

1 **Devices Profile for Web Services**

2 **May 2005**

3 **Co-Developers**

4 Shannon Chan, Microsoft

5 Chris Kaler, Microsoft

6 Thomas Kuehnel, Microsoft

7 Alain Regnier, Ricoh

8 Bryan Roe, Intel

9 Dale Sather, Microsoft

10 Jeffrey Schlimmer, Microsoft (Editor)

11 Hitoshi Sekine, Ricoh

12 Doug Walter, Microsoft

13 Jack Weast, Intel

14 Dave Whitehead, Lexmark

15 Don Wright, Lexmark

16 Yevgeniy Yarmosh, Intel

17 **Copyright Notice**

18 (c) 2004-2005 [Microsoft Corporation](#). All rights reserved.

19 Permission to copy and display the Devices Profile for Web Services (the "Profile"), in
20 any medium without fee or royalty is hereby granted, provided that you include the
21 following on ALL copies of the Profile, or portions thereof, that you make:

- 22 1. A link or URL to the Profile at this location.
- 23 2. The copyright notice as shown in the Profile.

24 THE PROFILE IS PROVIDED "AS IS," AND THE CO-DEVELOPERS MAKE NO
25 REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT
26 LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
27 PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE PROFILE
28 IS SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH
29 CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS,
30 TRADEMARKS OR OTHER RIGHTS.

31 THE CO-DEVELOPERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL,
32 INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO ANY
33 USE OR DISTRIBUTION OF THE PROFILE.

34 The name and trademarks of the Co-developers may NOT be used in any manner,
35 including advertising or publicity pertaining to the Profile or its contents without
36 specific, written prior permission. Title to copyright in the Profile will at all times
37 remain with Microsoft.

38 No other rights are granted by implication, estoppel or otherwise.

39	Abstract
40	This profile defines a minimal set of implementation constraints to enable secure
41	Web service messaging, discovery, description, and eventing on resource-
42	constrained endpoints.
43	Status
44	This is a public consultation draft release of this specification for community
45	evaluation and review. We welcome feedback on this specification through the WS-*
46	Workshop process.
47	Table of Contents
48	1. Introduction
49	1.1 Requirements
50	2. Terminology and Notation
51	2.1 Terminology
52	2.2 XML Namespaces
53	2.3 Notational Conventions
54	2.4 Compliance
55	3. Messaging
56	3.1 URI
57	3.2 UDP
58	3.3 HTTP
59	3.4 SOAP Envelope
60	3.5 WS-Addressing
61	3.6 Attachments
62	4. Discovery
63	5. Description
64	5.1 Characteristics
65	5.2 Hosting
66	5.3 WSDL
67	5.4 WS-Policy
68	6. Eventing
69	6.1 Subscription
70	6.2 Subscription Duration and Renewal
71	7. Security
72	7.1 Secure Communication
73	7.2 Security Policy Assertions

74 **8. Acknowledgements**
75 **9. References**
76 **10. Informative References**
77 **Appendix I – Constants**
78 **Appendix II – XML Schema**
79

80 **1. Introduction**

81 The Web services architecture includes a suite of specifications that define rich
82 functions and that may be composed to meet varied service requirements. To
83 promote both interoperability between resource-constrained Web service
84 implementations and interoperability with more flexible client implementations, this
85 profile identifies a core set of Web service specifications in the following areas:

- 86 • Sending secure messages to and from a Web service
- 87 • Dynamically discovering a Web service
- 88 • Describing a Web service
- 89 • Subscribing to, and receiving events from, a Web service

90 In each of these areas of scope, this profile defines minimal implementation
91 requirements for compliant Web service implementations.

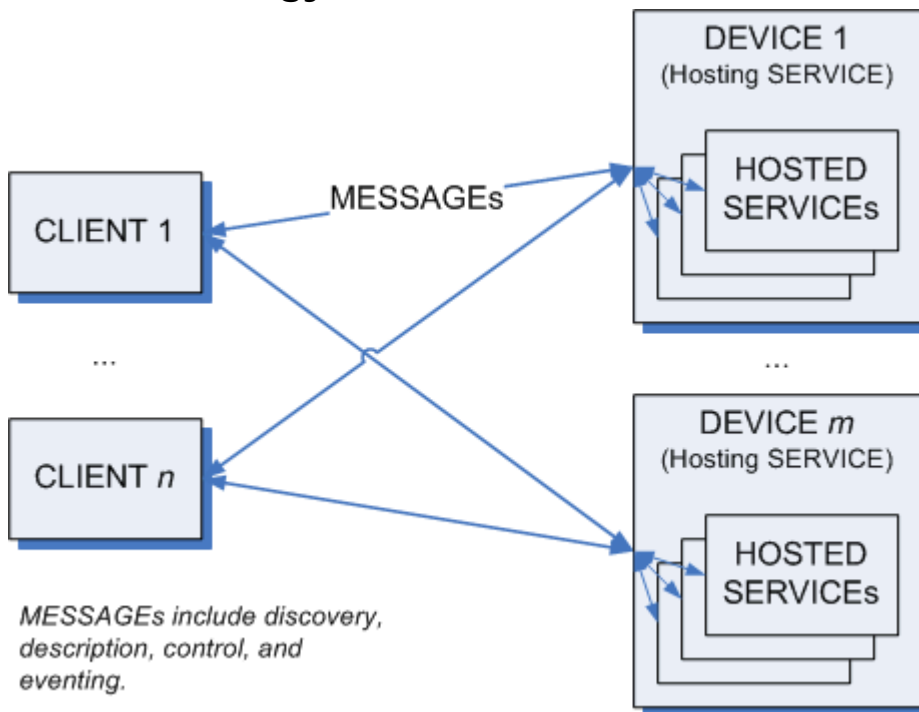
92 **1.1 Requirements**

93 This profile intends to meet the following requirements:

- 94 • Identify a minimal set of Web service specifications needed to enable secure
95 messaging, dynamic discovery, description, and eventing.
- 96 • Constrain Web services protocols and formats so Web services can be
97 implemented on peripheral-class and consumer electronics-class hardware.
- 98 • Define minimum requirements for compliance without constraining richer
99 implementations.

100 2. Terminology and Notation

101 2.1 Terminology



102

103 MESSAGE

104 Protocol elements that are exchanged, usually over a network, to affect a Web
105 service. Always includes a SOAP ENVELOPE. Typically also includes transport
106 framing information such as HTTP headers, TCP headers, and IP headers.

107 SOAP ENVELOPE

108 An XML Infoset that consists of a document information item [[XML Infoset](#)] with
109 exactly one member in its **[children]** property, which MUST be the SOAP
110 Envelope [[SOAP 1.2](#)] element information item.

111 MIME SOAP ENVELOPE

112 A SOAP ENVELOPE serialized using MIME Multipart Serialization [[MTOM](#)].

113 TEXT SOAP ENVELOPE

114 A SOAP ENVELOPE serialized as application/soap+xml.

115 CLIENT

116 A network endpoint that sends MESSAGEs to and/or receives MESSAGEs from a
117 SERVICE.

118 SERVICE

119 A network endpoint that receives and/or sends MESSAGEs to provide a service.

120 DEVICE

121 A distinguished type of SERVICE that hosts other SERVICES and sends and/or
122 receives one or more specific types of MESSAGEs.

123 HOSTED SERVICE

124 A distinguished type of SERVICE that is hosted by another SERVICE. The lifetime
125 of the HOSTED SERVICE is a subset of the lifetime of its host. The HOSTED

126 SERVICE is visible (not encapsulated) and is addressed separately from its host.
 127 Each HOSTED SERVICE has exactly one host. (The relationship is not transitive.)
 128 SENDER
 129 A CLIENT or SERVICE that sends a MESSAGE.
 130 RECEIVER
 131 A CLIENT or SERVICE that receives a MESSAGE.

132 2.2 XML Namespaces

133 The XML namespace URI that MUST be used by implementations of this specification
 134 is:

135 `http://schemas.xmlsoap.org/ws/2005/05/devprof`

136 Table 1 lists XML namespaces that are used in this specification. The choice of any
 137 namespace prefix is arbitrary and not semantically significant.

138 **Table 1: Prefixes and XML namespaces used in this specification.**

Prefix	XML Namespace	Specification(s)
soap	http://www.w3.org/2003/05/soap-envelope	[SOAP 1.2]
wsa	http://schemas.xmlsoap.org/ws/2004/08/addressing	[WS-Addressing]
wsd	http://schemas.xmlsoap.org/ws/2005/04/discovery	[WS-Discovery]
wsdp	http://schemas.xmlsoap.org/ws/2005/05/devprof	This profile
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL 1.1]
wse	http://schemas.xmlsoap.org/ws/2004/08/eventing	[WS-Eventing]
wsoap	http://schemas.xmlsoap.org/wsdl/soap12/	[WSDL Binding for SOAP 1.2]
wsp	http://schemas.xmlsoap.org/ws/2004/09/policy	[WS-Policy, WS-PolicyAttachment]
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd	[WS-Security 2004]
wsx	http://schemas.xmlsoap.org/ws/2004/09/mex	[WS-MetadataExchange]
xs	http://www.w3.org/2001/XMLSchema	[XML Schema Part 1, Part 2]

139 2.3 Notational Conventions

140 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
 141 "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
 142 document are to be interpreted as described in RFC 2119 [RFC 2119].

143 This specification uses the following syntax to define normative outlines for
 144 messages:

- 145 • The syntax appears as an XML instance, but values in italics indicate data types
 146 instead of literal values.
- 147 • Characters are appended to elements and attributes to indicate cardinality:

- 148 • "?" (0 or 1)
- 149 • "*" (0 or more)
- 150 • "+" (1 or more)
- 151 • The character "|" is used to indicate a choice between alternatives.
- 152 • The characters "(" and ")" are used to indicate that contained items are to be
- 153 treated as a group with respect to cardinality or choice.
- 154 • The characters "[" and "]" are used to call out references and property names.
- 155 • Ellipses (i.e., "...") indicate points of extensibility. Additional children and/or
- 156 attributes MAY be added at the indicated extension points but MUST NOT
- 157 contradict the semantics of the parent and/or owner, respectively. By default, if a
- 158 receiver does not recognize an extension, the receiver SHOULD ignore the
- 159 extension; exceptions to this processing rule, if any, are clearly indicated below.
- 160 • XML namespace prefixes (see [Table 1](#)) are used to indicate the namespace of the
- 161 element being defined.

162 This specification uses the **[action]** and Fault properties [[WS-Addressing](#)] to define

163 faults.

164 Normative statements in this profile are called out explicitly as follows:

165 *Rnnn: Normative statement text goes here.*

166 where "nnnn" is replaced by the statement number. Each statement contains exactly

167 one requirement level keyword (e.g., "MUST") and one conformance target keyword

168 (e.g., "MESSAGE").

169 2.4 Compliance

170 An endpoint MAY implement more than one of the roles defined herein. An endpoint

171 is not compliant with this specification if it fails to satisfy one or more of the MUST or

172 REQUIRED level requirements defined herein for the roles it implements.

173 Normative text within this specification takes precedence over normative outlines,

174 which in turn take precedence over the XML Schema [[XML Schema Part 1](#), [Part 2](#)]

175 descriptions, which in turn take precedence over examples.

176 3. Messaging

177 The scope of this section is the following set of Web services specifications. All of the

178 requirements in these specifications are included by reference except where

179 superseded by normative statements herein:

- 180 • [[SOAP 1.2, Part 1](#)]
- 181 • [[SOAP 1.2 Part 2, Section 7](#)]
- 182 • [[SOAP-over-UDP](#)]
- 183 • [[HTTP/1.1](#)]
- 184 • [[WS-Addressing](#)]
- 185 • [[MTOM](#)]

186 It is assumed that a DEVICE has obtained valid IPv4 and/or IPv6 addresses that do

187 not conflict with other addresses on the network. Mechanisms for obtaining IP

188 addresses are out of the scope of this profile. For more information, see [[DHCP](#)] and

189 [[IPv6 Autoconfig](#)].

190 **3.1 URI**

191 *R0025: A SERVICE MAY fail to process any URI with more than MAX_URI_SIZE*
192 *octets.*

193 *R0027: A SERVICE SHOULD NOT generate a URI with more than MAX_URI_SIZE*
194 *octets.*

195 The constant MAX_URI_SIZE is defined in [Appendix I – Constants](#).

196 **3.2 UDP**

197 *R0029: A SERVICE SHOULD NOT send a SOAP ENVELOPE that has more octets than*
198 *the MTU over UDP.*

199 To improve reliability, a SERVICE should minimize the size of SOAP ENVELOPEs sent
200 over UDP. However, some SOAP ENVELOPEs may be larger than an MTU; for
201 example, a signed Hello SOAP ENVELOPE. If a SOAP ENVELOPE is larger than an
202 MTU, the underlying IP network stacks may fragment and reassemble the UDP
203 packet.

