



Web Services Dynamic Discovery (WS-Discovery) Version 1.1

Committee Specification 01

14 May 2009

Specification URIs:

This Version:

<http://docs.oasis-open.org/ws-dd/discovery/1.1/cs-01/wsdd-discovery-1.1-spec-cs-01.html>
<http://docs.oasis-open.org/ws-dd/discovery/1.1/cs-01/wsdd-discovery-1.1-spec-cs-01.docx>
(Authoritative Format)
<http://docs.oasis-open.org/ws-dd/discovery/1.1/cs-01/wsdd-discovery-1.1-spec-cs-01.pdf>

Previous Version:

<http://docs.oasis-open.org/ws-dd/discovery/1.1/cd-04/wsdd-discovery-1.1-spec-cd-04.html>
<http://docs.oasis-open.org/ws-dd/discovery/1.1/cd-04/wsdd-discovery-1.1-spec-cd-04.docx>
<http://docs.oasis-open.org/ws-dd/discovery/1.1/cd-04/wsdd-discovery-1.1-spec-cd-04.pdf>

Latest Version:

<http://docs.oasis-open.org/ws-dd/discovery/1.1/wsdd-discovery-1.1-spec.html>
<http://docs.oasis-open.org/ws-dd/discovery/1.1/wsdd-discovery-1.1-spec.docx>
<http://docs.oasis-open.org/ws-dd/discovery/1.1/wsdd-discovery-1.1-spec.pdf>

Technical Committee:

OASIS Web Services Discovery and Web Services Devices Profile (WS-DD) TC

Chair(s):

Toby Nixon, Microsoft Corporation
Alain Regnier, Ricoh Company Limited

Editor(s):

Vipul Modi, Microsoft Corporation
Devon Kemp, Canon Inc.

Declared XML Namespace(s):

<http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01>

Abstract:

This specification defines a discovery protocol to locate services. In an ad hoc mode of operation, probes are sent to a multicast group, and target services that match return a response directly to the requester. To scale to a large number of endpoints and to extend the reach of the protocol, this protocol defines a managed mode of operation and a multicast suppression behavior if a discovery proxy is available on the network. To minimize the need for polling, target services that wish to be discovered send an announcement when they join and leave the network.

Status:

This document was last revised or approved by the WS-DD TC on the above date. The level of approval is also listed above. Check the "Latest Version" or "Latest Approved Version" location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the

“Send A Comment” button on the Technical Committee’s web page at <http://www.oasis-open.org/committees/ws-dd/>.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (<http://www.oasis-open.org/committees/ws-dd/ipr.php>).

The non-normative errata page for this specification is located at <http://www.oasis-open.org/committees/ws-dd/>.

Notices

Copyright © OASIS® 2009. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is trademarks of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <http://www.oasis-open.org/who/trademark.php> for above guidance.

Table of Contents

1	Introduction	6
1.1	Composable Architecture.....	6
1.2	Requirements.....	6
1.3	Non Requirements.....	7
1.4	Terminology	7
1.4.1	Notational Conventions.....	7
1.4.2	Terms and Definitions.....	7
1.5	XML Namespaces	8
1.6	XSD and WSDL Files	9
1.7	Example	9
1.8	Normative References	11
1.9	Non-Normative References.....	12
2	Model.....	13
2.1	Endpoint References	13
2.2	Operational Modes	13
2.2.1	Ad hoc Mode	13
2.2.2	Managed Mode	15
2.2.3	Dynamic Mode Switching.....	16
2.3	Conceptual Message Content.....	18
3	Protocol Assignments.....	20
3.1.1	Ad hoc mode over IP multicast	20
3.1.2	Managed mode over HTTP.....	20
3.1.3	Application Level Transmission Delay.....	20
4	Hello and Bye.....	21
4.1	Hello.....	21
4.1.1	Target Service	22
4.1.2	Client.....	24
4.1.3	Discovery Proxy	24
4.2	Bye.....	25
4.2.1	Target Service	25
4.2.2	Client.....	27
4.2.3	Discovery Proxy	27
5	Probe and Probe Match.....	28
5.1	Matching Types and Scopes.....	28
5.2	Probe	29
5.2.1	Client.....	30
5.2.2	Target Service	31
5.2.3	Discovery Proxy	31
5.3	Probe Match.....	31
5.3.1	Target Service	33
5.3.2	Discovery Proxy	33
6	Resolve and Resolve Match.....	35
6.1	Matching Endpoint Reference.....	35

6.2	Resolve	35
6.2.1	Client.....	35
6.2.2	Target Service	36
6.2.3	Discovery Proxy	36
6.3	Resolve Match.....	36
6.3.1	Target Service	37
6.3.2	Discovery Proxy	37
7	Application Sequencing.....	38
8	Security.....	39
8.1	Security Model.....	39
8.2	Compact Signature Format.....	39
8.3	Security Considerations	42
9	Conformance	44
A.	Acknowledgements	45
B.	Revision History	47

1 Introduction

This specification defines a discovery protocol to locate services. The primary scenario for discovery is a client searching for one or more target services. The protocol defines two modes of operation, an ad hoc mode and a managed mode. In an ad hoc mode, to find a target service by the type of the target service, a scope in which the target service resides, or both, a client sends a probe message to a multicast group; target services that match the probe send a response directly to the client. To locate a target service by name, a client sends a resolution request message to the same multicast group, and again, the target service that matches sends a response directly to the client.

To minimize the need for polling in an ad hoc network, when a target service joins the network, it sends an announcement message to the same multicast group. By listening to this multicast group, clients can detect newly available target services without repeated probing.

To scale to a large number of endpoints and to extend the reach of the protocol beyond the range of an ad hoc network, this specification defines a managed mode of operation and a multicast suppression behavior if a discovery proxy is available on the network. In managed mode, target services send unicast announcement messages to a discovery proxy and clients send unicast probe and resolve messages to a discovery proxy. To reduce multicast traffic, when a discovery proxy detects a probe or resolution request sent multicast on an ad hoc network, it sends an announcement for itself. By listening for these announcements, clients detect discovery proxies and switch to a managed mode of operation and send unicast probe and resolve messages directly to a discovery proxy. However, if a discovery proxy is unresponsive, clients revert to an ad hoc mode of operation.

To support networks with explicit network management services like DHCP, DNS, domain controllers, directories, etc., this specification acknowledges that clients and/or target services can be configured to behave differently than defined herein. For example, another specification may define a well-known DHCP record containing the address of a discovery proxy, and compliance with that specification may require client and target services to operate in a managed mode and send messages to this discovery proxy rather than to a multicast group. While the specific means of such configuration is beyond the scope of this specification, it is expected that any such configuration would allow clients and/or target services to migrate smoothly between carefully-managed and ad hoc networks.

1.1 Composable Architecture

The Web service specifications (WS-*) are designed to be composed with each other to provide a rich set of tools to provide security in the Web services environment. This specification specifically relies on other Web service specifications to provide secure, reliable, and/or transacted message delivery and to express Web service and client policy.

1.2 Requirements

This specification intends to meet the following requirements:

- Allow discovery of services in ad hoc networks with a minimum of networking services (e.g., no DNS or directory services).
- Leverage network services to reduce network traffic and allow discovery of services in managed networks where such network services exist.
- Enable smooth transitions between ad hoc and managed networks.
- Enable discovery of resource-limited service implementations.
- Support bootstrapping to other Web service protocols as well as other transports.
- Enable discovery of services by type and within scope.
- Leverage other Web service specifications for secure, reliable, transacted message delivery.
- Provide extensibility for more sophisticated and/or currently unanticipated scenarios.

46 1.3 Non Requirements

47 This specification does not intend to meet the following requirements:

- 48 • Provide liveness information on services.
- 49 • Define a data model for service description or define rich queries over that description.
- 50 • Support Internet-scale discovery.

51 1.4 Terminology

52 The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD
53 NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described
54 in [\[RFC 2119\]](#).

55 1.4.1 Notational Conventions

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

- 57 • The syntax appears as an XML instance, but values in italics indicate data types instead of literal
58 values.
- 59 • Characters are appended to elements and attributes to indicate cardinality:
 - 60 • "?" (0 or 1)
 - 61 • "*" (0 or more)
 - 62 • "+" (1 or more)
- 63 • The character "|" is used to indicate a choice between alternatives.
- 64 • The characters "[" and "]" are used to indicate that contained items are to be treated as a group with
65 respect to cardinality or choice.
- 66 • Ellipses (i.e., "...") indicate points of extensibility. Additional children and/or attributes MAY be added
67 at the indicated extension points but MUST NOT contradict the semantics of the parent and/or owner,
68 respectively. If a receiver does not recognize an extension, the receiver SHOULD ignore the
69 extension.
- 70 • XML namespace prefixes (see Table 1) are used to indicate the namespace of the element being
71 defined.

72 Elsewhere in this specification, the characters "[" and "]" are used to call out references and property
73 names. This specification uses the **[action]** and Fault properties [\[WS-Addressing\]](#) to define faults.

74 1.4.2 Terms and Definitions

75 Defined below are the basic definitions for the terms used in this specification.

76 **Target Service**

77 An endpoint that makes itself available for discovery.

78 **Client**

79 An endpoint that searches for Target Service(s).

80 **Discovery Proxy**

81 An endpoint that facilitates discovery of Target Services by Clients.

82 **Hello**

83 A message sent by a Target Service when it joins a network; this message contains key
84 information for the Target Service. A Hello message is also sent by a Discovery Proxy to reduce
85 multicast traffic on an ad hoc network; this message contains key information about the Discovery
86 Proxy.

- 87 **Bye**
 88 A best-effort message sent by a Target Service when it leaves a network.
- 89 **Probe**
 90 A message sent by a Client searching for a Target Service by Type and/or Scope.
- 91 **Resolve**
 92 A message sent by a Client searching for a Target Service by name.
- 93 **Type**
 94 An identifier for a set of messages an endpoint sends and/or receives (e.g., a WSDL 1.1
 95 portType, see [WSDL 1.1]).
- 96 **Scope**
 97 An extensibility point that allows Target Services to be organized into logical groups.
- 98 **Metadata**
 99 Information about the Target Service; includes, but is not limited to, transports and protocols a
 100 Target Service understands, Types it implements, and Scopes it is in.
- 101 **Ad hoc Mode**
 102 An operational mode of discovery in which the Hello, Bye, Probe and Resolve messages are sent
 103 multicast.
- 104 **Managed Mode**
 105 An operational mode of discovery in which the Hello, Bye, Probe and Resolve messages are sent
 106 unicast to a Discovery Proxy.
- 107 **Ad hoc Network**
 108 A network in which discovery is performed in an ad hoc mode.
- 109 **Managed Network**
 110 A network in which discovery is performed in a managed mode.

111 1.5 XML Namespaces

112 The XML Namespace URI that MUST be used by implementations of this specification is:

113

114 `http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01`

115

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

118 **Table 1: Prefix and XML Namespaces used in this specification.**

Prefix	XML Namespace	Specification(s)
s	(Either SOAP 1.1 or 1.2)	(Either SOAP 1.1 or 1.2)
s11	http://schemas.xmlsoap.org/soap/envelope/	[SOAP 1.1]
s12	http://www.w3.org/2003/05/soap-envelope	[SOAP 1.2]
a	http://www.w3.org/2005/08/addressing	[WS-Addressing]
d	http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01	This specification

ds	http://www.w3.org/2000/09/xmldsig#	[XML Sig]
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd	[WS-Security]
xs	http://www.w3.org/2001/XMLSchema	[XML Schema Part 1, 2]
ec	http://www.w3.org/2001/10/xml-exc-c14n#	[EXC-C14N]

119 1.6 XSD and WSDL Files

120 Dereferencing the XML namespace defined in Section 1.5 XML Namespaces will produce the Resource
 121 Directory Description Language (RDDL) [RDDL] document that describes this namespace, including the
 122 XML schema [XML Schema Part 1, 2] and WSDL [WSDL 1.1] declarations associated with this
 123 specification.

124 SOAP bindings for the WSDL [WSDL 1.1], referenced in the RDDL [RDDL] document, MUST use
 125 “document” for the *style* attribute.

126 1.7 Example

127 Table 2 lists an example Probe message sent multicast by a Client searching for a printer in an ad hoc
 128 mode.

129 **Table 2: Example Probe sent multicast in an ad hoc mode.**

```

130 (01) <s:Envelope
131 (02)   xmlns:a="http://www.w3.org/2005/08/addressing"
132 (03)   xmlns:d="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01"
133 (04)   xmlns:i="http://printer.example.org/2003/imaging"
134 (05)   xmlns:s="http://www.w3.org/2003/05/soap-envelope" >
135 (06)   <s:Header>
136 (07)     <a:Action>
137 (08)       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Probe
138 (09)     </a:Action>
139 (10)     <a:MessageID>
140 (11)       urn:uuid:0a6dc791-2be6-4991-9af1-454778a1917a
141 (12)     </a:MessageID>
142 (13)     <a:To>urn:docs-oasis-open-org:ws-dd:ns:discovery:2009:01</a:To>
143 (14)   </s:Header>
144 (15)   <s:Body>
145 (16)     <d:Probe>
146 (17)       <d:Types>i:PrintBasic</d:Types>
147 (18)       <d:Scopes>
148 (19)         MatchBy="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/ldap" >
149 (20)           ldap:///ou=engineering,o=examplecom,c=us
150 (21)         </d:Scopes>
151 (22)       </d:Probe>
152 (23)     </s:Body>
153 (24)   </s:Envelope>
154 (25)

```

155 Lines (07-09) in Table 2 indicate the message is a Probe, and Line (13) indicates it is being sent to a well-
 156 known address [RFC 2141].

157 Because there is no explicit ReplyTo SOAP header block [WS-Addressing], any response to this Probe
 158 message will be sent as a UDP packet to the source IP address and port of the Probe transport header
 159 [SOAP/UDP].

160 Lines (17-21) specify two constraints on the Probe: Line (17) constrains responses to Target Services
 161 that implement a basic print Type; Lines (18-21) constrain responses to Target Services in the Scope for
 162 an engineering department. Only Target Services that satisfy both of these constraints will respond.
 163 Though both constraints are included in this example of a Probe, they are OPTIONAL.

