Sections**

Section	Location	Multiplicity
<code>PlatformSection</code> Provides information from the deployment platform	Environment	Zero or one
<code>PropertySection</code> Contains key/value pairs corresponding to properties defined in the OVF descriptor	Environment Entity	Zero or one

1987 The environment document is extensible by providing new section types. A consumer of the document
 1988 should ignore unknown section types and elements.

1989 11.2 Transport

1990 The environment document information can be communicated in a number of ways to the guest software.
 1991 These ways are called transport types. The transport types are specified in the OVF descriptor by the
 1992 `ovf:transport` attribute of `VirtualHardwareSection`. Several transport types may be specified,
 1993 separated by a single space character, in which case an implementation is free to use any of them. The
 1994 transport types define methods by which the environment document is communicated from the
 1995 deployment platform to the guest software.

1996 To enable interoperability, this specification defines an "iso" transport type which all implementations
 1997 that support CD-ROM devices are required to support. The `iso` transport communicates the environment
 1998 document by making a dynamically generated ISO image available to the guest software. To support the

- 1999 `iso` transport type, prior to booting a virtual machine, an implementation shall make an ISO read-only
2000 disk image available as backing for a disconnected CD-ROM. If the `iso` transport is selected for a
2001 `VirtualHardwareSection`, at least one disconnected CD-ROM device shall be present in this section.
- 2002 The generated ISO image shall comply with the ISO 9660 specification with support for Joliet extensions.
- 2003 The ISO image shall contain the OVF environment for this particular virtual machine, and the environment
2004 shall be present in an XML file named `ovf-env.xml` that is contained in the root directory of the ISO
2005 image. The guest software can now access the information using standard guest operating system tools.
- 2006 If the virtual machine prior to booting had more than one disconnected CD-ROM, the guest software may
2007 have to scan connected CD-ROM devices in order to locate the ISO image containing the `ovf-env.xml`
2008 file.
- 2009 The ISO image containing the OVF environment shall be made available to the guest software on every
2010 boot of the virtual machine.
- 2011 Support for the "`iso`" transport type is not a requirement for virtual hardware architectures or guest
2012 operating systems which do not have CD-ROM device support.
- 2013 To be compliant with this specification, any transport format other than `iso` shall be given by a URI which
2014 identifies an unencumbered specification on how to use the transport. The specification need not be
2015 machine readable, but it shall be static and unique so that it may be used as a key by software reading an
2016 OVF descriptor to uniquely determine the format. The specification shall be sufficient for a skilled person
2017 to properly interpret the transport mechanism for implementing the protocols. The URIs should be
2018 resolvable.

ANNEX A (informative)

2019
2020
2021
2022

Symbols and Conventions

2023 XML examples use the XML namespace prefixes defined in Table 1. The XML examples use a style to
2024 not specify namespace prefixes on child elements. Note that XML rules define that child elements
2025 specified without namespace prefix are from the namespace of the parent element, and not from the
2026 default namespace of the XML document. Throughout the document, whitespace within XML element
2027 values is used for readability. In practice, a service can accept and strip leading and trailing whitespace
2028 within element values as if whitespace had not been used.

2029 Syntax definitions in this document use Augmented BNF (ABNF) as defined in IETF [RFC5234](#) with the
2030 following exceptions:

- 2031 • Rules separated by a bar (|) represent choices, instead of using a forward slash (/) as defined in
2032 ABNF.
- 2033 • Any characters must be processed case sensitively, instead of case-insensitively as defined in
2034 ABNF.
- 2035 • Whitespace (i.e., the space character U+0020 and the tab character U+0009) is allowed between
2036 syntactical elements, instead of assembling elements without whitespace as defined in ABNF.

2037

**ANNEX B
(normative)****OVF XSD**2038
2039
2040
2041

2042 Normative copies of the XML schemas for this specification may be retrieved by resolving the following
2043 URLs:

2044
2045
2046

<http://schemas.dmtf.org/ovf/envelope/2/dsp8023.xsd>
<http://schemas.dmtf.org/ovf/environment/1/dsp8027.xsd>

2047 Any `xs:documentation` content in XML schemas for this specification is informative and provided only
2048 for convenience.