204 **3.3 HTTP**

205 *R0001: A SERVICE MUST support transfer-coding = "chunked".*

206 *R0012: A SERVICE MUST at least support the SOAP HTTP Binding.*

207 *R0013: A SERVICE MUST at least implement the Responding SOAP Node of the SOAP*
208 *Request-Response Message Exchange Pattern*
209 *(<http://www.w3.org/2003/05/soap/mep/request-response/>).*

210 *R0014: A SERVICE MAY choose not to implement the Responding SOAP Node of the*
211 *SOAP Response Message Exchange Pattern*
212 *(<http://www.w3.org/2003/05/soap/mep/soap-response/>).*

213 *R0015: A SERVICE MAY choose not to support the SOAP Web Method Feature.*

214 R0014 and R0015 relax requirements in [\[SOAP 1.2, Part 2, Section 7\]](#).

215 *R0030: A SERVICE MUST at least implement the Responding SOAP Node of an HTTP*
216 *one-way Message Exchange Pattern where the SOAP ENVELOPE is carried in*
217 *the HTTP Request and the HTTP Response has a Status Code of 202 Accepted*
218 *and an empty Entity Body (no SOAP ENVELOPE).*

219 *R0017: A SERVICE MUST at least support Request Message SOAP ENVELOPEs and*
220 *one-way SOAP ENVELOPEs that are delivered using HTTP POST.*

221 **3.4 SOAP Envelope**

222 *R0034: A SERVICE MUST at least receive and send SOAP 1.2 [\[SOAP 1.2\]](#) SOAP*
223 *ENVELOPEs.*

224 *R0003: A SERVICE MAY reject a TEXT SOAP ENVELOPE with more than*
225 *MAX_ENVELOPE_SIZE octets.*

226 *R0026: A SERVICE SHOULD NOT send a TEXT SOAP ENVELOPE with more than*
227 *MAX_ENVELOPE_SIZE octets.*

228 Large SOAP ENVELOPEs are expected to be serialized using attachments.

229 3.5 WS-Addressing

230 *R0004: A DEVICE SHOULD use a uuid scheme URI as the **[address]** property of its*
231 *Endpoint Reference.*

232 *R0005: A DEVICE MUST use a stable, globally unique identifier that is constant*
233 *across network interfaces and IPv4/v6 addresses as the **[address]** property*
234 *of its Endpoint Reference.*

235 *R0006: A DEVICE MUST persist the **[address]** property of its Endpoint Reference*
236 *across re-initialization and changes in the metadata of the DEVICE and any*
237 *SERVICES it hosts.*

238 Because the **[address]** property of an Endpoint Reference [[WS-Addressing](#)] is a
239 SOAP-layer address, there is no requirement to use anything other than a UUID for
240 the **[address]** property.

241 *R0007: A DEVICE SHOULD NOT include any **[reference property]** properties in its*
242 *Endpoint Reference.*

243 The combination of the **[address]** and **[reference property]** properties defines the
244 identity of an Endpoint Reference. If the **[address]** property provides sufficient
245 identity information, there is no requirement to use **[reference property]**
246 properties to provide additional identity.

247 *R0042: A HOSTED SERVICE SHOULD use an HTTP transport address as the*
248 ***[address]** property of its Endpoint Reference.*

249 Use of other possible values of **[address]** by a HOSTED SERVICE is out of scope of
250 this profile.

251 *R0031: A SERVICE MAY reject an HTTP Request Message SOAP ENVELOPE if the*
252 ***[address]** of the **[reply endpoint]** is not*
253 *"http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous".*

254 *R0041: If an HTTP Request Message SOAP ENVELOPE generates a SOAP Fault, a*
255 *SERVICE MAY discard the SOAP Fault if the **[address]** of the **[fault***
256 ***endpoint]** of the HTTP Request Message is not*
257 *"http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous".*

258 The SOAP HTTP Binding requires the Response Message SOAP ENVELOPE to be
259 transmitted as the HTTP Response of the corresponding Request Message SOAP
260 ENVELOPE.

261 *R0019: A SERVICE MUST include a Message Information Header representing a*
262 ***[relationship]** property of type `wsa:Reply` in each Response Message SOAP*
263 *ENVELOPE the service generates.*

264 *R0040: A SERVICE MUST include a Message Information Header representing a*
265 ***[relationship]** property of type `wsa:Reply` in each SOAP Fault SOAP*
266 *ENVELOPE the service generates.*

267 3.6 Attachments

268 *R0022: If a SERVICE supports attachments, the SERVICE MUST support the HTTP*
269 *Transmission Optimization Feature.*

270 The HTTP Transmission Optimization Feature implies support for the Optimized MIME
271 Multipart Serialization and Abstract Transmission Optimization features.

272 *R0036: A SERVICE MAY reject a MIME SOAP ENVELOPE if the Content-Transfer-*
273 *Encoding header field mechanism of any MIME part is "binary".*

274 *R0037: A SERVICE MUST NOT send a MIME SOAP ENVELOPE unless the Content-*
275 *Transfer-Encoding header field mechanism of every MIME part is "binary".*

276 Even for the SOAP Envelope, the "binary" Content-Transfer-Encoding mechanism is
277 more appropriate than the "8bit" mechanism which is suitable only for data that may
278 be represented as relatively short lines of at most 998 octets [MIME].

279 *R0038: A SERVICE MAY reject a MIME SOAP ENVELOPE if the root part is not the first*
280 *body part in the Multipart/Related entity.*

281 *R0039: A SERVICE MUST NOT send a MIME SOAP ENVELOPE unless root part is the*
282 *first body part in the Multipart/Related entity.*

283 Per MTOM, the root part of the MIME SOAP ENVELOPE contains an XML
284 representation of the modified SOAP Envelope, with additional parts that contain
285 binary representations of each attachment. This root part must be the first part so a
286 RECEIVER does not have to buffer attachments.

287 **4. Discovery**

288 The scope of this section is the following set of Web services specifications. All of the
289 requirements in these specifications are included by reference except where
290 superseded by normative statements herein:

- 291 • [WS-Discovery]

292 If a CLIENT and a SERVICE are not on the same subnet, the CLIENT may not be able
293 to discover the SERVICE. However, if a CLIENT has an Endpoint Reference and
294 transport address for a SERVICE through some other means, the CLIENT and
295 SERVICE should be able to communicate within the scope of this profile.

296 *R1013: A DEVICE MUST be a compliant Target Service.*

297 *R1001: A HOSTED SERVICE SHOULD NOT be a Target Service.*

298 If each SERVICE were to participate in WS-Discovery, the network traffic generated
299 by a relatively small number of DEVICES hosting a relatively small number of
300 HOSTED SERVICES could overwhelm a bandwidth-limited network. Therefore, only
301 DEVICES act as Target Services.

302 *R1019: A DEVICE MUST at least support the*
303 *"http://schemas.xmlsoap.org/ws/2005/04/discovery/rfc2396" and*
304 *"http://schemas.xmlsoap.org/ws/2005/04/discovery/strcmp0" Scope*
305 *matching rules.*

306 *R1020: If a DEVICE includes Types in a Hello, Probe Match, or Resolve Match SOAP*
307 *ENVELOPE, it MUST include the wsx:MetadataExchange Type.*

308 Including the wsx:MetadataExchange Type indicates a DEVICE supports Get
309 Metadata [WS-MetadataExchange] which is the interoperable means for retrieving
310 metadata about the DEVICE and any HOSTED SERVICES.

311 *R1009: A DEVICE MUST at least support receiving Probe and Resolve SOAP*
312 *ENVELOPEs and sending Hello and Bye SOAP ENVELOPEs over multicast UDP.*

313 *R1016: A DEVICE MUST at least support sending Probe Match and Resolve Match*
314 *SOAP ENVELOPEs over unicast UDP.*

315 *R1018: A DEVICE MAY ignore a multicast UDP Probe or Resolve SOAP ENVELOPE if*
316 *the [address] of the [reply endpoint] is not*
317 *"http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous".*

318 WS-Discovery acknowledges that a CLIENT may include reply information in UDP
319 Probe and Resolve SOAP ENVELOPEs to specify a transport other than SOAP over
320 UDP. However, to establish a baseline for interoperability, DEVICES are required only
321 to support UDP responses.

322 *R1015: A DEVICE MUST support receiving a Probe SOAP ENVELOPE as an HTTP*
323 *Request.*

324 *R1021: If a DEVICE matches a Probe SOAP ENVELOPE received as an HTTP Request,*
325 *it MUST send a Probe Match SOAP ENVELOPE in the HTTP Response.*

326 *R1022: If a DEVICE does not match a Probe SOAP ENVELOPE received as an HTTP*
327 *Request, it MUST send an HTTP Response with a Status Code of 202 Accepted*
328 *and an empty Entity Body (no SOAP ENVELOPE).*

329 To support the scenario where a DEVICE has a known HTTP address, a CLIENT may
330 send a Probe over HTTP to that address and expect to receive either a Probe Match
331 (if the Probe matches the DEVICE listening on that address) or an empty HTTP
332 Response (otherwise).

333 5. Description

334 The scope of this section is the following set of Web services specifications. All of the
335 requirements in these specifications are included by reference except where
336 superseded by normative statements herein:

- 337 • [\[XML Schema Part 1, Part 2\]](#)
- 338 • [\[WSDL 1.1\]](#)
- 339 • [\[BP 1.1, Section 4\]](#)
- 340 • [\[WSDL Binding for SOAP 1.2\]](#)
- 341 • [\[WS-MetadataExchange\]](#)
- 342 • [\[WS-Policy\]](#)
- 343 • [\[WS-PolicyAttachment\]](#)

344 In highly-constrained circumstances, a CLIENT will know all it needs to know about a
345 DEVICE and its HOSTED SERVICES to correctly send and receive application-specific
346 MESSAGES. However, in development scenarios, or when a CLIENT wishes to inspect
347 a DEVICE and take advantage of extended or nonstandard capabilities, a CLIENT will
348 need to retrieve the description (a.k.a. metadata) for a DEVICE and/or its HOSTED
349 SERVICES.

350 The description for a DEVICE is retrieved by sending a Get Metadata SOAP
351 ENVELOPE to the DEVICE. The description conveys generic DEVICE characteristics
352 and may be extended to convey domain-specific SERVICE characteristics. Description
353 also indicates which HOSTED SERVICES are hosted by a DEVICE; in many
354 circumstances, a CLIENT will need to retrieve the description for one or more
355 HOSTED SERVICES as well as for the DEVICE.

356 Through WSDL, description also conveys the MESSAGES a SERVICE is capable of
357 receiving and sending. Through WS-Policy, description conveys the capabilities and

358 requirements of a SERVICE, particularly the transports over which it may be reached
359 and its security capabilities.

360 5.1 Characteristics

361 To express DEVICE characteristics that are typically fixed across all DEVICES of the
362 same model by their manufacturer, this profile defines extensible ThisModel
363 metadata as follows:

```
364 <wsdp:ThisModel ...>  
365   <wsdp:Manufacturer xml:lang="..."? >xs:string</wsdp:Manufacturer>+  
366   <wsdp:ManufacturerUrl>xs:anyURI</wsdp:ManufacturerUrl?>  
367   <wsdp:ModelName xml:lang="..."? >xs:string</wsdp:ModelName>+  
368   <wsdp:ModelNumber>xs:string</wsdp:ModelNumber?>  
369   <wsdp:ModelUrl>xs:anyURI</wsdp:ModelUrl?>  
370   <wsdp:PresentationUrl>xs:anyURI</wsdp:PresentationUrl?>  
371   ...  
372 </wsdp:ThisModel>
```

373 The following describes additional, normative constraints on the outline above:

374 wsdp:ThisModel/ wsdp:Manufacturer

375 Name of the manufacturer of the DEVICE. It MUST have fewer than
376 MAX_FIELD_SIZE Unicode characters, SHOULD be localized, and SHOULD be
377 repeated for each supported locale.

378 wsdp:ThisModel/ wsdp:ManufacturerUrl

379 URL to a Web site for the manufacturer of the DEVICE. It MUST have fewer than
380 MAX_URI_SIZE octets.

381 wsdp:ThisModel/ wsdp:ModelName

382 User-friendly name for this model of device chosen by the manufacturer. It MUST
383 have fewer than MAX_FIELD_SIZE Unicode characters, SHOULD be localized, and
384 SHOULD be repeated for each supported locale.