164 Table 3 lists an example Probe Match message sent in response to the Probe in Table 2.

165 **Table 3: Example ProbeMatch sent in response to the ad hoc Probe in Table 2.**

```
166 (01) <s:Envelope
167 (02)   xmlns:a="http://www.w3.org/2005/08/addressing"
168 (03)   xmlns:d="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01"
169 (04)   xmlns:i="http://printer.example.org/2003/imaging"
170 (05)   xmlns:s="http://www.w3.org/2003/05/soap-envelope" >
171 (06)   <s:Header>
172 (07)     <a:Action>
173 (08)       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/ProbeMatches
174 (09)     </a:Action>
175 (10)     <a:MessageID>
176 (11)       urn:uuid:e32e6863-ea5e-4ee4-997e-69539d1ff2cc
177 (12)     </a:MessageID>
178 (13)     <a:RelatesTo>
179 (14)       urn:uuid:0a6dc791-2be6-4991-9af1-454778a1917a
180 (15)     </a:RelatesTo>
181 (16)     <a:To>
182 (17)       http://www.w3.org/2005/08/addressing/anonymous
183 (18)     </a:To>
184 (19)     <d:AppSequence InstanceId="1077004800" MessageNumber="2" />
185 (20)   </s:Header>
186 (21)   <s:Body>
187 (22)     <d:ProbeMatches>
188 (23)       <d:ProbeMatch>
189 (24)         <a:EndpointReference>
190 (25)           <a:Address>
191 (26)             urn:uuid:98190dc2-0890-4ef8-ac9a-5940995e6119
192 (27)           </a:Address>
193 (28)         </a:EndpointReference>
194 (29)         <d:Types>i:PrintBasic i:PrintAdvanced</d:Types>
195 (30)         <d:Scopes>
196 (31)           ldap:///ou=engineering,o=examplecom,c=us
197 (32)           ldap:///ou=floor1,ou=b42,ou=anytown,o=examplecom,c=us
198 (33)           http://itdept/imaging/deployment/2004-12-04
199 (34)         </d:Scopes>
200 (35)         <d:XAddr>http://prn-example/PRN42/b42-1668-a</d:XAddr>
201 (36)         <d:MetadataVersion>75965</d:MetadataVersion>
202 (37)       </d:ProbeMatch>
203 (38)     </d:ProbeMatches>
204 (39)   </s:Body>
205 (40) </s:Envelope>
206 (41)
```

207 Lines (07-09) in Table 3 indicate this message is a Probe Match, and Lines (13-15) indicate that it is a
208 response to the Probe in Table 2. Because the Probe did not have an explicit ReplyTo SOAP header
209 block, Lines (16-18) indicate that the response was sent to the source IP address and port of the
210 transport header of the Probe. Line (19) contains an instance identifier as well as a message number; this
211 information allows the receiver to reorder discovery messages received from a Target Service.

212 Lines (23-37) describe a single Target Service.

213 Lines (24-28) contain the stable, unique identifier for the Target Service that is constant across network
214 interfaces, transport addresses, and IPv4/v6. In this case, the value is a UUID based URN [RFC 4122]
215 scheme URI, but it can be a transport URI (like the one in Line 35) if it meets stability and uniqueness
216 requirements.

217 Line (29) lists the Types (see, e.g., [WSDL 1.1]) implemented by the Target Service, in this example, a
218 basic print type that matched the Probe as well as an advanced print type.

219 Lines (30-34) list three administrative Scopes, one that matched the Probe (Line 31), one that is specific
220 to a particular physical location (Line 32), and one that includes data useful when switching over to new
221 infrastructure (Line 33). As in this case, the Scopes can be a heterogeneous collection of deployment-
222 related information.

223 Line (35) indicates the transport addresses where the Target Service can be reached; in this case, a
224 single HTTP transport address.
225 Line (36) contains the version of the metadata for the Target Service; as explained below, this version is
226 incremented if there is a change in the metadata for the Target Service (including Lines 29-34).

227 1.8 Normative References

- 228 **[RFC 2119]** S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
229 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
230
- 231 **[RDDL]** Jonathan Borden, et al, *Resource Directory Description Language (RDDL) 2.0*,
232 <http://www.openhealth.org/RDDL/20040118/rddl-20040118.html>, 18 January
233 2004.
234
- 235 **[IANA]** *Port Numbers*, <http://www.iana.org/assignments/port-numbers>.
236
- 237 **[RFC 2141]** R. Moats, *URN Syntax*, <http://www.ietf.org/rfc/rfc2141.txt>, IETF RFC 2141, May
238 1997.
239
- 240 **[RFC 3986]** T. Berners-Lee, et al, *Uniform Resource Identifiers (URI): Generic Syntax*,
241 <http://www.ietf.org/rfc/rfc3986.txt>, IETF RFC 3986, January 2005.
242
- 243 **[RFC 3987]** M. Duerst, et al, *Internationalized Resource Identifiers (IRIs)*,
244 <http://www.ietf.org/rfc/rfc3987.txt>, IETF RFC 3987, January 2005.
245
- 246 **[RFC 4514]** Zeilenga, K., Ed., *Lightweight Directory Access Protocol (LDAP): String*
247 *Representation of Distinguished Names*, <http://www.ietf.org/rfc/rfc4514.txt>, IETF
248 RFC 4514, June 2006
249
- 250 **[RFC 4516]** Smith, M., Ed. and T. Howes, *Lightweight Directory Access Protocol (LDAP):*
251 *Uniform Resource Locator*, <http://www.ietf.org/rfc/rfc4516.txt>, IETF RFC 4516,
252 June 2006.
253
- 254 **[RFC 4122]** P. Leach, et al, *A Universally Unique Identifier (UUID) URN Namespace*,
255 <http://www.ietf.org/rfc/rfc4122.txt>, IETF RFC 4122, July 2005.
256
- 257 **[Namespaces in XML 1.1]** W3C Recommendation, *Namespaces in XML 1.1 (Second Edition)*,
258 <http://www.w3.org/TR/2006/REC-xml-names11-20060816/>, 16 August 2006.
259
- 260 **[XML Schema, Part 1]** W3C Recommendation, *XML Schema Part 1: Structures Second Edition*,
261 <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/>, 28 October 2004.
262
- 263 **[XML Schema, Part 2]** W3C Recommendation, *XML Schema Part 2: Datatypes Second Edition*,
264 <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>, 28 October 2004.
265
- 266 **[XML Sig]** W3C Recommendation, *XML-Signature Syntax and Processing (Second*
267 *Edition)*, <http://www.w3.org/TR/2008/REC-xmlsig-core-20080610/>, 10 June
268 2008.
269
- 270 **[EXC-C14N]** W3C Recommendation, *Exclusive XML Canonicalization*,
271 <http://www.w3.org/TR/2002/REC-xml-exc-c14n-20020718/>, 18 July 2002.
272

- 273 [SOAP 1.1] W3C Note, *Simple Object Access Protocol (SOAP) 1.1*,
274 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>, 08 May 2000.
275
- 276 [SOAP 1.2] W3C Recommendation, *SOAP Version 1.2 Part 1: Messaging Framework*
277 *(Second Edition)*, <http://www.w3.org/TR/2007/REC-soap12-part1-20070427/>, 27
278 April 2007.
279
- 280 [SOAP 1.2 Part 2] W3C Recommendation, *SOAP Version 1.2 Part 2: Adjuncts (Second Edition)*,
281 <http://www.w3.org/TR/2007/REC-soap12-part2-20070427/>, 27 April 2007.
282
- 283 [SOAP/UDP] OASIS Committee Specification 01, *SOAP-over-UDP Version 1.1*,
284 [http://docs.oasis-open.org/ws-dd/soapoverudp/1.1/cs-01/wsdd-soapoverudp-1.1-](http://docs.oasis-open.org/ws-dd/soapoverudp/1.1/cs-01/wsdd-soapoverudp-1.1-spec-cs-01.docx)
285 [spec-cs-01.docx](http://docs.oasis-open.org/ws-dd/soapoverudp/1.1/cs-01/wsdd-soapoverudp-1.1-spec-cs-01.docx), May 2009.
286
- 287 [WSDL 1.1] W3C Note, *Web Services Description Language (WSDL) 1.1*,
288 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315/>, 15 March 2001.
289
- 290 [WS-Addressing] W3C Recommendation, *Web Services Addressing 1.0 - Core*,
291 <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>, 9 May 2006.
292
- 293 [WS-Security] OASIS Standard, *Web Services Security: SOAP Message Security 1.1 (WS-*
294 *Security 2004)*, [http://www.oasis-open.org/committees/download.php/16790/wss-](http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf)
295 [v1.1-spec-os-SOAPMessageSecurity.pdf](http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf), February 2006.

296 1.9 Non-Normative References

- 297 [WS-Trust] OASIS Standard, *WS-Trust 1.4*, [http://docs.oasis-open.org/ws-sx/ws-](http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/os/ws-trust-1.4-spec-os.doc)
298 [trust/v1.4/os/ws-trust-1.4-spec-os.doc](http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/os/ws-trust-1.4-spec-os.doc), February 2009.
299
- 300 [WS-SecureConversation] OASIS Standard, *WS-SecureConversation 1.4*, [http://docs.oasis-](http://docs.oasis-open.org/ws-sx/ws-secureconversation/v1.4/os/ws-secureconversation-1.4-spec-os.doc)
301 [open.org/ws-sx/ws-secureconversation/v1.4/os/ws-secureconversation-1.4-spec-](http://docs.oasis-open.org/ws-sx/ws-secureconversation/v1.4/os/ws-secureconversation-1.4-spec-os.doc)
302 [os.doc](http://docs.oasis-open.org/ws-sx/ws-secureconversation/v1.4/os/ws-secureconversation-1.4-spec-os.doc), February 2009.

303 2 Model

304 2.1 Endpoint References

305 As part of the discovery process, Target Services present to the network (a) a stable identifier and (b) one
306 or more transport addresses at which network messages can be directed.

307 The stable identifier is contained in an `a:EndpointReference` element [WS-Addressing]. Nearly all of
308 the SOAP messages defined herein contain the `a:EndpointReference` element, a facsimile is
309 reproduced here for convenience:

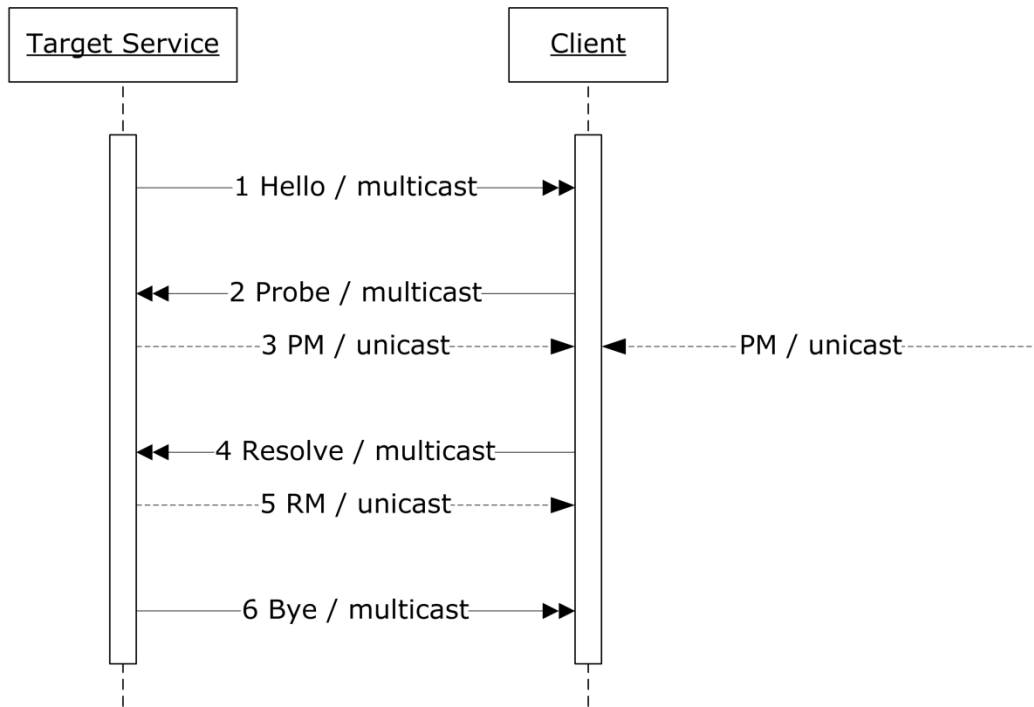
```
310 <a:EndpointReference>  
311   <a:Address>xs:anyURI</a:Address>  
312   <a:ReferenceParameters>xs:any*</a:ReferenceParameters>?  
313   <a:Metadata>xs:any*</a:Metadata>?  
314   ...  
315 </a:EndpointReference>
```

316 The `a:Address` element [WS-Addressing] is an absolute IRI [RFC 3987] that need not be a network-
317 resolvable transport address. By convention, it is RECOMMENDED that the value of this element be a
318 stable globally-unique identifier (GUID) based URN [RFC 4122] scheme URI that remains constant
319 across all network interfaces and throughout the lifetime of the Target Service. If the value of this element
320 is not a network-resolvable transport address, such transport address(es) are conveyed in a separate
321 `d:XAddrs` element defined herein (see below).

322 2.2 Operational Modes

323 2.2.1 Ad hoc Mode

324 In an ad hoc mode discovery messages are sent multicast and response messages are sent unicast.
325 Figure 1 depicts the message exchanges between a Target Service and a Client operating in an ad hoc
326 mode.