2049 Normative copies of the XML schemas for the WS-CIM mapping ([DSP0230](#)) of
2050 `CIM_ResourceAllocationSystemSettingsData`, `CIM_VirtualSystemSettingData`,
2051 `CIM_EthernetPortAllocationSettingData`, `CIM_StorageAllocationSettingData` and
2052 `CIM_OperatingSystem`, may be retrieved by resolving the following URLs:

2053
2054
2055
2056
2057
2058
2059

http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData.xsd
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData.xsd
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_EthernetPortAllocationSettingData.xsd
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData.xsd

2060 This specification is based on the following CIM MOFs:

2061
2062
2063
2064
2065
2066

`CIM_VirtualSystemSettingData.mof`
`CIM_ResourceAllocationSettingData.mof`
`CIM_EthernetPortAllocationSettingData.mof`
`CIM_StorageAllocationSettingData.mof`
`CIM_OperatingSystem.mof`

ANNEX C (informative)

OVF Mime Type Registration Template

- 2071 Registration Template
- 2072 To: ietf-types@iana.org
- 2073 Subject: Registration of media type Application/OVF
- 2074 Type name: Application
- 2075 Subtype name: OVF
- 2076 Required parameters: none
- 2077 Optional parameters: none
- 2078 Encoding considerations: binary
- 2079 Security considerations:
- 2080 • An OVF package contains active content that is expected to be launched in a virtual machine.
2081 The OVF standard, section 5.1 states: “An OVF package may be signed by signing the manifest
2082 file. The digest of the manifest file is stored in a certificate file with extension .cert file along with
2083 the base64-encoded X.509 certificate. The .cert file shall have the same base name as the .ovf
2084 file and be a sibling of the .ovf file. A consumer of the OVF package shall verify the signature and
2085 should validate the certificate.
 - 2086 • An OVF package may contain passwords as part of the configuration information. The OVF
2087 standard, section 9.5 states: “The optional Boolean attribute ovf:password indicates that the
2088 property value may contain sensitive information. The default value is FALSE. OVF
2089 implementations prompting for property values are advised to obscure these values when
2090 ovf:password is set to TRUE. This is similar to HTML text input of type password. Note that this
2091 mechanism affords limited security protection only. Although sensitive values are masked from
2092 casual observers, default values in the OVF descriptor and assigned values in the OVF
2093 environment are still passed in clear text. “
- 2094 Interoperability considerations:
- 2095 • OVF has demonstrated interoperability via multiple, interoperating implementations in the market.
- 2096 Published specification:
- 2097 • DSP0243_2.0.0.pdf
- 2098 Applications that use this media type:
- 2099 • Implementations of the DMTF Standard: Cloud Infrastructure Management Interface (CIMI)
2100 (DSP0263_1.0.0.pdf)
 - 2101 • Implementations of the SNIA Cloud Data Management Interface (CDMI) – OVF Extension
- 2102 Additional information:
- 2103 • Magic number(s): none
 - 2104 • File extension(s): .ova
 - 2105 • Macintosh file type code(s): none
 - 2106 • Person & email address to contact for further information:

- 2107 • Intended usage: (One of COMMON, LIMITED USE or OBSOLETE.)
- 2108 • Restrictions on usage: (Any restrictions on where the media type can be used go here.)
- 2109 • Author:
- 2110 • Change controller:

ANNEX D (informative)

Network Port Profile Examples

D.1 Example 1 (OVF Descriptor for One Virtual System and One Network with an Inlined Network Port Profile)

The example below shows an OVF descriptor that describes a virtual system and a network it connects to. The virtual system description in this example uses an inlined network port profile that is described as an XML element that contains child XML elements from epasd namespace. The network described in the network section uses the same network port profile description. The network port profile described in this example is used to reserve 1 Gbps of bandwidth.

```

2122 <?xml version="1.0" encoding="UTF-8"?>
2123 <Envelope xsi:schemaLocation="http://schemas.dmtf.org/ovf/envelope/2
2124 file:///C:/dsp8023_2.0.0_wgv0.9.5.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2125 xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2" xmlns="http://schemas.dmtf.org/ovf/envelope/2"
2126 xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData"
2127 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2128 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2129 schema/2/CIM_EthernetPortAllocationSettingData"
2130 xmlns:sasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData">
2131 <!-- References to all external files -->
2132 <References>
2133 <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="2000000000"/>
2134 </References>
2135 <!-- Describes meta-information for all virtual disks in the package -->
2136 <DiskSection>
2137 <Info>Describes the set of virtual disks</Info>
2138 <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="4294967296"
2139 ovf:format="http://www.examplecompany.com/interfaces/specifications/vmdk.html#sparse"/>
2140 </DiskSection>
2141 <!-- Describes all networks used in the package -->
2142 <NetworkSection>
2143 <Info>List of logical networks used in the package</Info>
2144 <Network ovf:name="VM Network">
2145 <Description>The network that the VMs connect to</Description>
2146 <NetworkPortProfile>
2147 <!-- Network port profile describing bandwidth reservation. Network port profile
2148 is identified by UUID. -->
2149 <Item>
2150 <epasd:AllocationUnits>bit / second * 10^9</epasd:AllocationUnits>
2151 <epasd:ElementName>Network Port Profile 1</epasd:ElementName>
2152 <epasd:InstanceID>1</epasd:InstanceID>
2153 <epasd:NetworkPortProfileID>aaaaaaaa-bbbb-cccc-dddd-
2154 eeeeeeeeeee</epasd:NetworkPortProfileID>
2155 <epasd:NetworkPortProfileIDType>3</epasd:NetworkPortProfileIDType>
2156 <epasd:Reservation>1</epasd:Reservation>
2157 </Item>
2158 </NetworkPortProfile>
2159 </Network>
2160 </NetworkSection>
2161 <VirtualSystem ovf:id="vm">
2162 <Info>Describes a virtual machine</Info>
2163 <Name>Virtual Appliance One</Name>
2164 <ProductSection>
2165 <Info>Describes product information for the appliance</Info>
2166 <Product>The Great Appliance</Product>
2167 <Vendor>Some Great Corporation</Vendor>
2168 <Version>13.00</Version>
2169 <FullVersion>13.00-b5</FullVersion>
2170 <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2171 <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2172 <Property ovf:key="admin.email" ovf:type="string">