385 wsdp:ThisModel/ wsdp:ModelNumber

386 Model number for this model of DEVICE. It MUST have fewer than
387 MAX_FIELD_SIZE Unicode characters.

388 wsdp:ThisModel/ wsdp:ModelUrl

389 URL to a Web site for this model of DEVICE. It MUST have fewer than
390 MAX_URI_SIZE octets.

391 wsdp:ThisModel/ wsdp:PresentationUrl

392 URL to an HTML page for this DEVICE. It MAY be relative to a base URL and MUST
393 have fewer than MAX_URI_SIZE octets.

394 CORRECT:

```
395 <wsdp:ThisModel  
396   xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof" >  
397   <wsdp:Manufacturer>ACME Manufacturing</wsdp:Manufacturer>  
398   <wsdp:ModelName xml:lang="en-GB" >ColourBeam 9</wsdp:ModelName>  
399   <wsdp:ModelName xml:lang="en-US" >ColorBeam 9</wsdp:ModelName>  
400 </wsdp:ThisModel>
```

402 A Dialect [[WS-MetadadataExchange](#)] equal to

403 "http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisModel" indicates an instance
404 of the ThisModel metadata format.

405 No Identifier [[WS-MetadataExchange](#)] is defined for instances of the ThisModel
406 metadata format.

407 *R2038: A DEVICE MUST have one Metadata Section with Dialect equal to*
408 *"http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisModel" for its*
409 *ThisModel metadata.*

410 *R2012: If no Dialect is specified in a Get Metadata SOAP ENVELOPE, in the*
411 *corresponding Get Metadata Response SOAP ENVELOPE, a DEVICE MUST*
412 *include the Metadata Section with Dialect equal to*
413 *"http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisModel".*

414 Get Metadata [[WS-MetadataExchange](#)] is the interoperable means for a CLIENT to
415 retrieve the ThisModel metadata for a DEVICE. A DEVICE may also provide other
416 means for a CLIENT to retrieve its ThisModel metadata.

417 *R2001: If a DEVICE changes any of its ThisModel metadata, it MUST increment the*
418 *Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP*
419 *ENVELOPEs as wsdl:MetadataVersion.*

420 Caching for the ThisModel metadata is controlled by the wsdl:MetadataVersion
421 construct [[WS-Discovery](#)].

422 To express DEVICE characteristics that typically vary from one DEVICE to another of
423 the same kind, this profile defines extensible ThisDevice metadata as follows:

```
424 <wsdp:ThisDevice ...>  
425   <wsdp:FriendlyName xml:lang="..."? >xs:string</wsdp:FriendlyName>+  
426   <wsdp:FirmwareVersion>xs:string</wsdp:FirmwareVersion?>  
427   <wsdp:SerialNumber>xs:string</wsdp:SerialNumber?>  
428   ...  
429 </wsdp:ThisDevice>
```

430 The following describes additional, normative constraints on the outline above:

431 wsdl:ThisDevice/ wsdl:FriendlyName
432 User-friendly name for this DEVICE. It MUST have fewer than MAX_FIELD_SIZE
433 Unicode characters, SHOULD be localized, and SHOULD be repeated for each
434 supported locale.

435 wsdl:ThisDevice/ wsdl:FirmwareVersion
436 Firmware version for this DEVICE. It MUST have fewer than MAX_FIELD_SIZE
437 Unicode characters.

438 wsdl:ThisDevice/ wsdl:SerialNumber
439 Manufacturer-assigned serial number for this DEVICE. It MUST have fewer than
440 MAX_FIELD_SIZE Unicode characters.

441 CORRECT:

```
442 <wsdp:ThisDevice  
443   xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof" >  
444   <wsdp:FriendlyName xml:lang="en-GB" >  
445     ACME ColourBeam Printer  
446   </wsdp:FriendlyName>  
447   <wsdp:FriendlyName xml:lang="en-US" >  
448     ACME ColorBeam Printer  
449   </wsdp:FriendlyName>  
450 </wsdp:ThisDevice>  
451
```

452 A Dialect [[WS-MetadadataExchange](#)] equal to
453 "http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisDevice" indicates an instance
454 of the ThisDevice metadata format.
455 No Identifier [[WS-MetadadataExchange](#)] is defined for instances of the ThisDevice
456 metadata format.

457 *R2039: A DEVICE MUST have a Metadata Section with Dialect equal to*
458 *"http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisDevice" for its*
459 *ThisDevice metadata.*

460 *R2014: If no Dialect is specified in a Get Metadata SOAP ENVELOPE, in the*
461 *corresponding Get Metadata Response SOAP ENVELOPE, a DEVICE MUST*
462 *include the Metadata Section with Dialect equal to*
463 *"http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisDevice".*

464 CORRECT:

```
465 <soap:Envelope
466     xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
467     xmlns:wsp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
468     xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
469     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
470 <soap:Header>
471     <wsa:Action>
472         http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata/Response
473     </wsa:Action>
474     <wsa:RelatesTo>
475         uuid:82204a83-52f6-475c-9708-174fa27659ec
476     </wsa:RelatesTo>
477     <wsa:To>
478         http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
479     </wsa:To>
480 </soap:Header>
481 <soap:Body>
482     <wsx:Metadata>
483
484         <wsx:MetadataSection
485 Dialect="http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisModel"
486         >
487             <wsp:ThisModel>
488                 <wsp:Manufacturer>ACME Manufacturing</wsp:Manufacturer>
489                 <wsp:ModelName xml:lang="en-GB" >
490                     ColourBeam 9
491                 </wsp:ModelName>
492                 <wsp:ModelName xml:lang="en-US" >
493                     ColorBeam 9
494                 </wsp:ModelName>
495             </wsp:ThisModel>
496         </wsx:MetadataSection>
497
498         <wsx:MetadataSection
499 Dialect="http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisDevice"
500         >
501             <wsp:ThisDevice>
502                 <wsp:FriendlyName xml:lang="en-GB" >
503                     ACME ColourBeam Printer
504                 </wsp:FriendlyName>
```

```

505     <wsdp:FriendlyName xml:lang="en-US" >
506         ACME ColorBeam Printer
507     </wsdp:FriendlyName>
508 </wsdp:ThisDevice>
509 </wsx:MetadataSection>
510
511     <!-- Other Metadata Sections omitted for brevity. -->
512
513 </wsx:Metadata>
514 </soap:Body>
515 </soap:Envelope>
516

```

517 Get Metadata [[WS-MetadataExchange](#)] is the interoperable means for a CLIENT to
518 retrieve the ThisDevice metadata for a DEVICE. A DEVICE may also provide other
519 means for a CLIENT to retrieve its ThisDevice metadata.

520 *R2002: If a DEVICE changes any of its ThisDevice metadata, it MUST increment the*
521 *Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP*
522 *ENVELOPES as wsdl:MetadataVersion.*

523 Caching for the ThisDevice metadata is controlled by the wsdl:MetadataVersion
524 construct [[WS-Discovery](#)].

525 5.2 Hosting

526 To express the relationship between a HOSTED SERVICE and its host, this profile
527 defines relationship metadata as follows:

```

528 <wsdp:Relationship Type="xs:anyURI" ... >
529   (<wsdp:Host>
530     <wsa:EndpointReference>endpoint-reference</wsa:EndpointReference>
531     <wsdp:Types>list of xs:QName</wsdp:Types>?
532     <wsdp:ServiceId>xs:anyURI</wsdp:ServiceId>?
533     ...
534   </wsdp:Host>)?
535   (<wsdp:Hosted>
536     <wsa:EndpointReference>endpoint-reference</wsa:EndpointReference>
537     <wsdp:Types>list of xs:QName</wsdp:Types>?
538     <wsdp:ServiceId>xs:anyURI</wsdp:ServiceId>?
539     ...
540   </wsdp:Hosted>)*
541   ...
542 </wsdp:Relationship>

```

543 The following describes additional, normative constraints on the outline above:

544 wsdl:Relationship
545 Relationship between two or more SERVICES.

546 wsdl:Relationship/@Type
547 The type of the relationship. This value should be compared directly, as a case-
548 sensitive string, with no attempt to make a relative URI into an absolute URI, to
549 unescape, or to otherwise canonicalize it.

550 wsdl:Relationship/@Type =
551 "http://schemas.xmlsoap.org/ws/2005/05/devprof/host"
552 Hosting relationship between a HOSTED SERVICE its host. This relationship type
553 defines the following additional content:

554 wsdp:Relationship/wsdp:Host
555 Endpoint Reference for the host. If omitted, implied value is the Endpoint
556 Reference of the SERVICE that returned this metadata in a Get Metadata
557 Response SOAP ENVELOPE. At least one of ./wsdp:Host or ./wsdp:Hosted MUST
558 be included.

559 wsdp:Relationship/wsdp:Host/wsdp:Types
560 Unordered set of Types implemented by the host. (See [\[WS-Discovery\]](#).) If
561 omitted, no implied value.

562 The Types element is explicitly copied from the WS-Discovery XML namespace
563 into this one to make the XML Schema deterministic. Reusing the wsdp:Types
564 element from WS-Discovery would introduce non-determinism because there
565 would be an optional element from another XML namespace (wsdp:Types),
566 followed by an optional element (wsdp:ServiceId) and an optional wildcard for
567 elements from other XML namespaces.

568 wsdp:Relationship/wsdp:Host/wsdp:ServiceId
569 Identifier for the host which MUST be persisted across re-initialization (see also
570 [R0005](#) and [R0006](#)) and MAY be shared across multiple Host elements if a host
571 has more than one Endpoint Reference. This value should be compared directly,
572 as a case-sensitive string, with no attempt to make a relative URI into an
573 absolute URI, to unescape, or to otherwise canonicalize it.

574 wsdp:Relationship/wsdp:Hosted
575 Endpoint Reference for the HOSTED SERVICE. If omitted, implied value is the
576 Endpoint Reference of the SERVICE that returned this metadata in a Get
577 Metadata Response SOAP ENVELOPE. At least one of ./wsdp:Host or
578 ./wsdp:Hosted MUST be included.

579 wsdp:Relationship/wsdp:Hosted/wsdp:Types
580 Unordered set of Types implemented by the HOSTED SERVICE. (See [\[WS-](#)
581 [Discovery\]](#).) If omitted, no implied value.

582 wsdp:Relationship/wsdp:Hosted/wsdp:ServiceId
583 Identifier for the HOSTED SERVICE which MUST be persisted across re-
584 initialization and MAY be shared across multiple Hosted elements if a HOSTED
585 SERVICE has more than one Endpoint Reference. This identifier allows a CLIENT
586 to recognize which Endpoint References refer to the same HOSTED SERVICE. This
587 value should be compared directly, as a case-sensitive string, with no attempt to
588 make a relative URI into an absolute URI, to unescape, or to otherwise
589 canonicalize it.

590 CORRECT:

```

591 <wsdp:Relationship
592    Type="http://schemas.xmlsoap.org/ws/2005/05/devprof/host"
593    xmlns:img="http://printer.example.org/imaging"
594    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
595    xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof" >
596 <wsdp:Hosted>
597    <wsa:EndpointReference>
598      <wsa:Address>http://172.30.184.244/print</wsa:Address>
599    </wsa:EndpointReference>
600    <wsdp:Types>
601      img:PrintBasicPortType img:PrintAdvancedPortType
602    </wsdp:Types>

```

```
603 <wsdp:ServiceId>
604   http://printer.example.org/imaging/PrintService
605 </wsdp:ServiceId>
606 </wsdp:Hosted>
607 </wsdp:Relationship>
608
```

609 A Dialect [[WS-MetadataExchange](#)] equal to
610 "http://schemas.xmlsoap.org/ws/2005/05/devprof/Relationship" indicates an
611 instance of the Relationship metadata format.

612 No Identifier [[WS-MetadataExchange](#)] is defined for instances of the Relationship
613 metadata format.

614 *R2040: If a SERVICE has any HOSTED SERVICES, it MUST have at least one*
615 *Metadata Section with Dialect equal to*
616 *"http://schemas.xmlsoap.org/ws/2005/05/devprof/Relationship" for its*
617 *Relationship metadata.*

618 *R2029: If no Dialect is specified in a Get Metadata SOAP ENVELOPE, in the*
619 *corresponding Get Metadata Response SOAP ENVELOPE, a SERVICE MUST*
620 *include the Metadata Section(s) with Dialect equal to*
621 *"http://schemas.xmlsoap.org/ws/2005/05/devprof/Relationship".*

622 Get Metadata [[WS-MetadataExchange](#)] is the interoperable means for a CLIENT to
623 retrieve the relationship metadata for a SERVICE. A SERVICE may provide other
624 means for a CLIENT to retrieve its relationship metadata.