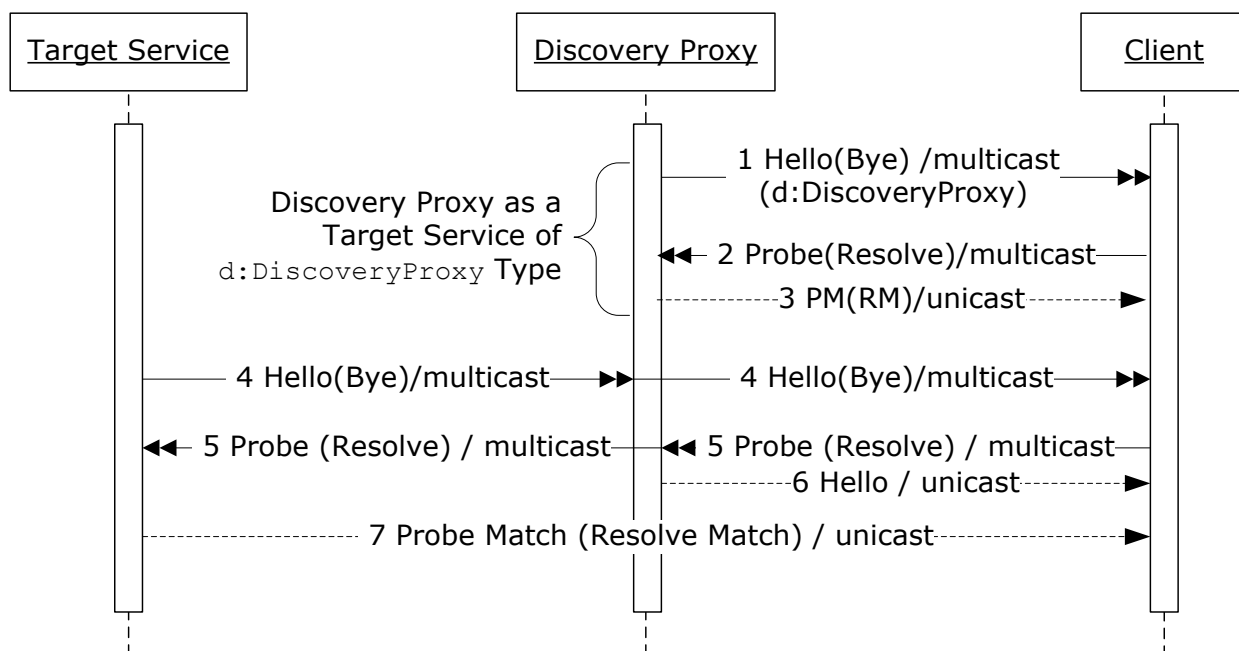


327

328 **Figure 1 : Message Exchanges in an ad hoc mode.**

329 A Target Service sends a multicast Hello message (1) when it joins a network (see Section 4.1.1 Target
 330 Service). A Client listens for multicast Hello messages (see Section 4.1.2 Client). A Client sends a
 331 multicast Probe message (2) to locate Target Services (see Section 5.2.1 Client). If a Target Service
 332 matches the Probe it responds with a unicast Probe Match (PM) message (3) (see Section 5.3.1 Target
 333 Service). Other matching Target Services MAY also send unicast Probe Match. A Target Service MAY
 334 also accept and respond to unicast Probe messages sent to its transport address(es) (see Section 5.2.2
 335 Target Service). A Client sends a multicast Resolve message (4) to locate a particular Target Service
 336 (see Section 6.2.1 Client). If a Target Service matches the Resolve it responds with a unicast Resolve
 337 Match (RM) message (5) (see Section 6.3.1 Target Service). A Target Service makes an effort to send a
 338 multicast Bye message (6) when it leaves a network (see Section 4.2.1 Target Service). A Client listens
 339 for multicast Bye messages (see 4.2.2 Client).

340 Figure 2 depicts the message exchanges in an ad hoc mode when a Discovery Proxy is present on the
 341 network.



342

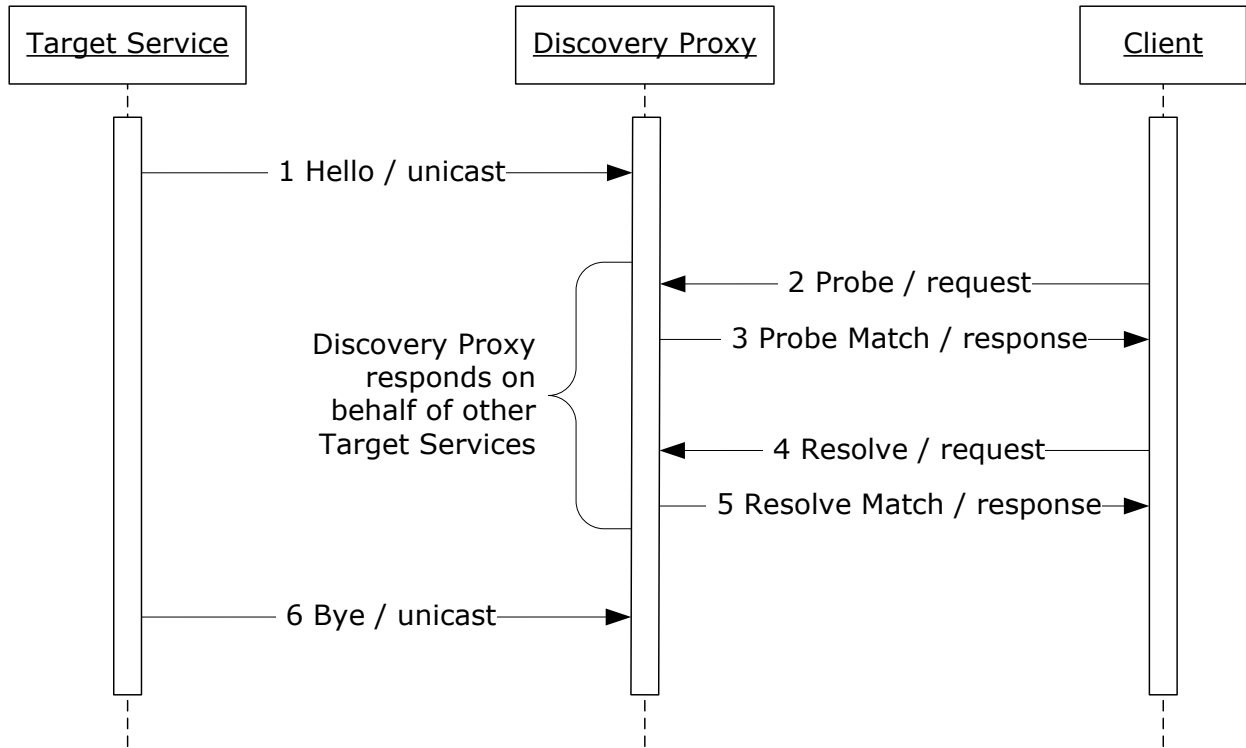
343 **Figure 2: Message exchanges in an ad hoc mode in the presence of a Discovery Proxy.**

344 A Target Service sends multicast Hello and Bye (4) and responds to matching multicast Probe and
 345 Resolve (5,7). A Client listens for multicast Hello and Bye (4) and sends multicast Probe and Resolve (5).
 346 A Discovery Proxy is also a Target Service of a well known `d:DiscoveryProxy` type and sends a
 347 multicast Hello message announcing its arrival on the network and a multicast Bye message announcing its
 348 departure from the network (1). It responds to the matching Probe and Resolve for itself (2), with a Probe
 349 Match (PM) and a Resolve Match (RM) respectively (3). If a Discovery Proxy is configured to reduce
 350 multicast traffic on the network, it listens for multicast Hello and Bye from other Target Services (4) and
 351 store/update information for corresponding Target Services (see Section 4.1.3 Discovery Proxy and 4.2.3
 352 Discovery Proxy). It responds to the multicast Probe and Resolve for other Target Services (5), with a
 353 Hello message (6) (see Section 4.1.3 Discovery Proxy), indicating the Client to switch to managed mode
 354 and to send unicast Probe and Resolve (see Section 2.2.2 Managed Mode).

355 2.2.2 Managed Mode

356 In a managed mode discovery messages are sent unicast to a Discovery Proxy. Figure 3 depicts the
 357 message exchanges between a Client, a Target Service and a Discovery Proxy in a managed mode.

358 A Target Service sends a unicast Hello message (1) to a Discovery Proxy when it joins a network (see
 359 Section 4.1.1 Target Service). A Client sends a unicast Probe request (2) to a Discovery Proxy to locate
 360 services (see Section 5.2.1 Client). A Discovery Proxy responds to a unicast Probe request with a Probe
 361 Match response (3) containing matching Target Services, if any (see Section 5.3.2 Discovery Proxy). A
 362 Client sends a unicast Resolve request (4) to a Discovery Proxy to locate a particular Target Service (see
 363 Section 6.2.1 Client). A Discovery Proxy respond to a unicast Resolve request with a Resolve Match
 364 response (4) containing the matching Target Service, if any (see Section 6.3.2 Discovery Proxy). A Target
 365 Service makes an effort to send a unicast Bye message (6) to a Discovery Proxy when it leaves a
 366 network (see Section 4.2.1 Target Service).



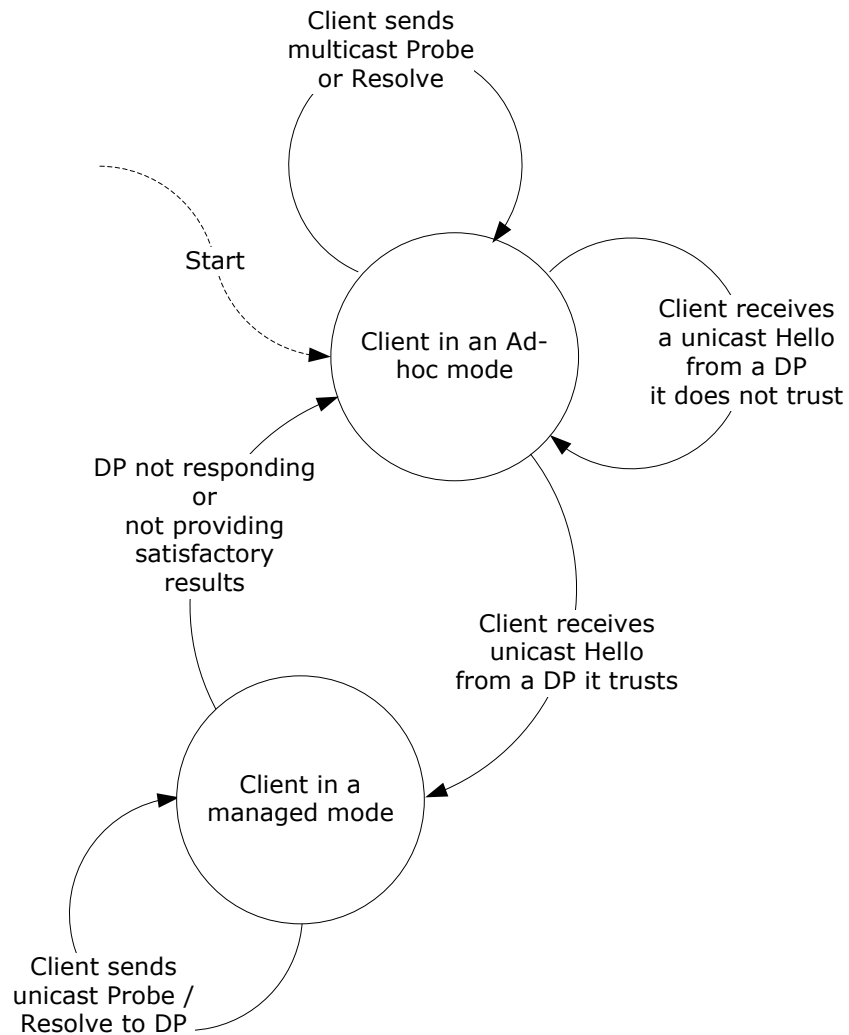
367

368 **Figure 3: Message exchanges in a managed mode.**

369 To operate in a managed mode a Target Service and a Client need an Endpoint Reference of the
 370 Discovery Proxy. A Target Service or a Client can acquire this information from a number of ways
 371 including, but not limited to explicit configuration, explicit Probe for Discovery Proxy, DNS or DHCP,
 372 specifics of which are outside the scope of this specification. One such method that reduces the traffic in
 373 an ad hoc network and allows Client to dynamically switch to managed mode is described below.

374 **2.2.3 Dynamic Mode Switching**

375 To limit multicast traffic, Clients MAY be configured to dynamically switch from an ad hoc mode to a
 376 managed mode and vice versa, depicted in Figure 4.



377

378 **Figure 4: State transitions of a Client configured to dynamically switch operational modes.**

379 By default, a Client assumes that no Discovery Proxy (DP) is available because a Discovery Proxy is an
 380 optional component and may not be present on the network. The Client operates in an ad hoc mode and
 381 listens for multicast Hello and Bye announcements, sends multicast Probe and/or Resolve messages,
 382 and listens for Probe Match and/or Resolve Match messages (see Section 2.2.1 Ad hoc Mode).

383 However, if one or more DP that provide multicast suppression are available, those DP send a unicast
 384 Hello that contains information about an endpoint that implements a well-known "discovery proxy" type
 385 `d:DiscoveryProxy` in managed mode in response to any multicast Probe or Resolve. As depicted in
 386 Figure 4, Clients listen for this signal that one or more DP are available, and for subsequent searches
 387 switch to a managed mode and instead of multicast, send Probe and Resolve messages unicast to one or
 388 more DP they trust whilst ignoring multicast Hello and Bye from Target Services.

389 In a managed mode, a Client communicates with a DP as described in Section 2.2.2 Managed Mode;
 390 using the transport information contained in the DP Hello; this is typically indicated by the scheme of a
 391 transport URI, e.g., "http:" (HTTP), "soap.udp:" (UDP [SOAP/UDP]), or other.

392 If the DP is unresponsive after `DP_MAX_TIMEOUT`, or if the Client finds the responses from the DP
 393 unsatisfactory, the Client reverts to using the multicast messages specified herein.