```

```

2173         <Description>Email address of administrator</Description>
2174     </Property>
2175     <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2176         <Description>The IP address of this appliance</Description>
2177     </Property>
2178 </ProductSection>
2179 <AnnotationSection ovf:required="false">
2180     <Info>A random annotation on this service. It can be ignored</Info>
2181     <Annotation>Contact customer support if you have any problems</Annotation>
2182 </AnnotationSection>
2183 <EulaSection>
2184     <Info>License information for the appliance</Info>
2185     <License>Insert your favorite license here</License>
2186 </EulaSection>
2187 <VirtualHardwareSection>
2188     <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2189     <Item>
2190         <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2191         <rasd:Description>Virtual CPU</rasd:Description>
2192         <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2193         <rasd:InstanceID>1</rasd:InstanceID>
2194         <rasd:Reservation>1</rasd:Reservation>
2195         <rasd:ResourceType>3</rasd:ResourceType>
2196         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2197     </Item>
2198     <Item>
2199         <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2200         <rasd:Description>Memory</rasd:Description>
2201         <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2202         <rasd:InstanceID>2</rasd:InstanceID>
2203         <rasd:ResourceType>4</rasd:ResourceType>
2204         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2205     </Item>
2206     <EthernetPortItem>
2207         <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
2208         <epasd:AllocationUnits>bit / second * 10^9 </epasd:AllocationUnits>
2209         <epasd:Connection>VM Network</epasd:Connection>
2210         <epasd:Description>Virtual NIC</epasd:Description>
2211         <epasd:ElementName>Ethernet Port</epasd:ElementName>
2212
2213         <epasd:InstanceID>3</epasd:InstanceID>
2214         <epasd:NetworkPortProfileID>aaaaaaa-bbbb-cccc-dddd-
2215 eeeeeeeee</epasd:NetworkPortProfileID>
2216         <epasd:NetworkPortProfileIDType>3</epasd:NetworkPortProfileIDType>
2217         <epasd:Reservation>1</epasd:Reservation>
2218         <epasd:ResourceType>10</epasd:ResourceType>
2219         <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
2220     </EthernetPortItem>
2221     <StorageItem>
2222         <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2223         <sasd:Description>Virtual Disk</sasd:Description>
2224         <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2225         <sasd:InstanceID>4</sasd:InstanceID>
2226         <sasd:Reservation>100</sasd:Reservation>
2227         <sasd:ResourceType>31</sasd:ResourceType>
2228         <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
2229     </StorageItem>
2230 </VirtualHardwareSection>
2231 <OperatingSystemSection ovf:id="58" ovf:required="false">
2232     <Info>Guest Operating System</Info>
2233     <Description>OS</Description>
2234 </OperatingSystemSection>
2235 </VirtualSystem>
2236 </Envelope>

```


2237 D.2 Example 2 (OVF Descriptor for One Virtual System and One Network with a 2238 Locally Referenced Network Port Profile)