625 CORRECT:

```
626 <soap:Envelope
627   xmlns:gen="http://example.org/general"
628   xmlns:img="http://printer.example.org/imaging"
629   xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
630   xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
631   xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
632   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
633 <soap:Header>
634   <wsa:Action>
635     http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata/Response
636   </wsa:Action>
637   <wsa:RelatesTo>
638     uuid:82204a83-52f6-475c-9708-174fa27659ec
639   </wsa:RelatesTo>
640   <wsa:To>
641     http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
642   </wsa:To>
643 </soap:Header>
644 <soap:Body>
645   <wsx:Metadata>
646     <wsx:MetadataSection
647       Dialect="http://schemas.xmlsoap.org/ws/2005/05/devprof/Relationship"
648     >
649       <wsdp:Relationship
650         Type="http://schemas.xmlsoap.org/ws/2005/05/devprof/host" >
651         <wsdp:Hosted>
652           <wsa:EndpointReference>
653             <wsa:Address>http://172.30.184.244/print</wsa:Address>
654           </wsa:EndpointReference>
```



```

655     <wsdp:Types>
656         img:PrintBasicPortType img:PrintAdvancedPortType
657     </wsdp:Types>
658     <wsdp:ServiceId>
659         http://printer.example.org/imaging/PrintService
660     </wsdp:ServiceId>
661 </wsdp:Hosted>
662 <wsdp:Hosted>
663     <wsa:EndpointReference>
664         <wsa:Address>http://[fdaa:23]/print1</wsa:Address>
665     </wsa:EndpointReference>
666     <wsdp:Types>
667         img:PrintBasicPortType img:PrintAdvancedPortType
668     </wsdp:Types>
669     <wsdp:ServiceId>
670         http://printer.example.org/imaging/PrintService
671     </wsdp:ServiceId>
672 </wsdp:Hosted>
673
674 <wsdp:Hosted>
675     <wsa:EndpointReference>
676         <wsa:Address>http://172.30.184.244/scan</wsa:Address>
677     </wsa:EndpointReference>
678     <wsdp:Types>img:ScanBasicPortType</wsdp:Types>
679     <wsdp:ServiceId>
680         http://printer.example.org/imaging/ScanService
681     </wsdp:ServiceId>
682 </wsdp:Hosted>
683 <wsdp:Hosted>
684     <wsa:EndpointReference>
685         <wsa:Address>http://[fdaa:24]/scan</wsa:Address>
686     </wsa:EndpointReference>
687     <wsdp:Types>img:ScanBasicPortType</wsdp:Types>
688     <wsdp:ServiceId>
689         http://printer.example.org/imaging/ScanService
690     </wsdp:ServiceId>
691 </wsdp:Hosted>
692 </wsdp:Relationship>
693 </wsx:MetadataSection>
694
695     <!-- Other Metadata Sections omitted for brevity. -->
696
697 </wsx:Metadata>
698 </soap:Body>
699 </soap:Envelope>
700

```

701 *R2030: If a DEVICE changes any of its relationship metadata, it MUST increment the*
702 *Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP*
703 *ENVELOPES as wsdl:MetadataVersion.*

704 Caching for relationship metadata is controlled by the wsdl:MetadataVersion
705 construct [[WS-Discovery](#)].

706 *R2042: A DEVICE MUST NOT change its relationship metadata based on temporary*
707 *changes in the network availability of the SERVICES described by the*
708 *metadata.*

709 Relationship metadata is intended to model fairly static relationships and should not
710 change if a SERVICE becomes temporarily unavailable. As in the general case, any
711 CLIENT attempting to contact such a SERVICE will need to deal with an Endpoint
712 Unavailable Fault [[WS-Addressing](#)], connection refusal, or other network indication
713 that the SERVICE is unavailable.

714 5.3 WSDL

715 *R2004: If a SERVICE exposes Notifications, its portType MUST include Notification*
716 *and/or Solicit-Response Operations describing those Notifications.*

717 R2004 relaxes R2303 in [[BP 1.1, Section 4](#)].

718 *R2019: A SERVICE MUST at least include a document-literal Binding for each*
719 *portType in its WSDL.*

720 Because the document-literal SOAP Binding is more general than an rpc-literal SOAP
721 Binding, there is no requirement to use anything other than the document-literal
722 Binding.

723 *R2020: A SERVICE MUST at least include a WSDL Binding for SOAP 1.2 for each*
724 *portType in its WSDL.*

725 *R2028: A SERVICE is not required to include any WSDL bindings for SOAP 1.1 in its*
726 *WSDL.*

727 Since this profile brings SOAP 1.2 into scope, it is sufficient to bind to that version of
728 SOAP. There is no requirement to bind to other SOAP versions and thus R2028
729 updates R2401 in [[BP 1.1, Section 4](#)] to SOAP 1.2.

730 *R2023: If a SERVICE receives a MESSAGE that is inconsistent with its WSDL*
731 *description, the SERVICE SHOULD generate a SOAP Fault with a Code Value*
732 *of "Sender", unless a "MustUnderstand" or "VersionMismatch" Fault is*
733 *generated.*

734 *R2024: If a SERVICE receives a MESSAGE that is inconsistent with its WSDL*
735 *description, it MUST check for "VersionMismatch", "MustUnderstand", and*
736 *"Sender" fault conditions in that order.*

737 Statements R2023 and R2024 update R2724 and R2725 [[BP 1.1, Section 4](#)] to SOAP
738 1.2.

739 *R2031: A SERVICE MUST have at least one Metadata Section with*
740 *Dialect="http://schemas.xmlsoap.org/wsdl/".*

741 For clarity, separation of levels of abstraction, and/or reuse of standardized
742 components, WSDL may be authored in a style that separates different elements of a
743 Service Definition into separate documents which may be imported as needed. Each
744 separate document is included in separate WSDL Metadata Sections.

745 *R2016: If no Dialect is specified in a Get Metadata SOAP ENVELOPE, in the*
746 *corresponding Get Metadata Response SOAP ENVELOPE, a SERVICE MUST*
747 *include the Metadata Section(s) with Dialect equal to*
748 *"http://schemas.xmlsoap.org/wsdl/".*

749 Get Metadata [[WS-MetadataExchange](#)] is the interoperable means for a CLIENT to
750 retrieve the WSDL for a HOSTED SERVICE. A HOSTED SERVICE may provide other
751 means for a CLIENT to retrieve its WSDL.

752 There is no requirement for a SERVICE to store its WSDL and include it in-line in a
753 Get Metadata Response SOAP ENVELOPE. The WSDL may be stored at a different
754 location, and the SERVICE may include a reference to it in a Get Metadata Response
755 SOAP ENVELOPE.

756 CORRECT:

```
757 <soap:Envelope
758     xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
759     xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
760     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
761   <soap:Header>
762     <wsa:Action>
763       http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata/Response
764     </wsa:Action>
765     <wsa:RelatesTo>
766       uuid:82204a83-52f6-475c-9708-174fa27659ec
767     </wsa:RelatesTo>
768     <wsa:To>
769       http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
770     </wsa:To>
771   </soap:Header>
772   <soap:Body>
773     <wsx:Metadata>
774
775       <wsx:MetadataSection
776         Dialect="http://schemas.xmlsoap.org/wSDL" >
777         <wsx:MetadataReference>
778           <wsa:Address>http://172.30.184.244/print</wsa:Address>
779           <wsa:ReferenceParameters>
780             <x:Acme xmlns:x="urn:acme.com:webservicess">
781               WSDL
782             </x:Acme>
783           </wsa:ReferenceParameters>
784         </wsx:MetadataReference>
785       </wsx:MetadataSection>
786
787       <!-- Other Metadata Sections omitted for brevity. -->
788
789     </wsx:Metadata>
790   </soap:Body>
791 </soap:Envelope>
792
```

793 *R2021: If a DEVICE changes its WSDL, it MUST increment the Metadata Version*
794 *exposed in Hello, Probe Match, and Resolve Match SOAP ENVELOPEs as*
795 *wsd:MetadataVersion.*

796 Caching for DEVICE WSDL is controlled by the wsd:MetadataVersion construct [[WS-](#)
797 [Discovery](#)]. Since a HOSTED SERVICE should not be a Target Service, any changes
798 to its WSDL should not be controlled by the wsd:MetadataVersion construct.

799 5.4 WS-Policy

800 To indicate that a DEVICE is compliant with this profile, this profile defines the
801 following WS-Policy [[WS-Policy](#)] assertion:

```
802 <wsp:Profile wsp:Optional="true"? ... />
```

803 The following describes additional, normative constraints on the outline above:

804 wsdp:Profile

805 Assertion indicating compliance with this profile is required. This assertion has
806 Endpoint Policy Subject [[WS-PolicyAttachment](#)]: a policy expression containing
807 this assertion MAY be attached to a wsdl:port, SHOULD be attached to a
808 wsdl:binding, but MUST NOT be attached to a wsdl:portType; the latter is
809 prohibited because the assertion specifies a concrete behavior whereas the
810 wsdl:portType is an abstract construct.

811 wsdp:Profile/@wsp:Optional="true"

812 Per WS-Policy [[WS-Policy](#)], this is compact notation for two policy alternatives,
813 one with and one without the assertion. The intuition is that the behavior
814 indicated by the assertion is optional, or in this case, that the SERVICE supports
815 but does not require compliance with this profile.

816 CORRECT:

```
817 <wsp:Policy
818     xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
819     xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" >
820   <wsdp:Profile />
821 </wsp:Policy>
```

823 *R2037: A SERVICE MUST include the wsdp:Profile assertion in its policy.*

824 To indicate that a SERVICE sends or receives MIME SOAP ENVELOPES, this profile
825 defines the following WS-Policy assertion:

```
826 <wsdp:OptimizedMimeSerialization wsp:Optional="true"? ... />
```

827 The following describes additional, normative constraints on the outline above:

828 wsdp:OptimizedMimeSerialization

829 A SOAP ENVELOPE MUST be serialized as a MIME SOAP ENVELOPE [[MTOM](#)]. This
830 assertion has Endpoint Policy Subject: a policy expression containing this
831 assertion MAY be attached to a wsdl:port, SHOULD be attached to a
832 wsdl:binding, but MUST NOT be attached to a wsdl:portType; the latter is
833 prohibited because the assertion specifies a concrete behavior whereas the
834 wsdl:portType is an abstract construct.

835 *R2011: If a SERVICE supports attachments, the SERVICE MUST include the*
836 *wsdp:OptimizedMimeSerialization assertion in its policy.*

837 CORRECT:

```
838 <wsp:Policy
839     xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
840     xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" >
841   <wsdp:Profile />
842   <wsdp:OptimizedMimeSerialization wsp:Optional="true" />
843 </wsp:Policy>
```

845 To indicate how a SERVICE supports eventing, this profile defines the following WS-
846 Policy assertions:

```
847 <wsdp:PushDelivery wsp:Optional="true"? .../>
848 <wsdp:DurationExpiration wsp:Optional="true"? .../>
849 <wsdp:ActionFilter wsp:Optional="true"? .../>
```

850 The following describes additional, normative constraints on the outline above:

851 wsdp:PushDelivery
 852 A Subscribe SOAP ENVELOPE MUST use the
 853 "http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push"
 854 Delivery Mode [[WS-Eventing](#)].
 855 wsdp:DurationExpiration
 856 If a Subscribe or Renew SOAP ENVELOPE includes an Expiration, it MUST be of
 857 type xs:duration.
 858 wsdp:ActionFilter
 859 If a Subscribe SOAP ENVELOPE includes a Filter, it MUST use the
 860 "http://schemas.xmlsoap.org/ws/2005/05/devprof/Action" Filter Dialect. (See
 861 Section [6.1 Subscription](#).)
 862 These assertions have Endpoint Policy Subject: a policy expression containing these
 863 assertion MAY be attached to a wsdl:port, SHOULD be attached to a wsdl:binding,
 864 but MUST NOT be attached to a wsdl:portType.

865 *R2032: If a SERVICE exposes Notifications, it MUST include the wsdp:PushDelivery*
 866 *assertion in its policy.*

867 *R2033: If a SERVICE exposes Notifications, it MUST include the*
 868 *wsdp:DurationExpiration assertion in its policy.*

869 *R2034: If a SERVICE exposes Notifications, it MUST include the wsdp:ActionFilter*
 870 *assertion in its policy.*

871 Including these assertions reflects requirements R3009, 3016, 3017, and R3008.
 872 (See Section [6. Eventing](#).)

873 CORRECT:

```
874 <wsp:Policy
875     xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
876     xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" >
877   <wsdp:Profile />
878   <wsdp:OptimizedMimeSerialization wsp:Optional="true" />
879   <wsdp:PushDelivery />
880   <wsdp:DurationExpiration />
881   <wsdp:ActionFilter />
882 </wsp:Policy>
883
```

884 *R2041: If a SERVICE uses wsp:PolicyReference/@URI to attach a policy identified by*
 885 *an absolute URI, the SERVICE MUST have a Metadata Section with Dialect*
 886 *equal to "http://schemas.xmlsoap.org/ws/2004/09/policy" and Identifier*
 887 *equal to that URI.*

888 *R2025: If a SERVICE uses wsp:PolicyReference/@URI to attach a policy identified by*
 889 *an absolute URI, then, if no Dialect is specified in a Get Metadata SOAP*
 890 *ENVELOPE, in the corresponding Get Metadata Response SOAP ENVELOPE,*
 891 *the SERVICE MUST include the Metadata Section with Dialect equal to*
 892 *"http://schemas.xmlsoap.org/ws/2004/09/policy" and Identifier equal to that*
 893 *URI.*

894 *R2035: If a SERVICE uses wsp:PolicyReference/@URI to attach a policy identified by*
 895 *a relative URI, the SERVICE MUST embed that policy as a child of*
 896 *wsdl:definitions, and the policy MUST have a @wsu:Id containing that URI.*

897 **R2036: A SERVICE MUST NOT use @wsp:PolicyURIs to attach policy.**

898 Because all components in WSDL are extensible via elements [\[BP 1.1, Section 4\]](#),
899 attachment using `wsp:PolicyReference/@URI` is sufficient.