394 Table 4 specifies the default value for this parameter.

395 **Table 4: Default value for Discovery Proxy timeout parameter.**

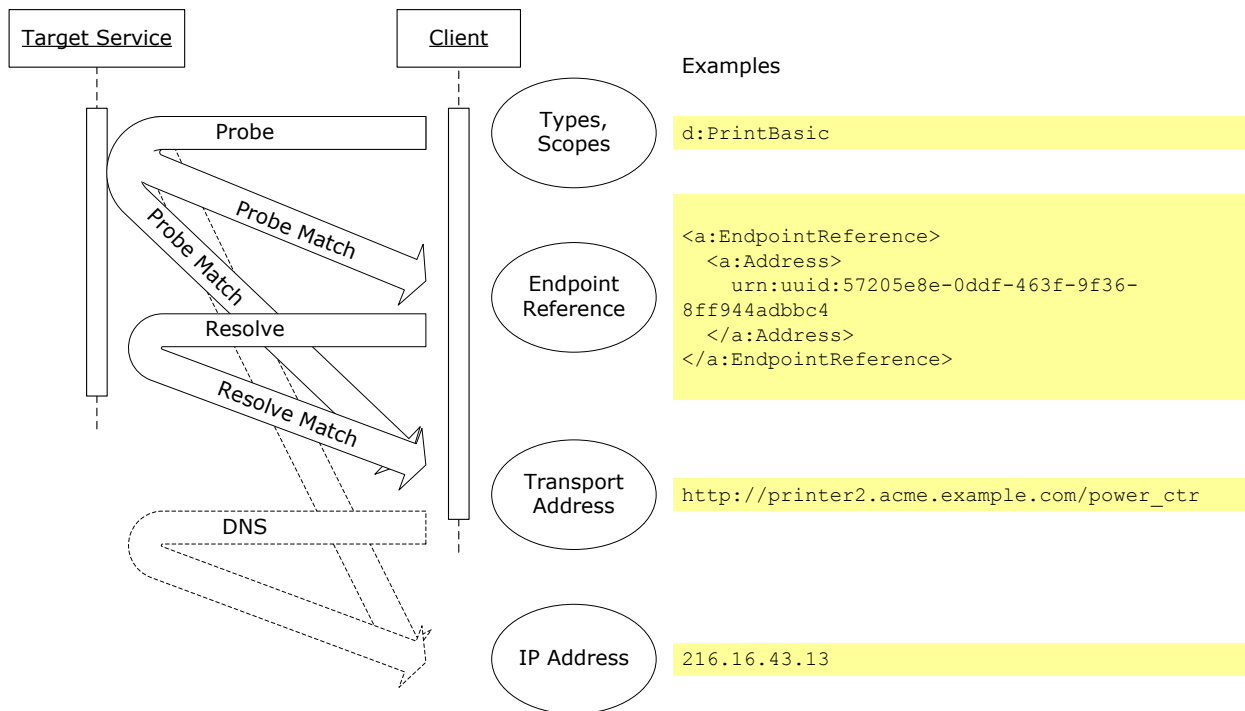
Parameter	Default Value
DP_MAX_TIMEOUT	5 seconds

396 This design minimizes discovery latency in ad hoc networks without increasing multicast traffic in
397 managed networks. To see this, note that a Client only generates multicast traffic when it sends a Probe
398 or Resolve; while a Client could Probe (or Resolve) for a DP *before* Probing (or Resolving) for a Target
399 Service of interest, this is just as expensive in a managed network (in terms of multicast network traffic)
400 as allowing the Client to Probe (or Resolve) for the Target Service directly and having the DP respond to
401 signal its presence; the reduced latency in ad hoc networks arises because the Client does not need to
402 explicitly search and wait for possible DP responses. Some Clients (for example, mobile clients frequently
403 moving within and beyond managed environments) MAY be configured to Probe first for a DP and only if
404 such Probe fails, switch to the operational mode described above. Specific means of such configuration is
405 beyond of the scope of this specification.

406 Unlike a Client, a Target Service operating in an ad hoc mode always sends (multicast) Hello and Bye,
407 and always responds to Probe and Resolve with (unicast) Probe Match and Resolve Match respectively.
408 A Target Service does not need to explicitly recognize and/or track the availability of a DP in an ad hoc
409 mode – a Target Service behaves the same way in an ad hoc mode regardless of the presence or
410 absence of a DP. This is because the Hello and Bye are too infrequent and therefore generate too little
411 multicast traffic to warrant adding complexity to Target Service behavior. However, some Target Services
412 MAY be configured to operate only in a managed mode and unicast Hello and Bye directly to a DP; these
413 would not multicast Hello and Bye or respond to Probe or Resolve; specific means of such configuration
414 are beyond the scope of this specification.

415 2.3 Conceptual Message Content

416 Conceptually, Hello, Probe Match, and Resolve Match contain different kinds of information as Figure 5
417 depicts.



418
419 **Figure 5 : Conceptual content of messages.**

420 Starting at the top of Figure 5, Probe maps from Types and/or Scopes to an Endpoint Reference [[WS-Addressing](#)] and one or more transport addresses (see Section 2.1 Endpoint References). Though not
421 depicted, Hello provides an Endpoint Reference. Resolve maps the Endpoint Reference to one or more
422 transport addresses (see Section 2.1 Endpoint References). Other address mappings may be needed,
423 e.g., DNS, but are beyond the scope of this specification.
424

425 The required components of each message are defined in detail below, but as an optimization, a Target
426 Service may short-circuit these message exchanges by including additional components; for instance, a
427 Hello may contain transport address(es) along with an Endpoint Reference, or a transport address may
428 use an IP address instead of a DNS name.

429 3 Protocol Assignments

430 3.1.1 Ad hoc mode over IP multicast

431 If IP multicast is used to send multicast messages described herein, they MUST be sent using the
432 following assignments:

- 433 • DISCOVERY_PORT: port 3702 [IANA]
- 434 • IPv4 multicast address: 239.255.255.250
- 435 • IPv6 multicast address: FF02::C (link-local scope)

436 Other address bindings MAY be defined but are beyond the scope of this specification.

437 Messages sent over UDP MUST be sent using SOAP over UDP [SOAP/UDP]. To compensate for
438 possible UDP unreliability, senders MUST use the example transmission algorithm in Appendix I of SOAP
439 over UDP. In order to improve interoperability and network efficiency use of SOAP 1.2 protocol [SOAP
440 1.2] is RECOMMENDED.

441 3.1.2 Managed mode over HTTP

442 If the messages described herein are sent unicast using HTTP protocol, they MUST be sent using SOAP
443 HTTP Binding as defined in Section 7 of SOAP 1.2 Part 2 [SOAP 1.2 Part 2].

444 3.1.3 Application Level Transmission Delay

445 As designated below, before sending some message types defined herein, a Target Service MUST wait
446 for a timer to elapse before sending the message using the bindings described above. This timer MUST
447 be set to a random value between 0 and APP_MAX_DELAY. Table 5 specifies the default value for this
448 parameter.

449 **Table 5: Default value for an application-level transmission parameter.**

Parameter	Default Value
APP_MAX_DELAY	500 milliseconds

450 The default value in Table 5 MAY be revised by other specifications.

451 *Note: The authors expect this parameter to be adjusted based on interoperability test results.*

452 Other transport bindings MAY be defined but are beyond the scope of this specification.

453 4 Hello and Bye

454 Support for messages described in this section MUST be implemented by a Target Service, MUST be
455 implemented by a Discovery Proxy, and MAY be implemented by a Client as described below.

456 4.1 Hello

457 Hello is a one-way message sent by a Target Service to announce its availability when it joins the
458 network. It is also sent by a Discovery Proxy to reduce multicast traffic on an ad hoc network.

459 The normative outline for Hello is:

```
460 <s:Envelope ... >  
461   <s:Header ... >  
462     <a:Action ... >  
463       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Hello  
464     </a:Action>  
465     <a:MessageID ... >xs:anyURI</a:MessageID>  
466     [<a:RelatesTo>  
467       xs:anyURI  
468     </a:RelatesTo>]?  
469     <a:To ... >urn:docs-oasis-open-org:ws-dd:ns:discovery:2009:01</a:To>  
470     [<d:AppSequence ... />]?  
471     ...  
472   </s:Header>  
473   <s:Body ... >  
474     <d:Hello ... >  
475       <a:EndpointReference ... </a:EndpointReference>  
476       [<d:Types>list of xs:QName</d:Types>]?  
477       [<d:Scopes>list of xs:anyURI</d:Scopes>]?  
478       [<d:XAddr>list of xs:anyURI</d:XAddr>]?  
479       <d:MetadataVersion>xs:unsignedInt</d:MetadataVersion>  
480       ...  
481     </d:Hello>  
482   </s:Body>  
483 </s:Envelope>
```

484 The following describes additional normative constraints on the outline listed above:

485 /s:Envelope/s:Header/*

486 Per SOAP [SOAP 1.1, SOAP 1.2], header blocks MAY appear in any order.

487 /s:Envelope/s:Header/a:RelatesTo

488 MUST be included only by a Discovery Proxy and if and only if Hello is sent unicast in response
489 to a multicast Probe (or Resolve). It MUST be the value of the **[message id]** property [WS-
490 Addressing] of the multicast Probe (Resolve).

491 /s:Envelope/s:Header/a:To

492 MUST be included.

493 In an ad hoc mode, it MUST be “urn:docs-oasis-open-org:ws-
494 dd:ns:discovery:2009:01” [RFC 2141] .

495 In a managed mode, it MUST be the **[address]** property [WS-Addressing] of the Endpoint
496 Reference of the Discovery Proxy.

497 /s:Envelope/s:Header/d:AppSequence
 498 MUST be included to allow ordering discovery messages from a Target Service (see Section 7
 499 Application Sequencing).
 500 SHOULD be omitted in a managed mode.

501 /s:Envelope/s:Body/d:Hello/a:EndpointReference
 502 Endpoint Reference for the Target Service (or Discovery Proxy) (see Section 2.1 Endpoint
 503 References).

504 /s:Envelope/s:Body/d:Hello/d:Types
 505 Unordered set of Types implemented by the Target Service (or Discovery Proxy).
 506 • For a Target Service, if omitted or empty, no implied value. A Target Service MAY omit Types
 507 due to security and message size considerations. In a managed mode, all supported Types
 508 SHOULD be included.
 509 • For a Discovery Proxy, MUST be included and MUST explicitly include `d:DiscoveryProxy`.

510 /s:Envelope/s:Body/d:Hello/d:Scopes
 511 Unordered set of Scopes the Target Service (or Discovery Proxy) is in, which MAY be of more
 512 than one URI scheme. If included, MUST be a set of absolute URIs, and contained URIs MUST
 513 NOT contain whitespaces. If omitted or empty, no implied value.
 514 In a managed mode, all Scopes SHOULD be included.

515 /s:Envelope/s:Body/d:Hello/d:XAddr
 516 Transport address(es) that MAY be used to communicate with the Target Service (or Discovery
 517 Proxy). Contained URIs MUST NOT contain whitespaces. If omitted or empty, no implied value.
 518 In a managed mode, all transport address(es) SHOULD be included.

519 /s:Envelope/s:Body/d:Hello/d:MetadataVersion
 520 Incremented by a positive value (≥ 1) whenever there is a change in the metadata of the Target
 521 Service. If a Target Service goes down and comes back up again, this value MAY be
 522 incremented but MUST NOT be decremented (see Section 7 Application Sequencing). Metadata
 523 includes, but is not limited to, `.. /d:Types` and `.. /d:Scopes`. By design, this value MAY be
 524 used by the Client and/or Discovery Proxy for cache control of Target Service metadata.

525 4.1.1 Target Service

526 A Target Service MUST send a Hello when any of the following occur:

- 527 • It joins a network. This MAY be detected through low-level mechanisms, such as wireless beacons,
 528 or through a change in IP connectivity on one or more of its network interfaces, or when it becomes
 529 available through one or more additional transport addresses.
- 530 • Its metadata changes (see `/s:Envelope/s:Body/d:Hello/d:MetadataVersion` above).

531 To minimize the risk of a network storm and to not overwhelm the recipient (e.g., after a network crash
 532 and recovery or power blackout and restoration), a Target Service MUST wait for a timer to elapse before
 533 sending the Hello as described in Section 3.1.3 Application Level Transmission Delay.

534 In an ad hoc mode,

- 535 • A Hello MUST be sent multicast to "urn:docs-oasis-open-org:ws-
 536 dd:ns:discovery:2009:01" [RFC 2141].
- 537 • A Target Service MAY vary the amount of metadata it includes in Hello messages (or Probe Match or
 538 Resolve Match messages), and consequently, a Client (or a Discovery Proxy) MAY receive two such
 539 messages containing the same `/s:Envelope/s:Body/*/d:MetadataVersion` but containing
 540 different metadata. If a Client (or a Discovery Proxy) chooses to cache metadata, it MAY, but is not
 541 constrained to, adopt any of the following behaviors:

- 542 - Cache the union of the previously cached and new metadata.
 - 543 - Replace the previously cached with new metadata.
 - 544 - Use some other means to retrieve more complete metadata.
- 545 However, to prevent network storms, a Client (or a Discovery Proxy) SHOULD NOT delete cached
546 metadata and SHOULD NOT repeat a Probe (or Resolve) if it detects differences in contained
547 metadata.

548 Table 6 lists an example Hello sent multicast in an ad hoc mode by the same Target Service that
549 responded with a Probe Match in Table 3.

550 **Table 6: Example Hello sent multicast in an ad hoc mode**

```

551 (01) <s:Envelope
552 (02)   xmlns:a="http://www.w3.org/2005/08/addressing"
553 (03)   xmlns:d="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01"
554 (04)   xmlns:s="http://www.w3.org/2003/05/soap-envelope" >
555 (05)   <s:Header>
556 (06)     <a:Action>
557 (07)       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Hello
558 (08)     </a:Action>
559 (09)     <a:MessageID>
560 (10)       urn:uuid:73948edc-3204-4455-bae2-7c7d0ff6c37c
561 (11)     </a:MessageID>
562 (12)     <a:To>urn:docs-oasis-open-org:ws-dd:ns:discovery:2009:01</a:To>
563 (13)     <d:AppSequence InstanceId="1077004800" MessageNumber="1" />
564 (14)   </s:Header>
565 (15)   <s:Body>
566 (16)     <d:Hello>
567 (17)       <a:EndpointReference>
568 (18)         <a:Address>
569 (19)           urn:uuid:98190dc2-0890-4ef8-ac9a-5940995e6119
570 (20)         </a:Address>
571 (21)       </a:EndpointReference>
572 (22)       <d:MetadataVersion>75965</d:MetadataVersion>
573 (23)     </d:Hello>
574 (24)   </s:Body>
575 (25) </s:Envelope>
576 (26)

```

577 Lines (06-08) indicate this is a Hello, and because Line (12) is set to the distinguished URI defined herein,
578 this is a multicast Hello. Line (13) contains an instance identifier as well as a message number; this
579 information allows the receiver to reorder Hello and Bye messages from a Target Service. Lines (17-21)
580 are identical to the corresponding lines in the Probe Match in Table 3.

581 **In a managed mode,**

- 582 • A Hello MUST be sent unicast to [address] property [WS-Addressing] of the Endpoint Reference of
583 the Discovery Proxy.
- 584 • A Target Service SHOULD include complete metadata information in the Hello message.