2239 The example below shows an OVF descriptor that describes a virtual system and a network it connects
2240 to. The virtual system description in this example uses a network port profile that is described in a local
2241 file that is contained in the same OVF package. The network described in the network section uses the
2242 same network port profile description. The network port profile described in this example is used to
2243 reserve 1 Gbps of bandwidth. The network described in the network section uses the same network port
2244 profile description. The network port profile described in this example is used to reserve 1 Gbps of
2245 bandwidth.

```
2246 <?xml version="1.0" encoding="UTF-8"?>
2247 <Envelope xsi:schemaLocation="http://schemas.dmtf.org/ovf/envelope/2
2248 file:///C:/dsp8023_2.0.0_wgv0.9.5.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2249 xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2" xmlns="http://schemas.dmtf.org/ovf/envelope/2"
2250 xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData"
2251 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2252 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2253 schema/2/CIM_EthernetPortAllocationSettingData"
2254 xmlns:sasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData">
2255 <!-- References to all external files -->
2256 <References>
2257 <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="2000000000"/>
2258 <File ovf:id="networkportprofile1" ovf:href="NetworkPortProfile1.xml"/>
2259 </References>
2260 <!-- Describes meta-information for all virtual disks in the package -->
2261 <DiskSection>
2262 <Info>Describes the set of virtual disks</Info>
2263 <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="4294967296"
2264 ovf:format="http://www.examplecompany.com/interfaces/specifications/vmdk.html#sparse"/>
2265 </DiskSection>
2266 <!-- Describes all networks used in the package -->
2267 <NetworkSection>
2268 <Info>List of logical networks used in the package</Info>
2269 <Network ovf:name="VM Network">
2270 <Description>The network that VMs connect to</Description>
2271 <NetworkPortProfileURI>file:networkportprofile1</NetworkPortProfileURI>
2272 </Network>
2273 </NetworkSection>
2274 <VirtualSystem ovf:id="vm">
2275 <Info>Describes a virtual machine</Info>
2276 <Name>Virtual Appliance One</Name>
2277 <ProductSection>
2278 <Info>Describes product information for the appliance</Info>
2279 <Product>The Great Appliance</Product>
2280 <Vendor>Some Great Corporation</Vendor>
2281 <Version>13.00</Version>
2282 <FullVersion>13.00-b5</FullVersion>
2283 <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2284 <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2285 <Property ovf:key="admin.email" ovf:type="string">
2286 <Description>Email address of administrator</Description>
2287 </Property>
2288 <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2289 <Description>The IP address of this appliance</Description>
2290 </Property>
2291 </ProductSection>
2292 <AnnotationSection ovf:required="false">
2293 <Info>A random annotation on this service. It can be ignored</Info>
2294 <Annotation>Contact customer support if you have any problems</Annotation>
2295 </AnnotationSection>
2296 <EulaSection>
2297 <Info>License information for the appliance</Info>
2298 <License>Insert your favorite license here</License>
2299 </EulaSection>
2300 <VirtualHardwareSection>
2301 <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2302 <Item>
```

```

2303     <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2304     <rasd:Description>Virtual CPU</rasd:Description>
2305     <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2306     <rasd:InstanceID>1</rasd:InstanceID>
2307     <rasd:Reservation>1</rasd:Reservation>
2308     <rasd:ResourceType>3</rasd:ResourceType>
2309     <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2310   </Item>
2311   <Item>
2312     <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2313     <rasd:Description>Memory</rasd:Description>
2314     <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2315     <rasd:InstanceID>2</rasd:InstanceID>
2316     <rasd:ResourceType>4</rasd:ResourceType>
2317     <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2318   </Item>
2319   <EthernetPortItem>
2320     <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
2321     <epasd:Connection>VM Network</epasd:Connection>
2322     <epasd:Description>Virtual NIC</epasd:Description>
2323     <epasd:ElementName>Ethernet Port</epasd:ElementName>
2324
2325     <epasd:InstanceID>3</epasd:InstanceID>
2326     <epasd:NetworkPortProfileID>file:networkportprofile1</epasd:NetworkPortProfileID>
2327     <epasd:NetworkPortProfileIDType>2</epasd:NetworkPortProfileIDType>
2328     <epasd:ResourceType>10</epasd:ResourceType>
2329     <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
2330   </EthernetPortItem>
2331   <StorageItem>
2332     <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2333     <sasd:Description>Virtual Disk</sasd:Description>
2334     <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2335     <sasd:InstanceID>4</sasd:InstanceID>
2336     <sasd:Reservation>100</sasd:Reservation>
2337     <sasd:ResourceType>31</sasd:ResourceType>
2338     <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
2339   </StorageItem>
2340 </VirtualHardwareSection>
2341 <OperatingSystemSection ovf:id="58" ovf:required="false">
2342   <Info>Guest Operating System</Info>
2343   <Description>OS</Description>
2344 </OperatingSystemSection>
2345 </VirtualSystem>
2346 </Envelope>

```

2347 **D.3 Example 3 (OVF Descriptor for One Virtual System and One Network with a** 2348 **Network Port Profile referenced by a URI)**

2349 The example below shows an OVF descriptor that describes a virtual system and a network it connects
2350 to. The virtual system description in this example uses a network port profile that is described by a URI.
2351 The network described in the network section uses the same network port profile description. The
2352 network port profile described in this example is used to reserve 1 Gbps of bandwidth.

```

2353 <?xml version="1.0" encoding="UTF-8"?>
2354 <Envelope xsi:schemaLocation="http://schemas.dmtf.org/ovf/envelope/2
2355 file:///C:/dsp8023_2.0.0_wgv0.9.5.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2356 xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2" xmlns="http://schemas.dmtf.org/ovf/envelope/2"
2357 xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData"
2358 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2359 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2360 schema/2/CIM_EthernetPortAllocationSettingData"
2361 xmlns:sasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData">
2362 <!-- References to all external files -->
2363 <References>
2364   <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="2000000000"/>
2365 </References>
2366 <!-- Describes meta-information for all virtual disks in the package -->
2367 <DiskSection>
2368   <Info>Describes the set of virtual disks</Info>