900 Get Metadata [\[WS-MetadataExchange\]](#) is the interoperable means for a CLIENT to
901 retrieve attached policy.

902 CORRECT:

```
903 <soap:Envelope
904   xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
905   xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
906   xmlns:wSDP="http://schemas.xmlsoap.org/ws/2005/05/devprof"
907   xmlns:wSOAP="http://schemas.xmlsoap.org/wSDL/soap12/"
908   xmlns:wSP="http://schemas.xmlsoap.org/ws/2004/09/policy"
909   xmlns:wSU
910 = "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
911 utility-1.0.xsd"
912   xmlns:wSX="http://schemas.xmlsoap.org/ws/2004/09/mex"
913   xmlns:wSA="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
914 <soap:Header>
915   <wSA:Action>
916     http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata/Response
917   </wSA:Action>
918   <wSA:RelatesTo>
919     uuid:82204a83-52f6-475c-9708-174fa27659ec
920   </wSA:RelatesTo>
921   <wSA:To>
922     http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
923   </wSA:To>
924 </soap:Header>
925 <soap:Body>
926   <wSX:Metadata>
927     <wSX:MetadataSection
928       Dialect="http://schemas.xmlsoap.org/wSDL/" >
929     <wSDL:definitions
930       targetNamespace="http://acme.example.com/colorbeam"
931       xmlns:image="http://printer.example.org/imaging" >
932     <wSP:Policy wsu:Id="Attachments" >
933       <wSDP:Profile />
934       <wSDP:OptimizedMimeSerialization wsp:Optional="true" />
935       <wSDP:PushDelivery />
936       <wSDP:DurationExpiration />
937       <wSDP:ActionFilter />
938     </wSP:Policy>
939
940     <!-- Other WSDL components omitted for brevity. -->
941
942     <wSDL:service name="MultiFunction" >
943       <wSDL:port name="Print" binding="image:PrintBinding" >
944         <wSOAP:address location="http://172.30.184.244/print" />
945         <wSP:PolicyReference URI="#Attachments"
946           wsdl:required="true" />
947       </wSDL:port>
948       <wSDL:port name="Scan" binding="image:ScanBinding" >
949         <wSOAP:address location="http://172.30.184.244/scan" />
950         <wSP:PolicyReference URI="#Attachments"
```

```

951         wsdl:required="true" />
952     </wsdl:port>
953 </wsdl:service>
954 </wsdl:definitions>
955 </wsx:MetadataSection>
956
957 <!-- Other Metadata Sections omitted for brevity. -->
958
959 </wsx:Metadata>
960 </soap:Body>
961 </soap:Envelope>
962

```

963 *R2022: If a DEVICE changes its Policy, it MUST increment the Metadata Version*
964 *exposed in Hello, Probe Match, and Resolve Match SOAP ENVELOPES as*
965 *wsd:MetadataVersion.*

966 Caching for DEVICE Policy is controlled by the wsd:MetadataVersion construct [[WS-](#)
967 [Discovery](#)]. Since a HOSTED SERVICE should not be a Target Service, any changes
968 to its Policy should not be controlled by the wsd:MetadataVersion construct.

969 6. Eventing

970 The scope of this section is the following set of Web services specifications. All of the
971 requirements in these specifications are included by reference except where
972 superseded by normative statements herein:

- 973 • [[WS-Eventing](#)]

974 6.1 Subscription

975 *R3009: A SERVICE MUST at least support Push Delivery Mode indicated by*
976 *"http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push".*

977 *R3010: A SERVICE MUST NOT generate a wse:DeliveryModeRequestedUnavailable*
978 *SOAP Fault in response to a Subscribe SOAP ENVELOPE with a Delivery Mode*
979 *of "http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push".*

980 The Push Delivery Mode [[WS-Eventing](#)] is the default Delivery Mode and indicates
981 the Event Source (SERVICE) will push Notifications to the Event Sink (CLIENT).

982 *R3017: If a SERVICE does not understand the [address] of the Notify To of a*
983 *Subscribe SOAP ENVELOPE, the SERVICE MUST generate a*
984 *wsa:DestinationUnreachable SOAP Fault.*

985 *R3018: If a SERVICE does not understand the [address] of the End To of a*
986 *Subscribe SOAP ENVELOPE, the SERVICE MUST generate a*
987 *wsa:DestinationUnreachable SOAP Fault.*

988 *R3019: If a SERVICE cannot deliver a Notification SOAP ENVELOPE to an Event Sink,*
989 *the SERVICE MAY terminate the corresponding Subscription and SHOULD*
990 *send a Subscription End SOAP ENVELOPE with a Status of*
991 *"http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryFailure".*

992 6.1.1 Filtering

993 To enable subscribing to one or more Notifications exposed by a SERVICE, this
994 profile defines a Filter Dialect designated
995 "http://schemas.xmlsoap.org/ws/2005/05/devprof/Action".

- 996 • A Filter in this Dialect contains a white space-delimited list of URIs that indicate
997 the **[action]** property of desired Notifications.
 - 998 • The content of a Filter in this Dialect is defined as
999 xs:list/@itemType="xs:anyURI" [[XML Schema Part 2](#)].
 - 1000 • A Filter in this Dialect evaluates to true for an Output Message of a Notification or
1001 Solicit-Response operation if and only if a URI in the Filter matches the **[action]**
1002 property of the Message using the
1003 "http://schemas.xmlsoap.org/ws/2005/04/discovery/rfc2396" matching rule
1004 [[WS-Discovery](#)].
- 1005 The Action Dialect uses the RFC 2396 prefix matching rule so CLIENTs can subscribe
1006 to a related set of Notifications by including the common prefix of the **[action]**
1007 property of those Notifications. Typically, the Notifications within a WSDL portType
1008 [[WSDL 1.1](#)] will share a common **[action]** property prefix, and specifying that prefix
1009 with the Action Dialect will be a convenient means to subscribe to all Notifications
1010 defined by a portType.

1011 *R3008: A SERVICE MUST at least support Filtering by the Dialect*
1012 *"http://schemas.xmlsoap.org/ws/2005/05/devprof/Action".*

1013 CORRECT:

```

1014 <soap:Envelope
1015   xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
1016   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1017   xmlns:wse="http://schemas.xmlsoap.org/ws/2004/08/eventing" >
1018   <soap:Header>
1019     <wsa:Action>
1020       http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe
1021     </wsa:Action>
1022     <wsa:MessageID>
1023       uuid:314bea3b-03af-47a1-8284-f495497f1e33
1024     </wsa:MessageID>
1025     <wsa:ReplyTo>
1026       <wsa:Address>
1027         http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
1028       </wsa:Address>
1029     </wsa:ReplyTo>
1030     <wsa:To>http://172.30.184.244/print</wsa:To>
1031   </soap:Header>
1032   <soap:Body>
1033     <wse:Subscribe>
1034       <wse:Delivery>
1035         <wse:NotifyTo>
1036           <wsa:Address>
1037             uuid:3726983d-02de-4d41-8207-d028ae92ce3d
1038           </wsa:Address>
1039         </wse:NotifyTo>
1040       </wse:Delivery>
1041       <wse:Expires>PT10M</wse:Expires>
1042       <wse:Filter
1043         Dialect="http://schemas.xmlsoap.org/ws/2005/05/devprof/Action"
1044       >
1045         http://printer.example.org/imaging/PrintBasicPortType/JobEndState
1046         http://printer.example.org/imaging/PrintBasicPortType/PrinterState
1047       </wse:Filter>

```



```

1048 </wse:Subscribe>
1049 </soap:Body>
1050 </soap:Envelope>
1051

```

1052 *R3011: A SERVICE MUST NOT generate a wse:FilteringNotSupported SOAP Fault in*
1053 *response to a Subscribe SOAP ENVELOPE.*

1054 A SERVICE must support filtering, at least by **[action]**, so the Filtering Not
1055 Supported SOAP Fault is not appropriate.

1056 *R3012: A SERVICE MUST NOT generate a wse:FilteringRequestedUnavailable SOAP*
1057 *Fault in response to a Subscribe SOAP ENVELOPE with a Filter Dialect of*
1058 *"http://schemas.xmlsoap.org/ws/2005/05/devprof/Action".*

1059 To indicate that a SERVICE does not expose any Notifications that would match the
1060 contents of a Filter with the Action Dialect, this profile defines the following SOAP
1061 Fault:

[action]	http://schemas.xmlsoap.org/ws/2005/05/devprof/Fault
[Code]	soap:Sender
[Subcode]	wsdp:FilterActionNotSupported
[Reason]	E.g., "no notifications match the supplied filter"
[Detail]	(None defined.)

1062 *R3020: If none of the Notifications exposed by a SERVICE match the [action] values*
1063 *in a Subscribe SOAP ENVELOPE Filter whose Dialect is*
1064 *"http://schemas.xmlsoap.org/ws/2005/05/devprof/Action", the SERVICE*
1065 *MUST generate a wsdp:FilterActionNotSupported SOAP Fault.*

1066 6.2 Subscription Duration and Renewal

1067 *R3005: If a Subscribe SOAP ENVELOPE contains a requested Expiration of type*
1068 *xs:dateTime, the SERVICE MAY include an Expiration of type xs:duration in*
1069 *the Subscribe Response SOAP ENVELOPE.*

1070 *R3006: If a Renew SOAP ENVELOPE contains a requested Expiration of type*
1071 *xs:dateTime, the SERVICE MAY include an Expiration of type xs:duration in*
1072 *the Renew Response SOAP ENVELOPE.*

1073 *R3016: A SERVICE MUST NOT generate a wse:UnsupportedExpirationType SOAP*
1074 *Fault in response to a Subscribe or Renew SOAP ENVELOPE with an Expiration*
1075 *type of xs:duration.*

1076 *R3013: A SERVICE MAY generate a wse:UnsupportedExpirationType SOAP Fault in*
1077 *response to a Subscribe or Renew SOAP ENVELOPE with an Expiration of type*
1078 *xs:dateTime.*

1079 Event Sources are required to have an internal clock, but there is no requirement
1080 that the clock be synchronized with other SERVICES. Therefore, Event Sources are
1081 required to express Subscription Expiration as a duration but are not required to
1082 express Subscription Expiration as an absolute time.

1083 *R3015: A SERVICE MAY generate a wsa:ActionNotSupported SOAP Fault in response*
1084 *to a Get Status SOAP ENVELOPE.*

1085 Event Sources are not required to support retrieving subscription status.

1086 7. Security

1087 This section defines a RECOMMENDED baseline for interoperable security between a
1088 DEVICE and a CLIENT. A DEVICE (or CLIENT) is free to support other security
1089 mechanisms in addition to, or in place of, this mechanism as specified by WSDL
1090 [[WSDL 1.1](#)], policies [[WS-Policy](#)], or other mechanisms. In the absence of an explicit
1091 indication stating that a different security mechanism is to be used, the default
1092 security mechanism defined here is assumed to apply.

1093 This section defines the protocols and message formats required to authenticate a
1094 DEVICE and securely communicate with a DEVICE. It references well-known
1095 algorithms and protocols for authentication, establishment of a session key, and
1096 encryption.