585 Table 7 lists an example Hello sent unicast in a managed mode to a Discovery Proxy.

586 **Table 7: Example Hello sent unicast in a managed mode to a Discovery Proxy**

```

587 (01) <s:Envelope
588 (02)   xmlns:a="http://www.w3.org/2005/08/addressing"
589 (03)   xmlns:d="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01"
590 (04)   xmlns:i="http://printer.example.org/2003/imaging"
591 (05)   xmlns:s="http://www.w3.org/2003/05/soap-envelope">
592 (06)   <s:Header>
593 (07)     <a:Action>
594 (08)       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Hello
595 (09)     </a:Action>
596 (10)     <a:MessageID>
597 (11)       urn:uuid:b10688d7-ea05-4bb1-a6bc-3aaf3be47f8e

```

```

598 (12) </a:MessageID>
599 (13) <a:To>http://example.com/DiscoveryProxy</a:To>
600 (14) </s:Header>
601 (15) <s:Body>
602 (16) <d:Hello>
603 (17) <a:EndpointReference>
604 (18) <a:Address>
605 (19) urn:uuid:98190dc2-0890-4ef8-ac9a-5940995e6119
606 (20) </a:Address>
607 (21) </a:EndpointReference>
608 (22) <d:Types>i:PrintBasic i:PrintAdvanced</d:Types>
609 (23) <d:Scopes>
610 (24) ldap:///ou=engineering,o=exampleorg,c=us
611 (25) ldap:///ou=floor1,ou=b42,ou=anytown,o=exampleorg,c=us
612 (26) http://itdept/imaging/deployment/2004-12-04
613 (27) </d:Scopes>
614 (28) <d:XAddrs>http://prn-example/PRN42/b42-1668-a</d:XAddrs>
615 (29) <d:MetadataVersion>75965</d:MetadataVersion>
616 (30) </d:Hello>
617 (31) </s:Body>
618 (32) </s:Envelope>
619 (33)

```

620 Lines (06-08) indicate this is a Hello, and Line (12) indicates it is sent unicast to Discovery Proxy over
621 HTTP. The AppSequence header is omitted here because the messages sent over HTTP are received in
622 the same order in which they are sent. The Lines (16-28) describe a single Target Service and they are
623 identical to corresponding lines (24-36) in the Probe Match in Table 3. This Hello message sent in a
624 managed mode contains complete information, Lines (16-28), about the Target Service, as opposed to
625 the one sent in the ad hoc mode, Lines (17-22) in Table 6.

626 4.1.2 Client

627 In an ad hoc mode,

- 628 • To minimize the need to Probe, Clients SHOULD listen for Hello messages and store (or update)
629 information for the corresponding Target Services.
- 630 • If a Client receives a Hello message from a Discovery Proxy in response to a multicast Probe (or
631 Resolve) (see Section 4.1.3 Discovery Proxy), the Client SHOULD switch to a managed mode and
632 send unicast Probe (or Resolve) to the Discovery Proxy (see Section 2.2.3 Dynamic Mode
633 Switching).

634 4.1.3 Discovery Proxy

635 In an ad hoc mode,

- 636 • A Discovery Proxy MUST send a Hello for itself (as a Target Service of `d:DiscoveryProxy` type)
637 as described in Section 4.1.1 Target Service.
- 638 • A Discovery Proxy MAY be configured to reduce multicast traffic on an ad hoc network, in this
639 capacity:
 - 640 • A Discovery Proxy MUST listen for multicast Hello messages and store (or update) information
641 for the corresponding Target Services.
 - 642 • A Discovery Proxy MUST listen for multicast Probe (and Resolve). In response to any multicast
643 Probe (or multicast Resolve) from a Client, a Discovery Proxy MUST send a unicast Hello to the
644 Client and SHOULD send the Hello without waiting for a timer to elapse.

645 In a managed mode,

- 646 • A Discovery Proxy MUST listen for unicast Hello messages and store (or update) information for the
647 corresponding Target Services.

648 4.2 Bye

649 Bye is a one-way message sent by a Target Service when it is preparing to leave the network.

650 The normative outline for Bye is:

```
651 <s:Envelope ... >
652   <s:Header ... >
653     <a:Action ... >
654       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Bye
655     </a:Action>
656     <a:MessageID ... >xs:anyURI</a:MessageID>
657     <a:To ...>urn:docs-oasis-open-org:ws-dd:ns:discovery:2009:01</a:To>
658     [<d:AppSequence ... />]?
659     ...
660   </s:Header>
661   <s:Body ... >
662     <d:Bye ... >
663       <a:EndpointReference ... </a:EndpointReference>
664       [<d:Types>list of xs:QName</d:Types>]?
665       [<d:Scopes>list of xs:anyURI</d:Scopes>]?
666       [<d:XAddrs>list of xs:anyURI</d:XAddrs>]?
667       [<d:MetadataVersion>xs:unsignedInt</d:MetadataVersion>]?
668       ...
669     </d:Bye>
670   </s:Body>
671 </s:Envelope>
```

672 The following describes additional normative constraints on the outline listed above:

673 /s:Envelope/s:Header/*

674 Per SOAP [SOAP 1.1, SOAP 1.2], header blocks MAY appear in any order.

675 /s:Envelope/s:Header/a:To

676 As constrained for Hello (see Section 4.1 Hello).

677 /s:Envelope/s:Header/d:AppSequence

678 As constrained for Hello (see Section 4.1 Hello).

679 /s:Envelope/s:Body/d:Bye/a:EndpointReference

680 Endpoint Reference for the Target Service (see Section 2.1 Endpoint References).

681 /s:Envelope/s:Body/d:Bye/d:Types

682 As constrained for Hello (see Section 4.1 Hello).

683 /s:Envelope/s:Body/d:Bye/d:Scopes

684 As constrained for Hello (see Section 4.1 Hello).

685 /s:Envelope/s:Body/d:Bye/d:XAddrs

686 Transport address(es) on which the Target Service (or Discovery Proxy) is no longer available.

687 Contained URIs MUST NOT contain whitespaces. If omitted or empty, no implied value.

688 /s:Envelope/s:Body/d:Bye/d:MetadataVersion

689 As constrained for Hello (see Section 4.1 Hello). If omitted, no implied value.

690 4.2.1 Target Service

691 A Target Service SHOULD send a Bye message when it is preparing to leave a network, such as when it
692 will no longer be accessible through one or more of its advertised transport addresses, or in a controlled
693 shutdown. (A Target Service MUST NOT send a Bye message when its metadata changes.)

694 A Target Service MAY send the Bye without waiting for a timer to elapse.

695 **In an ad hoc mode,**

- 696 • A Bye MUST be sent multicast to "urn:docs-oasis-open-org:ws-
697 dd:ns:discovery:2009:01" [RFC 2141].

698 Table 8 lists an example Bye message sent multicast in an ad hoc mode corresponding to the Hello in
699 Table 6.

700 **Table 8 Example Bye message sent multicast in an ad hoc mode.**

```
701 (01) <s:Envelope  
702 (02)   xmlns:a="http://www.w3.org/2005/08/addressing"  
703 (03)   xmlns:d="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01"  
704 (04)   xmlns:s="http://www.w3.org/2003/05/soap-envelope" >  
705 (05)   <s:Header>  
706 (06)     <a:Action>  
707 (07)       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Bye  
708 (08)     </a:Action>  
709 (09)     <a:MessageID>  
710 (10)       urn:uuid:337497fa-3b10-43a5-95c2-186461d72c9e  
711 (11)     </a:MessageID>  
712 (12)     <a:To>urn:docs-oasis-open-org:ws-dd:ns:discovery:2009:01</a:To>  
713 (13)     <d:AppSequence InstanceId="1077004800" MessageNumber="4" />  
714 (14)   </s:Header>  
715 (15)   <s:Body>  
716 (16)     <d:Bye>  
717 (17)       <a:EndpointReference>  
718 (18)         <a:Address>  
719 (19)           urn:uuid:98190dc2-0890-4ef8-ac9a-5940995e6119  
720 (20)         </a:Address>  
721 (21)       </a:EndpointReference>  
722 (22)     </d:Bye>  
723 (23)   </s:Body>  
724 (24) </s:Envelope>  
725 (25)
```

726 Lines (06-08) indicate this is a Bye, and like the Hello in Table 6, the distinguished URI in Line (12)
727 indicates it is a multicast Bye.

728 The sequence information in Line (13) indicates this message is to be ordered after the Hello in Table 6
729 because the Bye has a larger message number than the Hello within the same instance identifier. Note
730 that the Body (Lines 16-22) is an abbreviated form of the corresponding information in the Hello; when a
731 Target Service leaves a network, it is sufficient to send the stable identifier to indicate the Target Service
732 is no longer available.

733 **In a managed mode,**

- 734 • A Bye MUST be sent unicast to [address] property [WS-Addressing] of the Endpoint Reference of the
735 Discovery Proxy.

736 Table 9 lists an example Bye message corresponding to the Hello message in Table 7, sent unicast in a
737 managed mode to a Discovery Proxy.

738 **Table 9: Example Bye message sent unicast in a managed mode to a Discovery Proxy.**

```
739 (01) <s:Envelope  
740 (02)   xmlns:a="http://www.w3.org/2005/08/addressing"  
741 (03)   xmlns:d="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01"  
742 (04)   xmlns:s="http://www.w3.org/2003/05/soap-envelope">  
743 (05)   <s:Header>  
744 (06)     <a:Action>  
745 (07)       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Bye  
746 (08)     </a:Action>  
747 (09)     <a:MessageID>  
748 (10)       urn:uuid:cceb5804-1bcc-4721-bef3-dd688763b6aa  
749 (11)     </a:MessageID>  
750 (12)     <a:To>http://example.com/DiscoveryProxy</a:To>
```

```

751 (13) </s:Header>
752 (14) <s:Body>
753 (15)   <d:Bye>
754 (16)     <a:EndpointReference>
755 (17)       <a:Address>
756 (18)         urn:uuid:98190dc2-0890-4ef8-ac9a-5940995e6119
757 (19)       </a:Address>
758 (20)     </a:EndpointReference>
759 (21)   </d:Bye>
760 (22) </s:Body>
761 (23) </s:Envelope>
762 (24)

```

763 Lines (06-08) indicate this is a Bye, and like Hello in Table 7, Line (12) indicates that it is sent unicast to a
764 Discovery Proxy over HTTP. Like Hello in Table 7, the application sequencing information is omitted
765 because the messages sent unicast over HTTP are received in the same order in which they are sent.
766 Like Bye in Table 10 the Body (Lines 15-21) is an abbreviated form of the corresponding information in
767 the Hello.

768 4.2.2 Client

769 **In an ad hoc mode,** Clients SHOULD listen for Bye messages, marking or removing corresponding
770 information as invalid. Clients MAY wish to retain information associated with a Target Service that has
771 left the network, for instance if the Client expects the Target Service to rejoin the network at some point in
772 the future. Conversely, Clients MAY discard information associated with a Target Service at any time,
773 based on, for instance, preset maximums on the amount of memory allocated for this use, lack of
774 communication to the Target Service, preferences for other Target Service Types or Scopes, and/or other
775 application-specific preferences.

776 4.2.3 Discovery Proxy

777 **In an ad hoc mode,**

- 778 • A Discovery Proxy SHOULD send a Bye for itself (as a Target Service of `d:DiscoveryProxy` type)
779 when it is preparing to leave the network as described in Section 4.2.1 Target Service.
- 780 • A Discovery Proxy MAY be configured to reduce multicast traffic on an ad hoc network, in this
781 capacity:
 - 782 • A Discovery Proxy MUST listen for multicast Bye messages, marking or removing corresponding
783 information as invalid.

784 **In a managed mode,**

- 785 • A Discovery Proxy MUST listen for unicast Bye messages, marking or removing corresponding
786 information as invalid.

787 Note that both in an ad hoc mode and a managed mode, a Discovery Proxy MAY retain information
788 associated with a Target Service that has left the network, for instance if the Discovery Proxy expects the
789 Target Service to rejoin the network at some point in the future. Conversely, Discovery Proxy MAY
790 discard information associated with a Target Service at any time, based on, for instance, preset
791 maximums on the amount of memory allocated for this use, lack of communication to the Target Service,
792 preferences for other Target Service Types or Scopes, and/or other application-specific preferences.

793 5 Probe and Probe Match

794 To find Target Services by the Type of the Target Service, a Scope in which the Target Service resides,
795 both, or simply all Target Services, a Client sends a Probe.

796 Support for messages described in this section MUST be implemented by a Target Service, MUST be
797 implemented by a Discovery Proxy, and MAY be implemented by a Client as described below.

798 5.1 Matching Types and Scopes

799 A Probe includes zero, one, or two constraints on matching Target Services: a set of Types and/or a set
800 of Scopes. A Probe Match MUST include a Target Service if and only if all of the Types and all of the
801 Scopes in the Probe match the Target Service.

802 A Type T1 in a Probe matches Type T2 of a Target Service if the QNames match. Specifically, T1
803 matches T2 if all of the following are true:

- 804 • The namespace [[Namespaces in XML 1.1](#)] of T1 and T2 are the same.
- 805 • The local name of T1 and T2 are the same.

806 (The namespace prefix of T1 and T2 is relevant only to the extent that it identifies the namespace.)

807 A Scope S1 in a Probe matches Scope S2 of a Target Service per the rule indicated within the Probe.
808 This specification defines the following matching rules. Other matching rules MAY be used, but if a
809 matching rule is not recognized by a receiver of the Probe, S1 does not match S2 regardless of the value
810 of S1 and/or S2.

811 <http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/rfc3986>

812 Using a case-insensitive comparison,

- 813 • The `scheme` [[RFC 3986](#)] of S1 and S2 is the same and
- 814 • The `authority` of S1 and S2 is the same and

815 Using a case-sensitive comparison,

- 816 • The `path_segments` of S1 is a `segment-wise` (not string) prefix of the `path_segments` of S2
817 and
- 818 • Neither S1 nor S2 contain the `."` segment or the `."` segment.

819 All other components (e.g., `query` and `fragment`) are explicitly excluded from comparison. S1 and
820 S2 MUST be canonicalized (e.g., unescaping escaped characters) and trailing slashes (`/`) MUST be
821 removed before using this matching rule.

822 Note: this matching rule does NOT test whether the string representation of S1 is a prefix of the string
823 representation of S2. For example, `"http://example.com/abc"` matches `"http://example.com/abc/def"`
824 using this rule but `"http://example.com/a"` does not.

825 <http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/uuid>

826 S1 and S2 are universally-unique identifier (UUID) based URN [[RFC 4122](#)] scheme URIs and each of
827 the unsigned integer fields [[RFC 4122](#)] in S1 is equal to the corresponding field in S2, or equivalently,
828 the 128 bits of the in-memory representation of S1 and S2 are the same 128 bit unsigned integer.

829 <http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/ldap>

830 Using a case-insensitive comparison, the `scheme` of S1 and S2 is `"ldap"` and the `host` and the `port`
831 [[RFC 3986](#)] of S1 and S2 are the same and the `RDNSSequence` [[RFC 4514](#)] of the `dn` [[RFC 4516](#)] of
832 S1 is a prefix of the `RDNSSequence` [[RFC 4514](#)] of the `dn` [[RFC 4516](#)] of S2.

833 <http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/strcmp0>

834 Using a case-sensitive comparison, the string representation of S1 and S2 is the same.

835 <http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/none>

836 With this rule the Probe matches the Target Service if and only if the Target Service does not have
837 any Scopes. When a Probe specifies this rule it MUST NOT contain any Scopes.