```

```

2369     <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="4294967296"
2370 ovf:format="http://www.examplecompany.com/interfaces/specifications/vmdk.html#sparse"/>
2371   </DiskSection>
2372   <!-- Describes all networks used in the package -->
2373   <NetworkSection>
2374     <Info>List of logical networks used in the package</Info>
2375     <Network ovf:name="VM Network">
2376       <Description>The network that the VMs connect to</Description>
2377
2378     <NetworkPortProfileURI>http://www.dmtf.org/networkportprofiles/networkportprofile1.xml</Netwo
2379 rkPortProfileURI>
2380   </Network>
2381 </NetworkSection>
2382 <VirtualSystem ovf:id="vm">
2383   <Info>Describes a virtual machine</Info>
2384   <Name>Virtual Appliance One</Name>
2385   <ProductSection>
2386     <Info>Describes product information for the appliance</Info>
2387     <Product>The Great Appliance</Product>
2388     <Vendor>Some Great Corporation</Vendor>
2389     <Version>13.00</Version>
2390     <FullVersion>13.00-b5</FullVersion>
2391     <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2392     <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2393     <Property ovf:key="admin.email" ovf:type="string">
2394       <Description>Email address of administrator</Description>
2395     </Property>
2396     <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2397       <Description>The IP address of this appliance</Description>
2398     </Property>
2399   </ProductSection>
2400   <AnnotationSection ovf:required="false">
2401     <Info>A random annotation on this service. It can be ignored</Info>
2402     <Annotation>Contact customer support if you have any problems</Annotation>
2403   </AnnotationSection>
2404   <EulaSection>
2405     <Info>License information for the appliance</Info>
2406     <License>Insert your favorite license here</License>
2407   </EulaSection>
2408   <VirtualHardwareSection>
2409     <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2410     <Item>
2411       <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2412       <rasd:Description>Virtual CPU</rasd:Description>
2413       <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2414       <rasd:InstanceID>1</rasd:InstanceID>
2415       <rasd:Reservation>1</rasd:Reservation>
2416       <rasd:ResourceType>3</rasd:ResourceType>
2417       <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2418     </Item>
2419     <Item>
2420       <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2421       <rasd:Description>Memory</rasd:Description>
2422       <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2423       <rasd:InstanceID>2</rasd:InstanceID>
2424       <rasd:ResourceType>4</rasd:ResourceType>
2425       <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2426     </Item>
2427     <EthernetPortItem>
2428       <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
2429       <epasd:Connection>VM Network</epasd:Connection>
2430       <epasd:Description>Virtual NIC</epasd:Description>
2431       <epasd:ElementName>Ethernet Port</epasd:ElementName>
2432
2433       <epasd:InstanceID>3</epasd:InstanceID>
2434
2435       <epasd:NetworkPortProfileID>http://www.dmtf.org/networkportprofiles/networkportprofile1.xml</
2436 epasd:NetworkPortProfileID>
2437       <epasd:NetworkPortProfileIDType>2</epasd:NetworkPortProfileIDType>
2438       <epasd:ResourceType>10</epasd:ResourceType>
2439       <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>

```

```

2440 </EthernetPortItem>
2441 <StorageItem>
2442   <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2443   <sasd:Description>Virtual Disk</sasd:Description>
2444   <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2445   <sasd:InstanceID>4</sasd:InstanceID>
2446   <sasd:Reservation>100</sasd:Reservation>
2447   <sasd:ResourceType>31</sasd:ResourceType>
2448   <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
2449 </StorageItem>
2450 </VirtualHardwareSection>
2451 <OperatingSystemSection ovf:id="58" ovf:required="false">
2452   <Info>Guest Operating System</Info>
2453   <Description>OS</Description>
2454 </OperatingSystemSection>
2455 </VirtualSystem>
2456 </Envelope>

```

2457 **D.4 Example 4 (OVF Descriptor for Two Virtual Systems and One Network with** 2458 **Two Network Port Profiles referenced by URIs)**