1097 This scope of this section is the following set of Web services specifications. All of the
1098 requirements in these specifications are included by reference except where
1099 superseded by normative statements herein:

- 1100 • [[AES/TLS](#)]
- 1101 • [[HTTP Authentication](#)]
- 1102 • [[SHA1](#)]
- 1103 • [[TLS](#)]
- 1104 • [[UUID](#)]
- 1105 • [[X.509.v3](#)]

1106 7.1 Secure Communication

1107 7.1.1 Integrity

1108 Integrity is the process that protects MESSAGES against tampering while in transit.
1109 Integrity is an optional component of DEVICE security. However, if provided,
1110 integrity MUST adhere to the following requirements:

1111 *R4000: A SERVICE MUST not send a SOAP ENVELOPE without protecting the*
1112 *integrity of any Message Information Header blocks matching the following*
1113 *XPath expressions: (a) /soap:Envelope/soap:Header/wsa:Action, (b)*
1114 */soap:Envelope/soap:Header/wsa:MessageID, (c)*
1115 */soap:Envelope/soap:Header/wsa:To, (d)*
1116 */soap:Envelope/soap:Header/wsa:ReplyTo, (e)*
1117 */soap:Envelope/soap:Header/wsa:RelatesTo.*

1118 *R4063: A SERVICE MAY reject a SOAP ENVELOPE that has unprotected Message*
1119 *Information Header blocks.*

1120 *R4001: A SERVICE MUST not send a SOAP ENVELOPE without protecting the*
1121 *integrity of the SOAP ENVELOPE Body in conjunction with any Message*
1122 *Information Block(s) from R4000.*

1123 *R4064: A SERVICE MAY reject a SOAP ENVELOPE that does not protect the integrity*
1124 *of the SOAP ENVELOPE Body.*

1125 In this profile, the integrity of discovery SOAP ENVELOPES is protected using
1126 message-level signatures, while the integrity of other MESSAGES is protected using a

1127 Secure Channel. Other profiles may use alternate mechanisms to protect the
1128 integrity of MESSAGES.

1129 **7.1.2 Confidentiality**

1130 Confidentiality is the process by which sensitive information is protected against
1131 unauthorized disclosure. Confidentiality is an optional component of DEVICE security;
1132 however, if provided, confidentiality MUST adhere to the following requirements:

1133 *R4002: A SERVICE MUST NOT send a SOAP ENVELOPE without encrypting the SOAP*
1134 *ENVELOPE Body.*

1135 *R4067: A SERVICE MAY reject a SOAP ENVELOPE that does not encrypt the SOAP*
1136 *ENVELOPE Body.*

1137 *R4003: A SENDER MUST provide key transfer information to authorized RECEIVERS.*

1138 In this profile, discovery MESSAGES are not encrypted, while other MESSAGES are
1139 encrypted using a Secure Channel. Other profiles may use alternate mechanisms to
1140 encrypt MESSAGES.

1141 **7.1.3 Authentication**

1142 Authentication is the process by which the identity of the sender is determined by
1143 the recipient. Authentication is an optional component of DEVICE security; however,
1144 if provided, authentication MUST adhere to the following requirements:

1145 *R4004: A SENDER MUST authenticate itself to a RECEIVER using credentials*
1146 *acceptable to the RECEIVER.*

1147 In this profile, authentication is done using certificates, either through a shared trust
1148 root or through a PIN / Password exchanged out of band. Other profiles may use
1149 alternate authentication mechanisms.

1150 If multicast messages are secured, the following additional requirements apply:

1151 *R4005: On multicast MESSAGES, a CLIENT MUST use an authentication credential*
1152 *that is suitable for all DEVICES that could legitimately process the multicast*
1153 *MESSAGE.*

1154 **7.1.4 Trust**

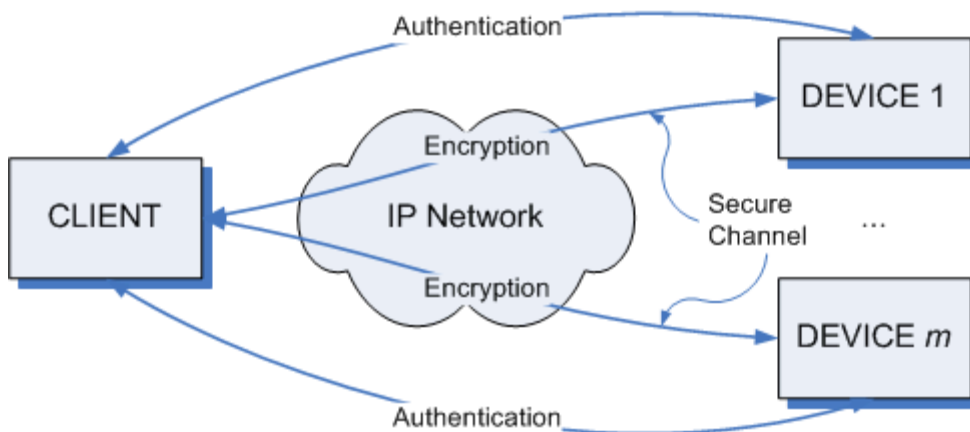
1155 There are different trust models associated with DEVICE security. The following
1156 requirements profile the kinds of trust that may be used with DEVICE security in this
1157 profile.

1158 *R4007: CLIENTs and DEVICES MUST have the necessary credentials to perform*
1159 *authentication.*

1160 The distribution of the credentials needed for establishing the trust relationship is out
1161 of the scope of this profile. The level of security as well as the supported protocols
1162 for a given CLIENT - DEVICE relationship are advertised in the policy assertions of
1163 the discovery MESSAGES defined herein.

1164 *R4008: A SERVICE MAY use additional mechanisms to verify the authenticity of the*
1165 *SENDER of any received MESSAGE by analyzing information provided by the*
1166 *lower networking layers.*

1167 **7.1.5 Network Model**



1168

1169 Following authentication, a DEVICE and a CLIENT communicate over a Secure (i.e.,
 1170 encrypted) Channel. The network is an IP-based network that can span one or more
 1171 administrative domains (such as a workgroup subnet), a domain comprised of
 1172 multiple subnets, or comprised of multiple administrative domains (such as the
 1173 global Internet). The level of security is determined by the security policies of the
 1174 administrative domain, which may vary between different environments.

1175 **R4009: Security MUST be applied for all MESSAGES received from, sent to, or**
 1176 **traversed through other administrative domains.**

1177 It is assumed that MESSAGES received from/via other administrative domains cannot
 1178 be trusted.

1179 **R4010: Except for MESSAGES exchanged during discovery, security SHALL be applied**
 1180 **at the Transport level. Discovery relies on MESSAGE security.**

1181 **7.1.6 Security Association**

1182 DEVICE association encompasses mutual authentication of DEVICE and CLIENT as
 1183 well as the establishment of a Secure Transport Channel over which the subsequent
 1184 communication between the CLIENT and the DEVICE takes place. The CLIENT
 1185 security requirements are advertised by the CLIENT during discovery as part of the
 1186 policy assertions carried in the respective Probe and Resolve SOAP ENVELOPES.
 1187 Security requirements can range from no security required to authentication and
 1188 communication over a Secure (i.e., encrypted) Channel.

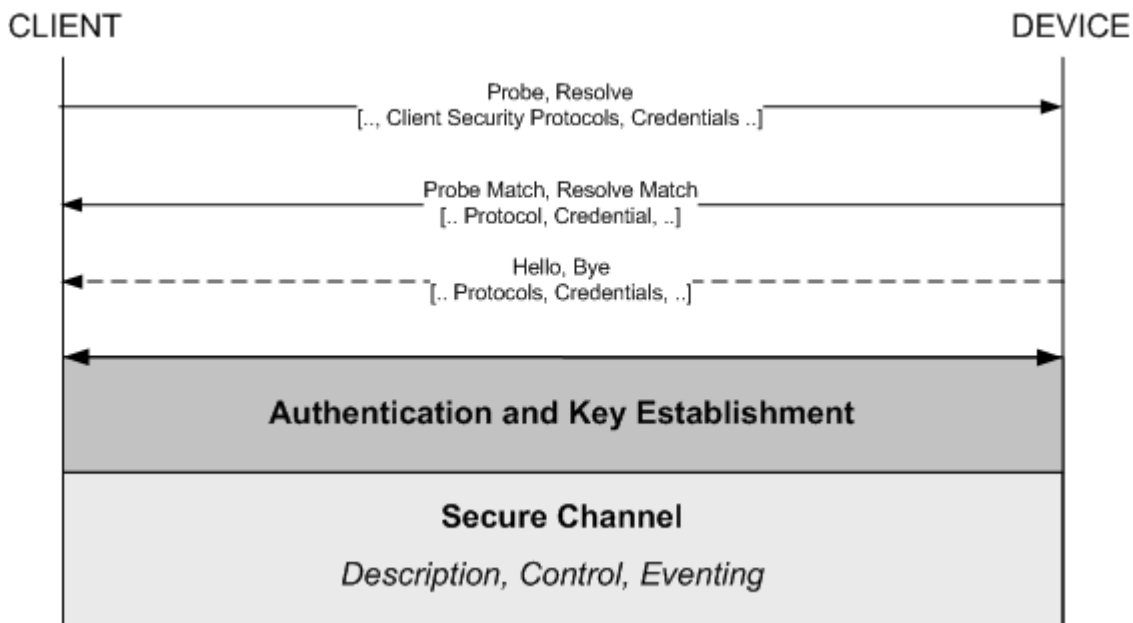
1189 The supported protocols for authentication and key establishment are advertised and
 1190 negotiated during discovery.

1191 **R4068: The CLIENT MAY include policy assertions in the Probe and Resolve SOAP**
 1192 **ENVELOPES containing the protocols it supports. If the CLIENT includes**
 1193 **multiple protocols, the protocols MUST be ordered with decreasing**
 1194 **preference, i.e., the first protocol listed is the preferred protocol the client**
 1195 **wishes to use.**

1196 **R4012: The DEVICE MUST select the protocol from the list of received protocols it**
 1197 **wishes to use for authentication and key establishment, and the DEVICE**
 1198 **MUST include the selected protocol in the policy assertion of the respective**
 1199 **Probe Match or Resolve Match SOAP ENVELOPE.**

1200 *R4013: Following discovery, the CLIENT MUST invoke the association process by*
 1201 *authenticating the DEVICE using a protocol for security and parameters*
 1202 *supported by both CLIENT and DEVICE as negotiated via Policy for the EPR.*

1203 The sequence for authentication and establishment of a Secure Channel is illustrated
 1204 below. It is assumed that credentials (certificates, shared secrets) are established by
 1205 an out-of-band mechanism prior or during the association phase. The out-of-band
 1206 mechanism is out of the scope of this profile. If the authentication is successful, a
 1207 Secure Channel is established. Subsequent operations like description, control, and
 1208 eventing use the Secure Channel.



1209
 1210 Once the DEVICE leaves the network, i.e., the DEVICE sends a Bye SOAP ENVELOPE,
 1211 the Secure Channel is removed, and the authentication information as well as
 1212 session keys become invalid.

1213 **7.1.7 DEVICE Behavior**

1214 *R4014: A DEVICE MAY require authentication of a CLIENT.*

1215 *R4015: To verify the authenticity of multicast messages sent by the DEVICE during*
 1216 *discovery, i.e., Hello and Bye SOAP ENVELOPES, multicast MESSAGES*
 1217 *SHOULD be signed.*

1218 *R4016: Unicast MESSAGES sent by a DEVICE in response to multicast MESSAGES,*
 1219 *i.e., Probe Match and Resolve Match SOAP ENVELOPES, SHOULD be signed.*

1220 *R4017: A CLIENT MAY ignore MESSAGES received during discovery that have no*
 1221 *signature or a nonverifiable signature.*

1222 *R4018: A DEVICE SHOULD cache authentication information for a CLIENT as valid as*
 1223 *long as the DEVICE is connected to the CLIENT.*

1224 **7.1.8 Security Protocols and Credentials**

1225 *R4025: A CLIENT MUST indicate the Security protocols and Credentials for*
 1226 *authentication and key establishment it supports in /soap:Envelope/*

1227 *soap:Header/ wsa:ReplyTo/ wsx:Metadata of a Probe and/or Resolve SOAP*
1228 *ENVELOPE.*

1229 *R4026: A DEVICE SHALL select from the list of Security Protocols and Credentials*
1230 *indicated by the CLIENT which Security Protocol the DEVICE wishes to use*
1231 *and return that selection in /soap:Envelope/ soap:Body/ */*
1232 *wsa:EndpointReference/ wsx:Metadata of the corresponding Probe Match (or*
1233 *Resolve Match) SOAP ENVELOPE.*

1234 Embedding a Metadata element [[WS-MetadataExchange](#)] within the extension point
1235 of an Endpoint Reference [[WS-Addressing](#)] is a means to provide metadata about the
1236 endpoint. This use of the Metadata element generalizes the existing **[policy]**
1237 property [[WS-Addressing](#)] and is the expected means to express WS-Policy in future
1238 versions of WS-Addressing.