838 5.2 Probe

839 The normative outline for Probe is:

```
840 <s:Envelope ... >  
841   <s:Header ... >  
842     <a:Action ... >  
843       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Probe  
844     </a:Action>  
845     <a:MessageID ... >xs:anyURI</a:MessageID>  
846     [<a:ReplyTo ... >endpoint-reference</a:ReplyTo>]?  
847     <a:To ... >xs:anyURI</a:To>  
848     ...  
849   </s:Header>  
850   <s:Body ... >  
851     <d:Probe ... >  
852       [<d:Types>list of xs:QName</d:Types>]?  
853       [<d:Scopes [MatchBy="xs:anyURI"? ... >  
854         list of xs:anyURI  
855       </d:Scopes>]?  
856       ...  
857     </d:Probe>  
858   </s:Body>  
859 </s:Envelope>
```

860 The following describes additional normative constraints on the outline listed above:

861 /s:Envelope/s:Header/*

862 Per SOAP [SOAP 1.1, SOAP 1.2], header blocks MAY appear in any order.

863 /s:Envelope/s:Header/a:ReplyTo

864 If included, MUST be of type a:EndpointReferenceType [WS-Addressing]. If omitted, implied
865 value of the [reply endpoint] property [WS-Addressing] is
866 "http://www.w3.org/2005/08/addressing/anonymous".

867 /s:Envelope/s:Header/a:ReplyTo/a:Address

868 If the value is "http://www.w3.org/2005/08/addressing/anonymous", [reply endpoint]
869 property is defined by the underlying transport. For example, if the Probe was received over UDP
870 using the assignments listed in Section 3.1.1 Ad hoc mode over IP multicast, the [reply
871 endpoint] is the IP source address and port number of the Probe transport header [SOAP/UDP].

872 /s:Envelope/s:Header/a:To

- 873 • If sent to a Target Service, MUST be "urn:docs-oasis-open-org:ws-
874 dd:ns:discovery:2009:01" [RFC 2141].
- 875 • If sent to a Discovery Proxy, MUST be the [address] property of the Endpoint Reference for
876 the Discovery Proxy, e.g., as contained in a Hello from the Discovery Proxy.

877 /s:Envelope/s:Body/d:Probe/d:Types

878 If omitted or empty, implied value is any Type.

879 /s:Envelope/s:Body/d:Probe/d:Scopes

880 If included, MUST be a list of absolute URIs, and contained URIs MUST NOT contain
881 whitespaces. The contained URIs MAY be of more than one URI scheme. If omitted or empty,
882 implied value is any Scope.

883 /s:Envelope/s:Body/d:Probe/d:Scopes/@MatchBy

884 If omitted, implied value is

885 " http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/rfc3986 ".

886 The value MUST be compared per RFC 3986 Section 6.2.1 Simple String Comparison [RFC
887 3986].

888 If a Target Service or a Discovery Proxy receives a unicast Probe and does not support the
889 matching rule, it MAY choose not to send a Probe Match and instead generate a fault, bound to
890 SOAP [WS-Addressing] as follows:

[action]	http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/fault
[Code]	s12:Sender
[Subcode]	d:MatchingRuleNotSupported
[Reason]	E.g., the matching rule specified is not supported.
[Detail]	<pre><d:SupportedMatchingRules> list of xs:anyURI </d:SupportedMatchingRules></pre>

891 To Probe for all Target Services, a Client MAY omit both /s:Envelope/s:Body/d:Probe/d:Types
892 and ./d:Scopes.

893 5.2.1 Client

894 A Client MAY send a Probe to find Target Services of a given Type and/or in a given Scope or to find
895 Target Services regardless of their Types or Scopes.

896 In an ad hoc mode,

- 897 • A Probe is a one-way message.
- 898 • A Probe MUST be sent multicast to "urn:docs-oasis-open-org:ws-
899 dd:ns:discovery:2009:01" [RFC 2141].

900 In an ad hoc network a Client may not know in advance how many Target Services (if any) will send
901 Probe Match therefore the Client MAY adopt either of the following behaviors:

- 902 • Wait for a sufficient number of Probe Match messages.
- 903 • Repeat the Probe several times until the Client is convinced that no further Probe Match messages
904 will be received. The Client MUST use the same value for the [message id] property [WS-
905 Addressing] in all copies of the Probe.

906 If a Client knows a transport address of a Target Service, the Probe MAY be sent unicast to that address.

907 Table 2 lists an example Probe message sent multicast by a Client searching for a printer in an ad hoc
908 mode.

909 In a managed mode,

- 910 • A Probe is a request message.
- 911 • A Probe MUST be sent unicast to [address] property [WS-Addressing] of the Endpoint Reference of
912 the Discovery Proxy.

913 Table 10 lists an example Probe message sent unicast to a Discovery Proxy by a Client searching for a
914 printer in a managed mode.

915 Table 10: Example Probe sent unicast to a Discovery Proxy in a managed mode.

```
916 (01) <s:Envelope
917 (02)   xmlns:a="http://www.w3.org/2005/08/addressing"
918 (03)   xmlns:d="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01"
919 (04)   xmlns:i="http://printer.example.org/2003/imaging"
920 (05)   xmlns:s="http://www.w3.org/2003/05/soap-envelope" >
921 (06)   <s:Header>
922 (07)     <a:Action>
923 (08)       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Probe
924 (09)     </a:Action>
```

```

925 (10) <a:MessageID>
926 (11) urn:uuid:d78c2d8d-1123-4a51-a814-955efdded812
927 (12) </a:MessageID>
928 (13) <a:To>http://example.com/DiscoveryProxy</a:To>
929 (14) </s:Header>
930 (15) <s:Body>
931 (16) <d:Probe>
932 (17) <d:Types>i:PrintBasic</d:Types>
933 (18) <d:Scopes
934 (19) MatchBy="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/ldap" >
935 (20) ldap:///ou=engineering,o=examplecom,c=us
936 (21) </d:Scopes>
937 (22) </d:Probe>
938 (23) </s:Body>
939 (24) </s:Envelope>
940 (25)

```

941 Lines (07-09) in Table 10 indicate this message is a Probe, and Line (13) indicates it is being sent to a
942 Discovery Proxy over HTTP.

943 Lines (17-21) specify two constants on the Target Services and they are identical to the corresponding
944 Lines (17-21) in Table 2.

945 5.2.2 Target Service

946 **In an ad hoc mode,**

- 947 • A Target Service MUST listen for multicast Probe messages and respond as described in Section
- 948 5.3.1 Target Service.
- 949 • A Target Service MAY listen for unicast Probe requests at its transport address(es) (see Section 2.1
- 950 Endpoint References) and respond to them as described in Section 5.3.1 Target Service.

951 5.2.3 Discovery Proxy

952 **In an ad hoc mode,**

- 953 • A Discovery Proxy MUST listen for multicast Probe messages for itself and respond as described in
- 954 Section 5.3.2 Discovery Proxy.
- 955 • A Discovery Proxy MAY be configured to reduce multicast traffic on an ad hoc network, in this
- 956 capacity, a Discovery Proxy MUST listen for multicast Probe for other Target Services and respond to
- 957 them with a Hello message as described in Section 4.1.3 Discovery Proxy.

958 **In a managed mode,**

- 959 • A Discovery Proxy MUST listen for unicast Probe request and respond to them as described in
- 960 Section 5.3.2 Discovery Proxy.

961 5.3 Probe Match

962 Probe Match is sent by a Target Service or a Discovery Proxy in response to a Probe.

963 The normative outline for Probe Match is:

```

964 <s:Envelope ... >
965 <s:Header ... >
966 <a:Action ... >
967 http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/ProbeMatches
968 </a:Action>
969 <a:MessageID ... >xs:anyURI</a:MessageID>
970 <a:RelatesTo ... >xs:anyURI</a:RelatesTo>
971 <a:To ... >xs:anyURI</a:To>
972 [<d:AppSequence ... />]?
973 ...
974 </s:Header>
975 <s:Body ... >

```

```

976 <d:ProbeMatches ... >
977   [<d:ProbeMatch ... >
978     <a:EndpointReference> ... </a:EndpointReference>
979     [<d:Types>list of xs:QName</d:Types>]?
980     [<d:Scopes>list of xs:anyURI</d:Scopes>]?
981     [<d:XAddr>list of xs:anyURI</d:XAddr>]?
982     <d:MetadataVersion>xs:unsignedInt</d:MetadataVersion>
983     ...
984   </d:ProbeMatch>]*
985   ...
986 </d:ProbeMatches>
987 </s:Body>
988 </s:Envelope>

```

989 The following describes additional normative constraints on the outline listed above:

990 /s:Envelope/s:Header/*

991 Per SOAP [SOAP 1.1, SOAP 1.2], header blocks MAY appear in any order.

992 /s:Envelope/s:Header/a:RelatesTo

993 MUST be the value of the [message id] property [WS-Addressing] of the Probe.

994 /s:Envelope/s:Header/a:To

995 If the [reply endpoint] property [WS-Addressing] of the corresponding Probe is the IP source
 996 address and port number of the Probe transport header (e.g., when the a:ReplyTo header block
 997 was omitted from the corresponding Probe), the value of this header block MUST be
 998 "http://www.w3.org/2005/08/addressing/anonymous".

999 /s:Envelope/s:Header/d:AppSequence

1000 MUST be included to allow ordering discovery messages from a Target Service (see Section 7
 1001 Application Sequencing).

1002 SHOULD be omitted in a managed mode.

1003 /s:Envelope/s:Body/d:ProbeMatches

1004 Matching Target Services.

- 1005 • If this Probe Match was sent by a Target Service, this element will contain one
 1006 d:ProbeMatch child. (If Target Service doesn't match the Probe, the Target Service does
 1007 not send a Probe Match at all.)
- 1008 • If this Probe Match was sent by a Discovery Proxy, this element will contain zero or more
 1009 d:ProbeMatch children. (Discovery Proxies always respond to Probe.)

1010 /s:Envelope/s:Body/d:ProbeMatches/d:ProbeMatch/a:EndpointReference

1011 Endpoint Reference for the Target Service (see Section 2.1 Endpoint References).

1012 /s:Envelope/s:Body/d:ProbeMatches/d:ProbeMatch/d:Types

1013 See /s:Envelope/s:Body/d:Hello/d:Types in Section 4.1 Hello.

1014 /s:Envelope/s:Body/d:ProbeMatches/d:ProbeMatch/d:Scopes

1015 See /s:Envelope/s:Body/d:Hello/d:Scopes in Section 4.1 Hello.

1016 /s:Envelope/s:Body/d:ProbeMatches/d:ProbeMatch/d:XAddr

1017 Transport address(es) that MAY be used to communicate with the Target Service (or Discovery
 1018 Proxy). Contained URIs MUST NOT contain whitespaces. If a Target Service (or Discovery
 1019 Proxy) has transport addresses (see Section 2.1 Endpoint References) at least one transport
 1020 address MUST be included. If omitted or empty, no implied value.

1021 /s:Envelope/s:Body/d:ProbeMatches/d:ProbeMatch/d:MetadataVersion

1022 See /s:Envelope/s:Body/d:Hello/d:MetadataVersion in Section 4.1 Hello.

1023 5.3.1 Target Service

1024 In an ad hoc mode,

- 1025 • If a Target Service receives a Probe that match, it MUST respond with a Probe Match message. If the
1026 Target Service receives more than one copy of the Probe as determined by the **[message id]**
1027 property **[WS-Addressing]**, it SHOULD respond only once. A Target Service MUST wait for a timer to
1028 elapse after receiving a Probe and before sending a Probe Match as described in Section 3.1.3
1029 Application Level Transmission Delay. The Probe Match MUST be unicast to the **[reply endpoint]**
1030 property **[WS-Addressing]** of the Probe.
- 1031 • If a Target Service receives a Probe and does not match the Probe, it MUST NOT respond with a
1032 Probe Match.

1033 Table 3 lists an example Probe Match message sent in response to the multicast Probe listed in Table 2.

1034 5.3.2 Discovery Proxy

1035 In an ad hoc mode,

- 1036 • If a Discovery Proxy receives a Probe for itself as determined by the presence of
1037 `d:DiscoveryProxy` in the Types, it MUST respond with a Probe Match message and MUST wait
1038 for a timer to elapse (see Section 3.1.3 Application Level Transmission Delay). The Probe Match
1039 MUST be unicast to the **[reply endpoint]** property **[WS-Addressing]** of the Probe.
- 1040 • A Discovery Proxy MAY be configured to reduce multicast traffic on an ad hoc network, in this
1041 capacity, if a Discovery Proxy receives a Probe for other Target Services it MUST respond with a
1042 Hello (see Section 4.1.3 Discovery Proxy).

1043 In a managed mode,

- 1044 • If a Discovery Proxy receives a Probe request it MUST respond with a Probe Match message without
1045 waiting for a timer to elapse. The Probe Match SHOULD include complete metadata information
1046 about the matching Target Services. However, the Probe Match MAY contain zero matches if the
1047 Discovery Proxy has no matching Target Services.

1048 Table 11 lists an example Probe Match message sent by the Discovery Proxy in response to the Probe
1049 message in Table 10.