2459 The example below shows an OVF descriptor that describes two virtual systems and a network they
2460 connect to. Each virtual system description in this example uses a network port profile that is described
2461 by a URI. The network described in the network section uses the same two network port profiles. The two
2462 network port profiles described in this example are used to reserve 1 Gbps of bandwidth and describe
2463 general network traffic respectively. Annex D.5 and D.6 are examples of these network port profiles.

```

2464 <?xml version="1.0" encoding="UTF-8"?>
2465 <Envelope xsi:schemaLocation="http://schemas.dmtf.org/ovf/envelope/2
2466 file:///C:/dsp8023_2.0.0_wgv0.9.5.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2467 xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2" xmlns="http://schemas.dmtf.org/ovf/envelope/2"
2468 xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData"
2469 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2470 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2471 schema/2/CIM_EthernetPortAllocationSettingData"
2472 xmlns:sasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData">
2473 <!-- References to all external files -->
2474 <References>
2475   <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="2000000000"/>
2476 </References>
2477 <!-- Describes meta-information for all virtual disks in the package -->
2478 <DiskSection>
2479   <Info>Describes the set of virtual disks</Info>
2480   <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="4294967296"
2481 ovf:format="http://www.examplecompany.com/interfaces/specifications/vmdk.html#sparse"/>
2482 </DiskSection>
2483 <!-- Describes all networks used in the package -->
2484 <NetworkSection>
2485   <Info>List of logical networks used in the package</Info>
2486   <Network ovf:name="VM Network">
2487     <Description>The network that the VMs connect to</Description>
2488     <!-- Network port profile for storage traffic -->
2489
2490     <NetworkPortProfileURI>http://www.dmtf.org/networkportprofiles/networkportprofile1.xml</Netwo
2491 rkPortProfileURI>
2492     <!-- Network port profile for networking traffic -->
2493
2494     <NetworkPortProfileURI>http://www.dmtf.org/networkportprofiles/networkportprofile2.xml</Netwo
2495 rkPortProfileURI>
2496   </Network>
2497 </NetworkSection>
2498 <VirtualSystemCollection ovf:id="vsc1">
2499   <Info>Collection of 2 VMs</Info>
2500   <VirtualSystem ovf:id="storage server">
2501     <Info>Describes a virtual machine</Info>
2502     <Name>Virtual Appliance One</Name>
2503     <ProductSection>
2504       <Info>Describes product information for the appliance</Info>

```

```

2505     <Product>The Great Appliance</Product>
2506     <Vendor>Some Great Corporation</Vendor>
2507     <Version>13.00</Version>
2508     <FullVersion>13.00-b5</FullVersion>
2509     <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2510     <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2511     <Property ovf:key="admin.email" ovf:type="string">
2512         <Description>Email address of administrator</Description>
2513     </Property>
2514     <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2515         <Description>The IP address of this appliance</Description>
2516     </Property>
2517 </ProductSection>
2518 <AnnotationSection ovf:required="false">
2519     <Info>A random annotation on this service. It can be ignored</Info>
2520     <Annotation>Contact customer support if you have any problems</Annotation>
2521 </AnnotationSection>
2522 <EulaSection>
2523     <Info>License information for the appliance</Info>
2524     <License>Insert your favorite license here</License>
2525 </EulaSection>
2526 <VirtualHardwareSection>
2527     <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2528     <Item>
2529         <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2530         <rasd:Description>Virtual CPU</rasd:Description>
2531         <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2532         <rasd:InstanceID>1</rasd:InstanceID>
2533         <rasd:Reservation>1</rasd:Reservation>
2534         <rasd:ResourceType>3</rasd:ResourceType>
2535         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2536     </Item>
2537     <Item>
2538         <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2539         <rasd:Description>Memory</rasd:Description>
2540         <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2541         <rasd:InstanceID>2</rasd:InstanceID>
2542         <rasd:ResourceType>4</rasd:ResourceType>
2543         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2544     </Item>
2545     <EthernetPortItem>
2546         <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
2547         <epasd:Connection>VM Network</epasd:Connection>
2548         <epasd:Description>Virtual NIC</epasd:Description>
2549
2550         <epasd:ElementName>Ethernet Port</epasd:ElementName>
2551
2552         <epasd:InstanceID>3</epasd:InstanceID>
2553
2554         <epasd:NetworkPortProfileID>http://www.dmtf.org/networkportprofiles/networkportprofile1.xml</
2555     epasd:NetworkPortProfileID>
2556         <epasd:NetworkPortProfileIDType>2</epasd:NetworkPortProfileIDType>
2557         <epasd:ResourceType>10</epasd:ResourceType>
2558         <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
2559     </EthernetPortItem>
2560     <StorageItem>
2561         <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2562         <sasd:Description>Virtual Disk</sasd:Description>
2563         <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2564         <sasd:InstanceID>4</sasd:InstanceID>
2565         <sasd:Reservation>100</sasd:Reservation>
2566         <sasd:ResourceType>31</sasd:ResourceType>
2567         <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
2568     </StorageItem>
2569 </VirtualHardwareSection>
2570 <OperatingSystemSection ovf:id="58" ovf:required="false">
2571     <Info>Guest Operating System</Info>
2572     <Description>OS</Description>
2573 </OperatingSystemSection>
2574 </VirtualSystem>
2575 <VirtualSystem ovf:id="web-server">