1239 *R4027: A CLIENT MUST use the Security Protocol and Credential indicated by the*
1240 *DEVICE in the Probe Match (or Resolve Match) SOAP ENVELOPE for*
1241 *authentication and key establishment.*

1242 *R4028: CLIENTs and DEVICEs SHOULD support the following Security Protocols and*
1243 *Credentials for authentication and key establishment: TLS with client*
1244 *certificates and server certificates, respectively.*

1245 *R4069: CLIENTs and DEVICEs MUST support HTTP Basic Authentication.*

1246 **7.1.9 Security for Discovery**

1247 In the discovery phase, the client learns of the existence of the device on the
1248 network. Subsequently, the identity of the device is verified, and the device is
1249 connected to the client. The policy assertions carried in the messages exchanged
1250 during Discovery contain the CLIENT Security Requirements as well as the Security
1251 Protocols supported by CLIENT and DEVICE for authentication and establishment of a
1252 Secure Channel.

1253 *R4029: If a DEVICE cannot meet the CLIENT Security Requirements or if a CLIENT*
1254 *and a DEVICE do not support intersecting Security Protocols and Credentials,*
1255 *no association SHALL take place.*

1256 Probe

1257 A CLIENT initiates the discovery process by probing the network for a DEVICE it is
1258 interested in.

1259 *R4030: A Probe SOAP ENVELOPE SHOULD contain the Security Protocols and*
1260 *Credentials in /soap:Envelope/ soap:Header/ wsa:ReplyTo/ wsp:Policy.*

1261 *R4031: In the absence of any policy assertion for security, no security SHALL be*
1262 *required.*

1263 *R4032: A Device MUST NOT send a Probe Match SOAP ENVELOPE if any of the*
1264 *following are true: (a) the DEVICE is outside the local subnet of the CLIENT,*
1265 *and the Probe SOAP ENVELOPE was sent as multicast, or (b) the DEVICE does*
1266 *not support the indicated CLIENT Security Protocols and Credentials.*

1267 *R4065: A CLIENT MUST discard a Probe Match SOAP ENVELOPE if it is received*
1268 *MATCH_TIMEOUT seconds or more later than the last corresponding Probe*
1269 *SOAP ENVELOPE was sent.*

1270 Hello

1271 *R4034: A DEVICE SHOULD sign a Hello SOAP ENVELOPE.*

1272 One or more CLIENTs may respond to the Hello SOAP ENVELOPE and associate with
1273 the DEVICE.

1274 *R4035: If a DEVICE has multiple credentials, it SHOULD send separate Hello SOAP*
1275 *ENVELOPEs using different credentials to sign each.*

1276 Resolve

1277 *R4036: A Device MUST NOT send a Resolve Match SOAP ENVELOPE if any of the*
1278 *following are true: (a) the DEVICE is outside the local subnet of the CLIENT,*
1279 *and the Probe SOAP ENVELOPE was sent as multicast, or (b) the DEVICE does*
1280 *not support the indicated CLIENT Security Protocols and Credentials.*

1281 *R4066: A CLIENT MUST discard a Resolve Match SOAP ENVELOPE if it is received*
1282 *MATCH_TIMEOUT seconds or more later than the last corresponding Resolve*
1283 *SOAP ENVELOPE was sent.*

1284 Bye

1285 *R4037: A DEVICE SHOULD sign a Bye SOAP ENVELOPE.*

1286 *R4038: If a DEVICE has different credentials applicable to multiple CLIENTs, it*
1287 *SHOULD send separate Bye SOAP ENVELOPEs with the credentials for each of*
1288 *the previously associated CLIENTs.*

1289 **7.1.10 Authentication**

1290 The authentication step that follows discovery verifies the credentials of the DEVICE
1291 and CLIENT in a secure manner. In addition to verifying the credentials, a session
1292 key is established in the authentication handshake. Credentials may be cached on
1293 the DEVICE and/or CLIENT to simplify subsequent authentications. The CLIENT
1294 invokes the authentication process using the protocols and credentials indicated in
1295 the DEVICE policy assertions conveyed during the discovery phase.

1296 Transport Layer Security (TLS)

1297 TLS provides mutual authentication of CLIENT and DEVICE as well as the
1298 establishment of a Secure Channel over which MESSAGEs are exchanged in a secure
1299 manner.

1300 DEVICE Authentication with TLS

1301 *R4039: If TLS is negotiated as the Security Protocol, the CLIENT MUST initiate*
1302 *authentication with the DEVICE by setting up a TLS session.*

1303 *R4070: A DEVICE MUST indicate the use of TLS for a MESSAGE exchange using the*
1304 *"https" scheme URI contained in the DEVICE description and WSDL.*

1305 *R4042: Following the establishment of a Secure Channel using TLS, subsequent*
1306 *MESSAGE exchanges over HTTP SHOULD use an existing TLS session.*

1307 Certificates

1308 *R4043: Each DEVICE SHOULD have its own, unique Certificate.*

1309 The Certificate contains information pertinent to the specific device including its
1310 public key. Typically, certificates are issued by a trusted authority or a delegate (2nd
1311 tier) or a delegate of the delegate.

1312 *R4045: The format of the certificate MUST follow the common standard X.509v3.*

1313 An example of a self-signed X.509 certificate is shown below.

Type	Element	Usage	Example
Type	Element	Usage	Example
Basic Elements	Version	TLS	3
	Certificate Serial Number		1234567
	Signature Algorithm Identifier		RSA
	Issuer		a7731471-4b54-4a64-942c-7d481dcb9614
	Validity Period		11/09/2001 - 01/07/2015
	Subject	UUID	a7731471-4b54-4a64-942c-7d481dcb9614
	Subject Public Key Information		rsaEncryption 1024 10888232e76740bd873462ea2c64ca1d a6f9112656a34b949d32cede0e476547 84ba0f7e62e143429d3217ee45ce5304 308e65a6eee6474cb4d9a3c0295c8267 761661ccb7546a09d5f03a8ea3b1160 dac9fb6e6ba94e54b6c8ee892e492f4c e3a96bbd9d7b4c4bb98b7c052ff361ba cee01718122c4f0d826efc123bb1b03d
Extensions	Extended Key Usage	Server Authentication	1.3.6.1.5.5.7.3.1
		Client Authentication	1.3.6.1.5.5.7.3.2
Signature	Certification Authority's Digital Signature		5938f9908916cca32321916a184a6e75 2becb14fb99c4f33a03b03c3c752117c 91b8fb163d3541fca78bca235908ba69 1f7e36004a2d499a8e23951bd8af961d 36be05307ec34467a7c66fbb7fb5e49c 25e8dbdae4084ca9ba244b5bc1a377e5 262b9ef543ce47ad8a6b1d28c9138d0a dc8f5e3b469e42a5842221f9cf0a50d1

1314 The Subject field (listed above) contains the UUID in string representation format.

1315 Certificate management is out of the scope of this profile.

1316 TLS Authentication with Client Certificate

1317 *R4071: If the CLIENT and the DEVICE exchanged certificates during the TLS*
1318 *handshake, and the DEVICE as well as the CLIENT were able to verify the*
1319 *certificates, the CLIENT and DEVICE are mutually authenticated, and no*
1320 *further steps SHALL be required.*

1321 *R4046: A DEVICE MAY require an additional authentication step after the TLS*
1322 *handshake, if the DEVICE was not able to verify the certificate, or if the*
1323 *CLIENT did not provide a certificate during the TLS handshake.*

1324 *R4047: A DEVICE MAY require HTTP Authentication.*

1325 *R4048: If the HTTP authentication is successful, and the CLIENT presents a*
1326 *certificate to the DEVICE, the DEVICE SHOULD cache the certificate in its local*
1327 *certificate store of trusted certificates for future authentication of the CLIENT.*

1328 This avoids the need for HTTP authentication for subsequent associations.

1329 HTTP Authentication

1330 *R4049: The CLIENT MAY be required to authenticate itself to the DEVICE during the*
1331 *association phase.*

1332 HTTP authentication requires credentials in the form of username and password. It is
1333 assumed that how the CLIENT and DEVICE share knowledge of the username and
1334 password is out-of-band and beyond the scope of this profile.

1335 Because the authentication is performed over the Secure Channel established during
1336 TLS handshake, HTTP Basic authentication may be used safely.

1337 *R4050: If a DEVICE requires HTTP authentication, the DEVICE SHALL challenge the*
1338 *CLIENT using the HTTP 401 response code.*

1339 *R4051: A CLIENT MUST authenticate using one of the options listed in the HTTP-*
1340 *Authenticate header.*

1341 *R4052: HTTP Authentication MUST use the following parameters for username and*
1342 *password of the HTTP Request: UserName, PIN / Password.*

1343 The UserName is supplied to the DEVICE during HTTP authentication and MAY be
1344 used for establishing multiple access control classes, such as administrators, users,
1345 and guests. The naming and use of UserName is implementation-dependent and out
1346 of the scope of this profile.

1347 *R4053: If no UserName is provided, "admin" SHALL be used as the default*
1348 *UserName.*

1349 The purpose of the PIN / Password is to authenticate the CLIENT to the DEVICE
1350 during the HTTP authentication. In addition, the PIN / Password verifies the
1351 certificate that the DEVICE supplied during the TLS handshake.

1352 *R4054: The RECOMMENDED size of a PIN / Password is at least 8 characters using at*
1353 *least a 32 character alphabet.*

1354 *R4055: The PIN / Password that is unique to the DEVICE SHALL be conveyed to the*
1355 *CLIENT out-of-band. The methods of conveying the PIN out-of-band are out*
1356 *of the scope of this profile.*

1357 *R4056: To reduce the attack surface, the DEVICE and CLIENT MAY limit the number*
1358 *of failed authentication attempts as well as the time interval successive*
1359 *attempts are made for one TLS session.*

1360 Upon successful authentication, the DEVICE is associated with the CLIENT.

1361 **7.1.11 Secure Channel**

1362 Following Authentication, a Secure (i.e., encrypted) Channel at the transport level is
1363 established between CLIENT and DEVICE.

1364 *R4057: All secure communication for Description, Control, and Eventing between the*
1365 *CLIENT and DEVICE MUST use the Secure Channel. The protocols for*
1366 *encryption as well as the keys used for encryption are negotiated during the*
1367 *authentication phase.*

1368 *R4072: A DEVICE MUST support receiving and responding to a Probe SOAP*
1369 *ENVELOPE over HTTP using the Secure Channel.*

1370 *R4073: A DEVICE MAY ignore a Probe SOAP ENVELOPE sent over HTTP that does not*
1371 *use the Secure Channel.*

1372 As prescribed by R1015, a CLIENT may send a Probe over HTTP; this Probe (and
1373 Probe Match, if any) are sent using the Secure Channel.

1374 **7.1.12 TLS Ciphersuites**

1375 *R4059: It is the responsibility of the sender to convert the embedded URL to use*
1376 *HTTPS as different transport security mechanisms can be negotiated.*

1377 *R4060: A DEVICE MUST support the following TLS Ciphersuite:*
1378 *TLS_RSA_WITH_RC4_128_SHA.*

1379 *R4061: It is recommended that a DEVICE also support the following TLS Ciphersuite:*
1380 *TLS_RSA_WITH_AES_128_CBC_SHA.*

1381 *R4062: Additional Ciphersuites MAY be supported. They are negotiated during the*
1382 *TLS handshake.*

1383 **7.2 Security Policy Assertions**

1384 This profile defines the following assertions to indicate the Security Protocols.

```
1385 <wsdp:Tls wsp:Optional="true"? ... />  
1386 <wsdp:X509Cert wsp:Optional="true"? ... />
```

1387 The following describes additional, normative constraints on the outline above:

1388 wsdp:Tls

1389 This assertion indicates the SERVICE (or CLIENT) requires TLS.

1390 wsdp:X509Cert

1391 This assertion indicates the SERVICE (or CLIENT) requires X.509 certificates for
1392 authentication.

1393 These assertions have Endpoint Policy Subject [[WS-PolicyAttachment](#)]: a policy
1394 expression containing one of these assertions MAY be attached to a wsdl:port,
1395 SHOULD be attached to a wsdl:binding, but MUST NOT be attached to a
1396 wsdl:portType; the latter is prohibited because these assertions specify concrete
1397 behaviors whereas the wsdl:portType is an abstract construct.

1398 **8. Acknowledgements**

1399 This profile has been developed as a result of joint work with many individuals and
1400 teams, including: Don Box (Microsoft), Mike Fenelon (Microsoft), Bertus Greeff

1401 (Microsoft), Rob Hain (Microsoft), Rich Hasha (Microsoft), Gopal Kakivaya
1402 (Microsoft), Chris Kurt (Microsoft), David Lindsey (Lexmark), Jonathan Marsh
1403 (Microsoft), Sam Rhodus (Lexmark), Adam Sapek (Microsoft), Stacy Simpson
1404 (Lexmark), Lifan Tian (Ricoh), David Turner (Microsoft), Mike Vernal (Microsoft),
1405 Yaotian Wang (Ricoh), Kenny Wolf (Microsoft).