1050 **Table 11: Example Probe Match sent in response to the managed Probe in Table 10**

```
1051 (01) <s:Envelope  
1052 (02)   xmlns:a="http://www.w3.org/2005/08/addressing"  
1053 (03)   xmlns:d="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01"  
1054 (04)   xmlns:i="http://printer.example.org/2003/imaging"  
1055 (05)   xmlns:s="http://www.w3.org/2003/05/soap-envelope" >  
1056 (06)   <s:Header>  
1057 (07)     <a:Action>  
1058 (08)       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/ProbeMatches  
1059 (09)     </a:Action>  
1060 (10)     <a:MessageID>  
1061 (11)       urn:uuid:7e5bb4ee-621a-4ea6-b326-3db7d99ddb47  
1062 (12)     </a:MessageID>  
1063 (13)     <a:RelatesTo>  
1064 (14)       urn:uuid:d78c2d8d-1123-4a51-a814-955efdded812  
1065 (15)     </a:RelatesTo>  
1066 (16)   </s:Header>  
1067 (17)   <s:Body>  
1068 (18)     <d:ProbeMatches>  
1069 (19)       <d:ProbeMatch>  
1070 (20)         <a:EndpointReference>  
1071 (21)           <a:Address>  
1072 (22)             urn:uuid:98190dc2-0890-4ef8-ac9a-5940995e6119  
1073 (23)           </a:Address>  
1074 (24)         </a:EndpointReference>  
1075 (25)       <d:Types>i:PrintBasic i:PrintAdvanced</d:Types>  
1076 (26)     </d:Scopes>
```

```

1077 (27) ldap:///ou=engineering,o=examplecom,c=us
1078 (28) ldap:///ou=floor1,ou=b42,ou=anytown,o=examplecom,c=us
1079 (29) http://itdept/imaging/deployment/2004-12-04
1080 (30) </d:Scopes>
1081 (31) <d:XAddr>http://prn-example/PRN42/b42-1668-a</d:XAddr>
1082 (32) <d:MetadataVersion>75965</d:MetadataVersion>
1083 (33) </d:ProbeMatch>
1084 (34) <d:ProbeMatch>
1085 (35) <a:EndpointReference>
1086 (36) <a:Address>
1087 (37) urn:uuid:70eda11c-200a-4a5e-b60e-d6793e77ace3
1088 (38) </a:Address>
1089 (39) </a:EndpointReference>
1090 (40) <d:Types>i:PrintBasic</d:Types>
1091 (41) <d:Scopes>
1092 (42) ldap:///ou=engineering,o=examplecom,c=us
1093 (43) ldap:///ou=floor1,ou=b42,ou=anytown,o=examplecom,c=us
1094 (44) http://itdept/imaging/deployment/2008-10-16
1095 (45) </d:Scopes>
1096 (46) <d:XAddr>http://prn-example/PRN42/b42-1668-b</d:XAddr>
1097 (47) <d:MetadataVersion>23654</d:MetadataVersion>
1098 (48) </d:ProbeMatch>
1099 (49) </d:ProbeMatches>
1100 (50) </s:Body>
1101 (51) </s:Envelope>
1102 (52)

```

1103 Lines (07-09) in Table 11 indicate this message is a Probe Match; and Lines (13-15) indicate that it is a
1104 response to the Probe message in Table 10. Since this Probe Match message was sent over HTTP in
1105 response to the Probe message and since messages sent over HTTP are received in the order they are
1106 sent, it does not contain a header that identifies the instance number and message number like Line (19)
1107 in Table 3.

1108 Lines (20-32) describe a Target Service and they are identical to the corresponding lines (24-36) in Table
1109 3.

1110 Lines (35-47) describe another Target Service, a basic printer service; that match the Probe in Table 10.

1111 6 Resolve and Resolve Match

1112 To locate a Target Service, i.e., to retrieve its transport address(es), a Client sends a Resolve.
1113 Support for messages described in this section MUST be implemented by a Target Service, MUST be
1114 implemented by a Discovery Proxy and MAY be implemented by a Client as described below.

1115 6.1 Matching Endpoint Reference

1116 A Resolve includes a constraint on matching Target Service: an Endpoint Reference [WS-Addressing]. A
1117 Resolve Match MUST include a Target Service if and only if the Endpoint Reference in the Resolve
1118 match the Target Service.

1119 An Endpoint Reference E1 in a Resolve matches Endpoint Reference E2 of a Target Service if the
1120 [address] property [WS-Addressing] of E1 matches the [address] property [WS-Addressing] of E2 per
1121 Section 6 of RFC 3986 [RFC 3986].

1122 6.2 Resolve

1123 The normative outline for Resolve is:

```
1124 <s:Envelope ... >  
1125   <s:Header ... >  
1126     <a:Action ... >  
1127       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/Resolve  
1128     </a:Action>  
1129     <a:MessageID ... >xs:anyURI</a:MessageID>  
1130     [<a:ReplyTo ... >endpoint-reference</a:ReplyTo>]?  
1131     <a:To ... >xs:anyURI</a:To>  
1132     ...  
1133   </s:Header>  
1134   <s:Body>  
1135     <d:Resolve ... >  
1136       <a:EndpointReference ... </a:EndpointReference>  
1137       ...  
1138     </d:Resolve>  
1139   </s:Body>  
1140 </s:Envelope>
```

1141 The following describes additional normative constraints on the outline above:

1142 /s:Envelope/s:Header/*

1143 Per SOAP [SOAP 1.1, SOAP 1.2], header blocks MAY appear in any order.

1144 /s:Envelope/s:Header/a:ReplyTo

1145 As constrained for Probe (see Section 5.2 Probe).

1146 /s:Envelope/s:Header/a:To

1147 As constrained for Probe (see Section 5.2 Probe).

1148 /s:Envelope/s:Body/d:Resolve/a:EndpointReference

1149 Endpoint Reference for the Target Service (see Section 2.1 Endpoint References).

1150 6.2.1 Client

1151 A Client MAY send a Resolve to retrieve network transport information for a Target Service if it has an
1152 Endpoint Reference [WS-Addressing] for the Target Service.

- 1153 **In an ad hoc mode,**
- 1154 • A Resolve is a one-way message.
- 1155 • A Resolve MUST be sent multicast to "urn:docs-oasis-open-org:ws-
- 1156 dd:ns:discovery:2009:01" [RFC 2141].
- 1157 **In a managed mode,**
- 1158 • A Resolve MUST be sent unicast to [address] property [WS-Addressing] of the Endpoint Reference
- 1159 of the Discovery Proxy.

1160 6.2.2 Target Service

- 1161 **In an ad hoc mode,**
- 1162 • A Target Service MUST listen for multicast Resolve messages and respond to them as described in
- 1163 Section 6.3.1 Target Service.

1164 6.2.3 Discovery Proxy

- 1165 **In an ad hoc mode,**
- 1166 • A Discovery Proxy MUST listen for multicast Resolve messages for itself and respond to them as
- 1167 described in Section 6.3.2 Discovery Proxy.
- 1168 • A Discovery Proxy MAY be configured to reduce multicast traffic on an ad hoc network, in this
- 1169 capacity, a Discovery Proxy MUST listen for multicast Resolve for other Target Services and respond
- 1170 to them with a Hello message as described in Section 4.1.3 Discovery Proxy.
- 1171 **In a managed mode,**
- 1172 • A Discovery Proxy MUST listen for unicast Resolve requests and respond to them as described in
- 1173 Section 6.3.2 Discovery Proxy.

1174 6.3 Resolve Match

1175 Resolve Match is sent by a Target Service or a Discovery Proxy in response to a Resolve.

1176 The normative outline for Resolve Match is:

```

1177 <s:Envelope ... >
1178   <s:Header ... >
1179     <a:Action ... >
1180       http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/ResolveMatches
1181     </a:Action>
1182     <a:MessageID ... >xs:anyURI</a:MessageID>
1183     <a:RelatesTo ... >xs:anyURI</a:RelatesTo>
1184     <a:To ... >xs:anyURI</a:To>
1185     [<d:AppSequence ... />]?
1186     ...
1187   </s:Header>
1188   <s:Body ... >
1189     <d:ResolveMatches ... >
1190       [<d:ResolveMatch ... >
1191         <a:EndpointReference ... </a:EndpointReference>
1192         [<d:Types>list of xs:QName</d:Types>]?
1193         [<d:Scopes>list of xs:anyURI</d:Scopes>]?
1194         [<d:XAddrs>list of xs:anyURI</d:XAddrs>]?
1195         <d:MetadataVersion>xs:unsignedInt</d:MetadataVersion>
1196         ...
1197       </d:ResolveMatch>]?
1198       ...
1199     </d:ResolveMatches>
1200   </s:Body>
1201 </s:Envelope>

```

1202 The following describes additional normative constraints on the outline listed above:

1203 /s:Envelope/s:Header/*
 1204 Per SOAP [SOAP 1.1, SOAP 1.2], header blocks MAY appear in any order.
 1205 /s:Envelope/s:Header/a:RelatesTo
 1206 MUST be the value of the [message id] property [WS-Addressing] of the Resolve.
 1207 /s:Envelope/s:Header/a:To
 1208 As constrained for Probe Match (see Section 5.3 Probe Match).
 1209 /s:Envelope/s:Header/d:AppSequence
 1210 As constrained for Probe Match (see Section 5.3 Probe Match).
 1211 /s:Envelope/s:Body/d:ResolveMatches
 1212 Matching Target Service.
 1213 /s:Envelope/s:Body/d:ResolveMatches/d:ResolveMatch/a:EndpointReference
 1214 Endpoint Reference for the Target Service (see Section 2.1 Endpoint References).
 1215 /s:Envelope/s:Body/d:ResolveMatches/d:ResolveMatch/d:Types
 1216 See /s:Envelope/s:Body/d:Hello/d:Types in Section 4.1 Hello.
 1217 /s:Envelope/s:Body/d:ResolveMatches/d:ResolveMatch/d:Scopes
 1218 See /s:Envelope/s:Body/d:Hello/d:Scopes in Section 4.1 Hello.
 1219 /s:Envelope/s:Body/d:ResolveMatches/d:ResolveMatch/d:XAddr
 1220 As constrained for Probe Match (see Section 5.3 Probe Match).
 1221 /s:Envelope/s:Body/d:ResolveMatches/d:ResolveMatch/d:MetadataVersion
 1222 See /s:Envelope/s:Body/d:Hello/d:MetadataVersion in Section 4.1 Hello.

1223 6.3.1 Target Service

1224 In an ad hoc mode,

- 1225 • If a Target Service receives a Resolve that matches it MUST respond with a Resolve Match
 1226 message. If the Target Service receives more than one copy of the Resolve as determined by the
 1227 [message id] property [WS-Addressing], it SHOULD respond only once. The Resolve Match MUST
 1228 be unicast to the [reply endpoint] property [WS-Addressing] of the Resolve without waiting for a
 1229 timer to elapse.
- 1230 • If a Target Service receives a Resolve that does not match, it MUST NOT respond with a Resolve
 1231 Match.

1232 6.3.2 Discovery Proxy

1233 In an ad hoc mode,

- 1234 • If a Discovery Proxy receives a Resolve for itself, it MUST respond with a Resolve Match message. If
 1235 the Discovery Proxy receives more than one copy of the Resolve as determined by the [message id]
 1236 property [WS-Addressing], it SHOULD respond only once. The Resolve Match MUST be unicast to
 1237 the [reply endpoint] property [WS-Addressing] of the Resolve without waiting for a timer to elapse.
- 1238 • A Discovery Proxy MAY be configured to reduce multicast traffic on an ad hoc network, in this
 1239 capacity, if a Discovery Proxy receives a Resolve for other Target Services, it SHOULD respond with
 1240 a Hello (see Section 4.1.3 Discovery Proxy).

1241 In a managed mode,

- 1242 • If a Discovery Proxy receives a Resolve request and it has a Target Service that matches the
 1243 Resolve, it MUST respond with a Resolve Match message. The Resolve Match SHOULD include
 1244 complete metadata information about the matching Target Service. However, the Resolve Match
 1245 MAY contain zero matches if the Discovery Proxy has no matching Target Service.

7 Application Sequencing

1246

1247 The Application Sequencing header block allows a receiver to order messages that contain this header
1248 block though they might have been received out of order. It is used by this specification to allow ordering
1249 messages from a Target Service; it is also expected that this header block will be useful in other
1250 applications.

1251 The normative outline for the application sequence header block is:

```
1252 <s:Envelope ...>  
1253   <s:Header ...>  
1254     <d:AppSequence InstanceId="xs:unsignedInt"  
1255       [SequenceId="xs:anyURI"]?  
1256       MessageNumber="xs:unsignedInt"  
1257       ... />  
1258   </s:Header>  
1259   <s:Body ...> ... </s:Body>  
1260 </s:Envelope>
```

1261 The following describes normative constraints on the outline listed above:

1262 /s:Envelope/s:Header/d:AppSequence/@Instanceid

1263 MUST be incremented by a positive value (≥ 1) each time the service has gone down, lost state,
1264 and came back up again. SHOULD NOT be incremented otherwise. Means to set this value
1265 include, but are not limited to:

- 1266 • A counter that is incremented on each 'cold' boot
- 1267 • The boot time of the service, expressed as seconds elapsed since midnight January 1, 1970

1268 /s:Envelope/s:Header/d:AppSequence/@Sequenceid

1269 Identifies a sequence within the context of an instance identifier. If omitted, implied value is null.
1270 MUST be unique within ./@Instanceid. MUST be compared per RFC 3986 Section 6.2.1 Simple
1271 String Comparison [RFC 3986]. The ordering of messages with different value of Sequenceid but
1272 the same value of Instanceid within the Application Sequencing Header block is undefined.

1273 /s:Envelope/s:Header/d:AppSequence/@MessageNumber

1274 Identifies a message within the context of a sequence identifier and an instance identifier. MUST
1275 be incremented by a positive value (≥ 1) for each message sent. Transport-level retransmission
1276 MUST preserve this value.

1277 Other components of the outline above are not further constrained by this specification.

1278 8 Security

1279 8.1 Security Model

1280 This specification does not require that endpoints participating in the discovery process be secure.
1281 However, this specification RECOMMENDS that security be used to mitigate various types of attacks (see
1282 Section 8.3 Security Considerations).

1283 If a Target Service wishes to secure Hello, Bye, Probe Match and/or Resolve Match, it SHOULD use the
1284 compact signature format defined in Section 8.2 Compact Signature Format. A Client MAY choose to
1285 ignore Hello, Bye, Probe Match, and/or Resolve Match if it cannot verify the signature.

1286 If a Client wishes to secure Probe and Resolve, it SHOULD use the compact signature format defined in
1287 Section 8.2 Compact Signature Format. A Target Service MAY chose to ignore received Probe and/or
1288 Resolve if it cannot verify the signature.

1289 There is no requirement for a Target Service to respond to a Probe (or Resolve) if any of the following are
1290 true:

- 1291 • The Target Service is in a different administrative domain than the Client, and the Probe (or
1292 Resolve) was sent as multicast, or
- 1293 • The Target Service fails to verify the signature contained in the Probe (or Resolve).

1294 To avoid participating in a Distributed Denial of Service attack, a Target Service or Discovery Proxy
1295 SHOULD NOT respond to a message without a valid signature and MUST NOT respond to a message
1296 without a valid signature if the **[reply endpoint]** is not
1297 "<http://www.w3.org/2005/08/addressing/anonymous>".

1298 A Client MAY discard a Probe Match (or Resolve Match) if any of the following are true:

- 1299 • The Probe Match (or Resolve Match) is received MATCH_TIMEOUT seconds or more later than
1300 the last corresponding Probe was sent, or
- 1301 • The Client fails to verify the signature contained in the Probe Match (or Resolve Match).