```

```

2576     <Info>Describes a virtual machine</Info>
2577     <Name>Virtual Appliance Two</Name>
2578     <ProductSection>
2579     <Info>Describes product information for the appliance</Info>
2580     <Product>The Great Appliance</Product>
2581     <Vendor>Some Great Corporation</Vendor>
2582     <Version>13.00</Version>
2583     <FullVersion>13.00-b5</FullVersion>
2584     <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2585     <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2586     <Property ovf:key="admin.email" ovf:type="string">
2587         <Description>Email address of administrator</Description>
2588     </Property>
2589     <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2590         <Description>The IP address of this appliance</Description>
2591     </Property>
2592 </ProductSection>
2593 <AnnotationSection ovf:required="false">
2594     <Info>A random annotation on this service. It can be ignored</Info>
2595     <Annotation>Contact customer support if you have any problems</Annotation>
2596 </AnnotationSection>
2597 <EulaSection>
2598     <Info>License information for the appliance</Info>
2599     <License>Insert your favorite license here</License>
2600 </EulaSection>
2601 <VirtualHardwareSection>
2602     <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2603     <Item>
2604         <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2605         <rasd:Description>Virtual CPU</rasd:Description>
2606         <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2607         <rasd:InstanceID>1</rasd:InstanceID>
2608         <rasd:Reservation>1</rasd:Reservation>
2609         <rasd:ResourceType>3</rasd:ResourceType>
2610         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2611     </Item>
2612     <Item>
2613         <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2614         <rasd:Description>Memory</rasd:Description>
2615         <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2616         <rasd:InstanceID>2</rasd:InstanceID>
2617         <rasd:ResourceType>4</rasd:ResourceType>
2618         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2619     </Item>
2620     <EthernetPortItem>
2621         <epasd:Address>00-16-8B-DB-00-5F</epasd:Address>
2622         <epasd:Connection>VM Network</epasd:Connection>
2623         <epasd:Description>Virtual NIC</epasd:Description>
2624
2625         <epasd:ElementName>Ethernet Port</epasd:ElementName>
2626         <!-- Virtual NIC for networking traffic -->
2627         <epasd:InstanceID>3</epasd:InstanceID>
2628
2629         <epasd:NetworkPortProfileID>http://www.dmtf.org/networkportprofiles/networkportprofile2.xml</
2630     epasd:NetworkPortProfileID>
2631         <epasd:NetworkPortProfileIDType>2</epasd:NetworkPortProfileIDType>
2632         <epasd:ResourceType>10</epasd:ResourceType>
2633         <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
2634     </EthernetPortItem>
2635     <StorageItem>
2636         <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2637         <sasd:Description>Virtual Disk</sasd:Description>
2638         <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2639         <sasd:InstanceID>4</sasd:InstanceID>
2640         <sasd:Reservation>100</sasd:Reservation>
2641         <sasd:ResourceType>31</sasd:ResourceType>
2642         <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
2643     </StorageItem>
2644 </VirtualHardwareSection>
2645 <OperatingSystemSection ovf:id="58" ovf:required="false">
2646     <Info>Guest Operating System</Info>

```

```

2647         <Description>OS</Description>
2648     </OperatingSystemSection>
2649 </VirtualSystem>
2650 </VirtualSystemCollection>
2651 </Envelope>

```

2652 D.5 Example 5 (networkportprofile1.xml)

2653
2654 Network Port profile example for bandwidth reservation.

```

2655 <?xml version="1.0" encoding="UTF-8"?>
2656 <NetworkPortProfile xsi:schemaLocation="http://schemas.dmtf.org/ovf/networkportprofile/1
2657 http://schemas.dmtf.org/ovf/networkportprofile/1/dsp8049.xsd"
2658 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2659 xmlns="http://schemas.dmtf.org/ovf/networkportprofile/1"
2660 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2661 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2662 schema/2/CIM_EthernetPortAllocationSettingData">
2663     <Item>
2664         <epasd:AllocationUnits>bit / second * 10^9</epasd:AllocationUnits>
2665         <epasd:ElementName>Network Port Profile 1</epasd:ElementName>
2666         <epasd:InstanceID>1</epasd:InstanceID>
2667         <epasd:NetworkPortProfileID>aaaaaaaa-bbbb-cccc-dddd-
2668 eeeeeeeeeeee</epasd:NetworkPortProfileID>
2669         <epasd:NetworkPortProfileIDType>3</epasd:NetworkPortProfileIDType>
2670         <epasd:Reservation>1</epasd:Reservation>
2671     </Item>
2672 </NetworkPortProfile>