1406 **9. References**

1407 **[AES/TLS]**

1408 P. Chown, "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer
1409 Security (TLS)," June 2004. (See <http://www.ietf.org/rfc/rfc3268.txt>.)

1410 **[BP 1.1, Section 4]**

1411 K. Ballinger, et al, "Basic Profile Version 1.1, Section 4: Service Description,"
1412 August 2004. (See [http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-
1413 24.html#description](http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html#description).)

1414 **[HTTP/1.1]**

1415 R. Fielding, et al, "Hypertext Transfer Protocol -- HTTP/1.1," June 1999. (See
1416 <http://www.ietf.org/rfc/rfc2616.txt>.)

1417 **[HTTP Authentication]**

1418 J. Franks, et al, "HTTP Authentication: Basic and Digest Access Authentication,"
1419 June 1999. (See <http://www.ietf.org/rfc/rfc2617.txt>.)

1420 **[MIME]**

1421 N. Freed, et al, "Multipurpose Internet Mail Extensions (MIME) Part One: Format
1422 of Internet Message Bodies," November 1996. (See
1423 <http://www.ietf.org/rfc/rfc2045.txt>.)

1424 **[MTOM]**

1425 N. Mendelsohn, et al, "SOAP Message Transmission Optimization Mechanism,"
1426 January 2005. (See [http://www.w3.org/TR/2005/REC-soap12-mtom-
1427 20050125/](http://www.w3.org/TR/2005/REC-soap12-mtom-20050125/).)

1428 **[SHA1]**

1429 "Secure Hash Standard," April 1995. (See
1430 <http://www.itl.nist.gov/fipspubs/fip180-1.htm>.)

1431 **[SOAP 1.2, Part 1]**

1432 M. Gudgin, et al, "SOAP Version 1.2 Part 1: Messaging Framework," June 2003.
1433 (See <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>.)

1434 **[SOAP 1.2, Part 2, Section 7]**

1435 M. Gudgin, et al, " SOAP Version 1.2 Part 2: Adjuncts, Section 7: SOAP HTTP
1436 Binding," June 2003. (See [http://www.w3.org/TR/2003/REC-soap12-part2-
1437 20030624/#soapinhttp](http://www.w3.org/TR/2003/REC-soap12-part2-20030624/#soapinhttp).)

1438 **[SOAP-over-UDP]**

1439 H. Combs, et al, "SOAP-over-UDP," September 2004. (See
1440 <http://schemas.xmlsoap.org/ws/2004/09/soap-over-udp>.)

1441 **[TLS]**

1442 T. Dierks, et al, "The TLS Protocol, Version 1.0," January 1999. (See
1443 <http://www.ietf.org/rfc/rfc2246.txt>.)

1444 **[UUID]**

1445 P. Leach, et al, "A UUID URN Namespace," December 2004. (See
1446 <http://www.ietf.org/internet-drafts/draft-mealling-uuid-urn-05.txt>.)

- 1447 **[WS-Addressing]**
1448 D. Box, et al, "Web Services Addressing (WS-Addressing)," August 2004. (See
1449 <http://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810/>.)
- 1450 **[WS-Discovery]**
1451 J. Beatty, et al, "Web Services Dynamic Discovery (WS-Discovery)," April 2005.
1452 (See <http://schemas.xmlsoap.org/ws/2005/04/discovery/>.)
- 1453 **[WSDL 1.1]**
1454 E. Christensen, et al, "Web Services Description Language (WSDL) 1.1," March
1455 2001. (See <http://www.w3.org/TR/2001/NOTE-wsdl-20010315/>.)
- 1456 **[WSDL Binding for SOAP 1.2]**
1457 K. Ballinger, et al, "WSDL Binding for SOAP 1.2," April 2002. (See
1458 <http://groups.yahoo.com/group/soapbuilders/files/soap12WSDL.htm>.)
- 1459 **[WS-Eventing]**
1460 L. Cabrera, et al, "Web Services Eventing (WS-Eventing)," August 2004. (See
1461 <http://msdn.microsoft.com/ws/2004/08/ws-eventing/>.)
- 1462 **[WS-MetadataExchange]**
1463 K. Ballinger, et al, "Web Services Metadata Exchange (WSMetadataExchange),"
1464 September 2004. (See [http://msdn.microsoft.com/ws/2004/09/ws-
1465 metadataexchange/](http://msdn.microsoft.com/ws/2004/09/ws-metadataexchange/).)
- 1466 **[WS-Policy]**
1467 S. Bajaj, et al, "Web Services Policy Framework (WS-Policy)," September 2004.
1468 (See <http://schemas.xmlsoap.org/ws/2004/09/policy/>.)
- 1469 **[WS-PolicyAttachment]**
1470 S. Bajaj, et al, "Web Services Policy Attachment (WS-PolicyAttachment),"
1471 September 2004. (See <http://schemas.xmlsoap.org/ws/2004/09/policy/>.)
- 1472 **[WS-Security 2004]**
1473 A. Nadalin, et al, "Web Services Security: SOAP Message Security 1.0 (WS-
1474 Security 2004)," March 2004. (See [http://docs.oasis-
1475 open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf).)
- 1476 **[X.509.v3]**
1477 "ITU-T X.509.v3 Information technology - Open Systems Interconnection - The
1478 Directory: Public-key and attribute certificate frameworks (ISO/IEC/ITU 9594-
1479 8)."
- 1480 **[XML Schema, Part 1]**
1481 H. Thompson, et al, "XML Schema Part 1: Structures," May 2001. (See
1482 <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>.)
- 1483 **[XML Schema, Part 2]**
1484 P. Biron, et al, "XML Schema Part 2: Datatypes," May 2001. (See
1485 <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>.)

1486 **10. Informative References**

1487 The following documents are referenced for informational purposes only. They are
1488 not part of the scope of the profile:

- 1489 **[IPv6 Autoconfig]**
1490 S. Thomson, et al, "IPv6 Stateless Address Autoconfiguration," December 1998.
1491 (See <http://www.ietf.org/rfc/rfc2462.txt>.)

- 1492 **[DHCP]**
 1493 R. Droms, "Dynamic Host Configuration Protocol," March 1997. (See
 1494 <http://www.ietf.org/rfc/rfc2131.txt>.)
- 1495 **[RFC 2119]**
 1496 S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," RFC
 1497 2119, March 1997. (See <http://www.ietf.org/rfc/rfc2119.txt>.)
- 1498 **[XML Infoset]**
 1499 J. Cowan, et al, "XML Information Set (Second Edition)," February 2004. (See
 1500 <http://www.w3.org/TR/2004/REC-xml-infoset-20040204/>.)

1501 **Appendix I – Constants**

1502 The following constants are used throughout this profile. The values listed below
 1503 supersede other values defined in other specifications listed below.

Constant	Value	Specification
APP_MAX_DELAY	5,000 milliseconds	[WS-Discovery]
DISCOVERY_PORT	3702	[WS-Discovery]
MATCH_TIMEOUT	10 seconds	[WS-Discovery]
MAX_ENVELOPE_SIZE	32,767 octets	This profile
MAX_FIELD_SIZE	256 Unicode characters	This profile
MAX_URI_SIZE	2,048 octets	This profile
MULTICAST_UDP_REPEAT	2	[SOAP-over-UDP]
UDP_MAX_DELAY	250 milliseconds	[SOAP-over-UDP]
UDP_MIN_DELAY	50 milliseconds	[SOAP-over-UDP]
UDP_UPPER_DELAY	450 milliseconds	[SOAP-over-UDP]
UNICAST_UDP_REPEAT	2	[SOAP-over-UDP]

1504 **Appendix II – XML Schema**

1505 A normative copy of the XML Schema [[XML Schema Part 1](#), [Part 2](#)] description for
 1506 this specification can be retrieved from the following address:

1507 <http://schemas.xmlsoap.org/ws/2005/05/devprof/devicesprofile.xsd>

1508 A non-normative copy of the XML Schema description is listed below for convenience.

```

1509 <xs:schema
1510     targetNamespace="http://schemas.xmlsoap.org/ws/2005/05/devprof"
1511     xmlns:tns="http://schemas.xmlsoap.org/ws/2005/05/devprof"
1512     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1513     xmlns:xs="http://www.w3.org/2001/XMLSchema"
1514     elementFormDefault="qualified"
1515     blockDefault="#all" >
1516
1517     <xs:import
1518         namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1519         schemaLocation
1520         ="http://schemas.xmlsoap.org/ws/2004/08/addressing/addressing.xsd"
  
```

```

1521     />
1522
1523     <xs:element name="ThisModel" type="tns:ThisModelType" />
1524     <xs:complexType name="ThisModelType" >
1525         <xs:sequence>
1526             <xs:element name="Manufacturer" type="tns:LocalizedStringType"
1527                 maxOccurs="unbounded" />
1528             <xs:element name="ManufacturerUrl" type="xs:anyURI"
1529                 minOccurs="0" />
1530             <xs:element name="ModelName" type="tns:LocalizedStringType"
1531                 maxOccurs="unbounded" />
1532             <xs:element name="ModelNumber" type="xs:string" minOccurs="0" />
1533             <xs:element name="ModelUrl" type="xs:anyURI" minOccurs="0" />
1534             <xs:element name="PresentationUrl" type="xs:anyURI"
1535                 minOccurs="0" />
1536             <xs:any namespace="##other" processContents="lax"
1537                 minOccurs="0" maxOccurs="unbounded" />
1538         </xs:sequence>
1539         <xs:anyAttribute namespace="##other" processContents="lax" />
1540     </xs:complexType>
1541
1542     <xs:element name="ThisDevice" type="tns:ThisDeviceType" />
1543     <xs:complexType name="ThisDeviceType" >
1544         <xs:sequence>
1545             <xs:element name="FriendlyName" type="tns:LocalizedStringType"
1546                 maxOccurs="unbounded" />
1547             <xs:element name="FirmwareVersion" type="xs:string"
1548                 minOccurs="0" />
1549             <xs:element name="SerialNumber" type="xs:string" minOccurs="0" />
1550             <xs:any namespace="##other" processContents="lax"
1551                 minOccurs="0" maxOccurs="unbounded" />
1552         </xs:sequence>
1553         <xs:anyAttribute namespace="##other" processContents="lax" />
1554     </xs:complexType>
1555
1556     <xs:complexType name="LocalizedStringType" >
1557         <xs:simpleContent>
1558             <xs:extension base="xs:string" >
1559                 <xs:anyAttribute namespace="##other" processContents="lax" />
1560             </xs:extension>
1561         </xs:simpleContent>
1562     </xs:complexType>
1563
1564     <xs:element name="Relationship" >
1565         <xs:complexType>
1566             <xs:sequence>
1567                 <xs:any namespace="##any" processContents="lax"
1568                     minOccurs="0" maxOccurs="unbounded" />
1569             </xs:sequence>
1570             <xs:attribute name="Type" type="xs:anyURI" use="required" />
1571             <xs:anyAttribute namespace="##other" processContents="lax" />
1572         </xs:complexType>
1573     </xs:element>
1574
1575     <xs:element name="Host" type="tns:HostServiceType" />
1576     <xs:element name="Hosted" type="tns:HostServiceType" />
1577     <xs:complexType name="HostServiceType" >

```

```
1578     <xs:sequence>
1579         <xs:element ref="wsa:EndpointReference" />
1580         <xs:element ref="tns:Types" minOccurs="0" />
1581         <xs:element ref="tns:ServiceId" minOccurs="0" />
1582         <xs:any namespace="##other" processContents="lax"
1583             minOccurs="0" maxOccurs="unbounded" />
1584     </xs:sequence>
1585     <xs:anyAttribute namespace="##other" processContents="lax" />
1586 </xs:complexType>
1587
1588 <xs:element name="ServiceId" type="xs:anyURI" />
1589 <xs:element name="Types" type="tns:QNameListType" />
1590 <xs:simpleType name="QNameListType" >
1591     <xs:list itemType="xs:QName" />
1592 </xs:simpleType>
1593
1594 <xs:element name="Profile" type="tns:AssertionType" />
1595 <xs:element name="OptimizedMimeSerialization"
1596     type="tns:AssertionType" />
1597 <xs:element name="PushDelivery" type="tns:AssertionType" />
1598 <xs:element name="DurationExpiration" type="tns:AssertionType" />
1599 <xs:element name="ActionFilter" type="tns:AssertionType" />
1600 <xs:element name="Tls" type="tns:AssertionType" />
1601 <xs:element name="X509Cert" type="tns:AssertionType" />
1602
1603 <xs:complexType name="AssertionType" >
1604     <xs:complexContent>
1605         <xs:restriction base="xs:anyType">
1606             <xs:anyAttribute namespace="##other" processContents="lax" />
1607         </xs:restriction>
1608     </xs:complexContent>
1609 </xs:complexType>
1610
1611 </xs:schema>
1612
1613
1614
1615
```