```

2673 D.6 Example 6 (networkportprofile2.xml)

2674
2675 Network Port Profile example showing priority setting.

```

2676 <?xml version="1.0" encoding="UTF-8"?>
2677 <NetworkPortProfile xsi:schemaLocation="http://schemas.dmtf.org/ovf/networkportprofile/1
2678 http://schemas.dmtf.org/ovf/networkportprofile/1/dsp8049.xsd"
2679 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2680 xmlns="http://schemas.dmtf.org/ovf/networkportprofile/1"
2681 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2682 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2683 schema/2/CIM_EthernetPortAllocationSettingData">
2684     <Item>
2685         <epasd:AllowedPriorities>0</epasd:AllowedPriorities>
2686         <epasd:AllowedPriorities>1</epasd:AllowedPriorities>
2687         <epasd:DefaultPriority>0</epasd:DefaultPriority>
2688         <epasd:ElementName>Network Port Profile 2</epasd:ElementName>
2689         <epasd:InstanceID>2</epasd:InstanceID>
2690         <epasd:NetworkPortProfileID>aaaaaaaa-bbbb-cccc-dddd-
2691 ffffffff</epasd:NetworkPortProfileID>
2692         <epasd:NetworkPortProfileIDType>3</epasd:NetworkPortProfileIDType>
2693     </Item>
2694 </NetworkPortProfile>
2695

```

ANNEX E (informative)

Bibliography

- 2696
2697
2698
2699
- 2700 ISO 9660, *Joliet Extensions Specification*, May 1995,
2701 <http://bmrc.berkeley.edu/people/chaffee/jolspec.html>
- 2702 W3C, *Best Practices for XML Internationalization*, October 2008,
2703 <http://www.w3.org/TR/2008/NOTE-xml-i18n-bp-20080213/>
- 2704 DMTF DSP1044, *Processor Device Resource Virtualization Profile 1.0*
2705 http://www.dmtf.org/standards/published_documents/DSP1044_1.0.pdf
- 2706 DMTF DSP1045, *Memory Resource Virtualization Profile 1.0*
2707 http://www.dmtf.org/standards/published_documents/DSP1045_1.0.pdf
- 2708 DMTF DSP1047, *Storage Resource Virtualization Profile 1.0*
2709 http://www.dmtf.org/standards/published_documents/DSP1047_1.0.pdf
- 2710 DMTF DSP1022, *CPU Profile 1.0*,
2711 http://www.dmtf.org/standards/published_documents/DSP1022_1.0.pdf
- 2712 DMTF DSP1026, *System Memory Profile 1.0*,
2713 http://www.dmtf.org/standards/published_documents/DSP1026_1.0.pdf
- 2714 DMTF DSP1014, *Ethernet Port Profile 1.0*,
2715 http://www.dmtf.org/standards/published_documents/DSP1014_1.0.pdf
- 2716 DMTF DSP1050, *Ethernet Port Resource Virtualization Profile 1.1*
2717 http://www.dmtf.org/standards/published_documents/DSP1050_1.1.pdf
- 2718 DMTF DSP8049, *Network Port Profile XML Schema 1.0*
2719 http://schema.dmtf.org/ovf/networkportprofile/1/DSP8049_1.0.xsd
2720

ANNEX F (informative)

Change Log

2721
2722
2723
2724

Version	Date	Description
1.0.0	2009-02-22	DMTF Standard
1.1.0	2010-01-12	DMTF Standard
1.1.1	2010-06-01	Incorporate ANSI editor changes (wgv0.5.0)
	2010-06-23	Address Mantis 0000691 (wgv0.5.1)
	2010-06-24	Update POSIX reference to ISO/IEC/IEEE 9945:2009 (wgv0.6.0)
2.0.0a wgv 0.7.0	2011-06-28	Work in Progress release
2.0.0b wgv 0.9.0	2011-12-01	Work in Progress release candidate - Result of F2F, incorporated review comments, moved to Word 2010 & new template
2.0.0b wgv 0.9.1	2011-12-14	Work in Progress release candidate - Result of WG ballot Change 10.6 to Shishir's material, update list of contributors, added XML encryption to normative references
2.0.0 wgv 0.9.2	2012-5-18	NetworkSection and VirtualHardwareSection related section changes based on OVF 2 schema changes for network port profiles.
2.0.0 wgv 0.9.3	2012-05-24	Specs changes to reflect the new definitions of NetworkSection, VirtualHardwareSection, and ResourceAllocationSection.
2.0.0c wgv 0.9.4	2012-05-24	Work in Progress release candidate.
2.0.0 wgv 0.9.5	2012-08-24	Updated based on F2F WG discussions.
2.0.0 wgv 0.9.6	2012-08-29	Added Network Port Profile Annexes;
2.0.0 wgv 0.9.7	2012-09-13	Hemal update of Annex E descriptive text
2.0.0 wgv 0.9.8	2012-09-13	Added SHA reference
2.0.0d wgv 0.9.9	2012-09-20	Minor editorial fixes from ballot comments and include Mantis 1509 & 1510. Moved change log to end of file. Work in Progress release candidate

2725