1302 Table 12 specifies the default value for the MATCH_TIMEOUT parameter.

1303 **Table 12: Default value for an application-level parameter.**

Parameter	Default Value
MATCH_TIMEOUT	APP_MAX_DELAY + 100 milliseconds

1304 If a Target Service has multiple credentials, it SHOULD send separate Hello, Bye, Probe Match, and/or
1305 Resolve Match using different credentials to sign each.

1306 The same security requirements as defined for a Target Service apply to a Discovery Proxy.

1307 8.2 Compact Signature Format

1308 This section defines the compact signature format for signing UDP unicast and multicast messages. A
1309 sender creates the compact signature from a full XML Signature [[XML Sig](#)] for optimized transmission. A
1310 receiver expands the compact signature to a full XML Signature [[XML Sig](#)] for verification.

1311 To minimize the number of XML namespace declarations in messages, the following global attribute is
1312 defined:

1313 @d:Id

1314 An alternate ID reference mechanism with the same meaning as @wsu:Id [[WS-Security](#)]. This
1315 attribute MAY be used to identify which message parts are signed by the compact signature.

1316 The compact signature itself is of the following form:

```
1317 <d:Security ... >
1318   [<d:Sig Scheme="xs:anyURI"
1319     [KeyId="xs:base64Binary"]?
1320     Refs="xs:IDREFS"
1321     [PrefixList="xs:NMTOKENS"]?
1322     Sig="xs:base64Binary"
1323     ... />]?
1324   ...
1325 </d:Security>
```

1326 d:Security

1327 A sub-class of the `wsse:Security` header block [WS-Security] that has the same processing
1328 model and rules but is restricted in terms of content and usage. The `d:Sig` child element
1329 provides a compact message signature. Its format is a compact form of XML Signature. To
1330 process the signature, the compact form is parsed, and an XML Signature `ds:SignedInfo`
1331 block is created and used for signature verification.

1332 d:Security/@s11:mustUnderstand | d:Security/@s12:mustUnderstand

1333 Processing of the `d:Security` header block is not mandatory; therefore, the `d:Security`
1334 header block SHOULD NOT be marked `mustUnderstand` with a value of "true".

1335 d:Security/d:Sig/@Scheme

1336 The governing scheme of the signature. Provides exactly one algorithm for digests and
1337 signatures.

1338 The value MUST be compared per RFC 3986 Section 6.2.1 Simple String Comparison [RFC
1339 3986].

1340 d:Security/d:Sig/@Scheme = "http://docs.oasis-open.org/ws-
1341 dd/ns/discovery/2009/01/rsa"

1342 Exclusive C14N is used for all canonicalization, SHA1 is used for all digests, and Signatures use
1343 RSA. Specifically:

1344 `http://www.w3.org/2001/10/xml-exc-c14n#`

1345 `http://www.w3.org/2000/09/xmldsig#sha1`

1346 `http://www.w3.org/2000/09/xmldsig#rsa-sha1`

1347 d:Security/d:Sig/@KeyId

1348 The key identifier of the signing token in Base64-encoded form. MUST be specified if a public key
1349 token is used. If included, MUST be the Thumbprint (SHA-1 hash of the raw octets) of the signing
1350 token. If omitted, the semantics are undefined.

1351 d:Security/d:Sig/@Refs

1352 Parts of the message that have been canonicalized and digested. Each part is referenced by
1353 `@d:Id` (see above). Only the immediate children of the security header, top-level SOAP header
1354 blocks (`/s:Envelope/s:Header/*`) other than the security header
1355 (`/s:Envelope/d:Security`), and the full SOAP Body (`/s:Envelope/s:Body`) can be
1356 referenced in this list. The value is a space-separated list of IDs to elements within the message.

1357 d:Security/d:Sig/@PrefixList

1358 If present, MUST NOT be empty and MUST be the value of **InclusiveNamespaces PrefixList**
1359 parameter [EXC-C14N] passed to the exclusive canonicalization method. If omitted, no implied
1360 value. The **IncludeNamespaces PrefixList** MUST include the prefixes that declare the XML
1361 namespace for the Types (`/s:Envelope/s:Body/* /d:Types`) and MAY include other content
1362 of the type `xs:QName` in the message, as the exclusive canonicalization method excludes (see
1363 Exclusive XML Canonicalization Section 1.3 [EXC-C14N]) the namespaces that are not visibly
1364 utilized.

1365 d:Security/d:Sig/@Sig

1366 The Base64-encoded value of the signature.

1367 Table 13 lists an example compact signature.

1368 **Table 13: Example compact signature.**

```
1369 (01) <d:Sig xmlns:d="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01"  
1370 (02) Scheme="http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01/rsa"  
1371 (03) KeyId="Dx42/9g="  
1372 (04) Refs="ID1"  
1373 (05) PrefixList="i"  
1374 (06) Sig="ru5Ef76xGz5Y5IB2iAzDuMvR5Tg=" />  
1375 (07)
```

1376 A compact signature is expanded into an XML Signature `ds:SignedInfo` using the following pseudo-
1377 code. The `SignedInfo` block within the expanded XML Signature **MUST NOT** use whitespaces inside the
1378 character content. This ensures that each party can compute a consistent digest value.

- 1379 1. Create an XML Signature `ds:SignedInfo` block. Because canonicalization includes the
1380 namespace prefix, this **MUST** use an XML namespace prefix of "ds" so each party can compute a
1381 consistent digest value.
- 1382 2. Populate the block with the appropriate canonicalization and algorithm blocks based on the scheme
1383 in `d:Security/d:Sig/@Scheme`.
 - 1384 • First add a `ds:CanonicalizationMethod` element with `Algorithm` attribute set to
1385 `http://www.w3.org/2001/10/xml-exc-c14n#`.
 - 1386 • Next add a `ds:SignatureMethod` element with `Algorithm` attribute value set to
1387 `http://www.w3.org/2000/09/xmldsig#rsa-sha1`.
- 1388 3. For each ID in `d:Security/d:Sig/@Refs` create a corresponding XML Signature Reference
1389 element to the identified part (using URI fragments) annotated with the canonicalization and digest
1390 algorithms from the scheme in `d:Security/d:Sig/@Scheme`. Note that individual digests need to
1391 be computed on the fly.
 - 1392 • Add a `ds:Reference` element.
 - 1393 • The `@URI` attribute's value is a "#" followed by the specified ID.
 - 1394 • Inside the `ds:Reference` element add a `ds:Transforms` element that contains a
1395 `ds:Transform` element indicating the selected canonicalization algorithm.
 - 1396 • If `d:Security/d:Sig/@PrefixList` is present, create an `ec:InclusiveNamespaces`
1397 element inside `ds:Transform` element. Because canonicalization includes the namespace
1398 prefix, this **MUST** use an XML namespace prefix of "ec" so each party can compute a consistent
1399 digest value. Add `PrefixList` attribute to `ec:InclusiveNamespaces` element with value
1400 equal to that of `d:Security/d:Sig/@PrefixList`.
 - 1401 • Inside the `ds:Reference` element add a `ds:DigestMethod` element with `Algorithm` attribute
1402 set to `http://www.w3.org/2000/09/xmldsig#sha1`.
 - 1403 • Inside the `ds:Reference` element add a `ds:DigestValue` element with the computed digest
1404 value of the part represented by this `ds:Reference` element.
- 1405 4. `d:Security/d:Sig/@KeyId`, if present, can be processed as a `SecurityTokenReference`
1406 [[WS-Security](#)] with an embedded `KeyIdentifier` [[WS-Security](#)] specifying the indicated value.
1407 While it isn't required to construct a `wsse:SecurityTokenReference` element, the following steps
1408 illustrate how one would be created:
- 1409 5. Create a `wsse:SecurityTokenReference` element.
- 1410 6. Within this, add a `wsse:KeyIdentifier` element with the value of the `KeyId` attribute's value.

1411 Table 14 lists the expanded signature obtained by applying above steps to the corresponding compact
1412 form in Table 13.

1413 **Table 14: Example expanded signature corresponding to the compact form in Table 13.**

```
1414 (01) <ds:Signature  
1415 (02)   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"  
1416 (03)   xmlns:wss="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-  
1417   wssecurity-secext-1.0.xsd" >  
1418 (04)   <ds:SignedInfo><ds:CanonicalizationMethod  
1419   Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" /><ds:SignatureMethod  
1420   Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" /><ds:Reference  
1421   URI="#ID1" ><ds:Transforms><ds:Transform  
1422   Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"><ec:InclusiveNamespaces  
1423   PrefixList="i" xmlns:ec="http://www.w3.org/2001/10/xml-exc-  
1424   c14n#" /></ds:Transform></ds:Transforms><ds:DigestMethod  
1425   Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"  
1426   /><ds:DigestValue>ODE3NDkyNzI5</ds:DigestValue></ds:Reference></ds:SignedInfo>  
1427 (05)   <ds:SignatureValue>ru5Ef76xGz5Y5IB2iAzDuMvR5Tg=</ds:SignatureValue>  
1428 (06)   <ds:KeyInfo>  
1429 (07)     <wss:SecurityTokenReference>  
1430 (08)       <wss:KeyIdentifier>Dx42/9g=</wss:KeyIdentifier>  
1431 (09)     </wss:SecurityTokenReference>  
1432 (10)   </ds:KeyInfo>  
1433 (11) </ds:Signature>  
1434 (12)
```

1435 Once expanded, compute the final signature, and verify that it matches.

1436 8.3 Security Considerations

1437 Message discovery, both announcements and searches, are subject to a wide variety of attacks.
1438 Therefore communication SHOULD be secured using the mechanisms described in Section 8.2 Compact
1439 Signature Format.

1440 The following list summarizes common classes of attacks and mitigations provided by this protocol:

- 1441 • **Message alteration** – An attacker can change message content. To prevent this, the message
1442 SHOULD be signed. The Body and all relevant headers SHOULD be included in the signature.
1443 Specifically, the Application Sequencing header, WS-Addressing [WS-Addressing] headers and any
1444 headers identified in Endpoint References SHOULD be signed together with the Body to "bind" them
1445 together.
- 1446 • **Availability (Denial of Service)** – An attacker can send messages that consume resources. To
1447 prevent this, a signature assures that a message is of genuine origin. To avoid unnecessary
1448 processing, the signature SHOULD be validated before performing beginning any significant
1449 processing of message content.
- 1450 • **Replay** – An attacker can resend a valid message and cause duplicate processing. To prevent this, a
1451 replayed message is detected by a duplicate [message id] property [WS-Addressing] or an older
1452 Application Sequencing header and SHOULD be discarded. Implementations MAY also use the
1453 Timestamps mechanism defined in [WS-Security] to protect against the replay attack. In that case the
1454 wsu:Timestamp element [WS-Security] SHOULD be included in the d:Security header and
1455 SHOULD be signed.
- 1456 • **Spoofing** – An attacker sends a message that pretends to be of genuine origin. To prevent this, the
1457 signature SHOULD be unique to the sender.

1458 To provide mitigation against other possible attacks, e.g., message disclosure, mechanisms defined in
1459 WS-Security [WS-Security], WS-SecureConversation [WS-SecureConversation], and/or WS-Trust [WS-
1460 Trust] MAY be applied.

1461 If a Client communicates with a Discovery Proxy, the Client SHOULD establish end-to-end security with
1462 the Discovery Proxy; to improve the efficiency of security operations, the Client SHOULD establish a
1463 security context using the mechanisms described in WS-Trust [WS-Trust] and WS-SecureConversation

1464 [\[WS-SecureConversation\]](#). In such cases, separate derived keys SHOULD be used to secure each
1465 message.

1466 9 Conformance

1467 To be conformant with this specification an endpoint MUST implement at least one of the roles; Target
1468 Service, Discovery Proxy, and Client; and MAY implement it in more than one of the modes; ad hoc and
1469 managed; however, for each implemented role and mode, it MUST implement them as specified herein.

1470 An implementation is not conformant with this specification if it fails to satisfy one or more of the MUST or
1471 REQUIRED level requirements defined herein for the roles and modes it implements.

1472 Normative text within this specification takes precedence over normative outlines, which in turn take
1473 precedence over the XML Schema [[XML Schema Part 1](#), [Part 2](#)] and WSDL [[WSDL 1.1](#)] descriptions,
1474 which in turn take precedence over examples.

1475

A. Acknowledgements

1476 The following individuals have participated in the creation of this specification and are gratefully
1477 acknowledged:

1478 **Participants:**

1479 Geoff Bullen, Microsoft Corporation
1480 Steve Carter, Novell
1481 Dan Conti, Microsoft Corporation
1482 Doug Davis, IBM
1483 Scott deDeugd, IBM
1484 Oliver Dohndorf, Technische Universitat Dortmund

1485 Dan Driscoll, Microsoft Corporation
1486 Colleen Evans, Microsoft Corporation
1487 Max Feingold, Microsoft Corporation
1488 Travis Grigsby, IBM
1489 Francois Jammes, Schneider Electric
1490 Ram Jeyaraman, Microsoft Corporation
1491 Mike Kaiser, IBM
1492 Supun Kamburugamuva, WSO2
1493 Devon Kemp, Canon Inc.
1494 Akira Kishida, Canon Inc.
1495 Jan Krueger, Technische Universitaet Dortmund

1496 Mark Little, Red Hat
1497 Dr. Ingo Lueck, Technische Universitaet Dortmund
1498 Jonathan Marsh, WSO2
1499 Carl Mattocks
1500 Antoine Mensch
1501 Jaime Meritt, Progress Software
1502 Vipul Modi, Microsoft Corporation
1503 Anthony Nadalin, IBM
1504 Tadahiro Nakamura, Canon Inc.
1505 Masahiro Nishio, Canon Inc.
1506 Toby Nixon, Microsoft Corporation
1507 Shin Ohtake, Fuji Xerox Co., Ltd.
1508 Venkat Reddy, CA
1509 Alain Regnier, Ricoh Company, Ltd.
1510 Hitoshi Sekine, Ricoh Company, Ltd.
1511 Yasuji Takeuchi, Konica Minolta Business Technologies

1512 Hiroshi Tamura, Ricoh Company, Ltd.
1513 Minoru Torii, Canon Inc.
1514 Asir S Vedamuthu, Microsoft Corporation
1515 David Whitehead, Lexmark International Inc.
1516 Don Wright, Lexmark International Inc.
1517 Prasad Yendluri, Software AG, Inc.
1518 Elmar Zeeb, University of Rostock
1519 Gottfried Zimmermann

1520

1521 **Co-Developers of the initial contributions:**

1522 This document is based on initial contributions to the OASIS WS-DD Technical Committee by the
1523 following co-developers.

1524 Gopal Kakivaya, Microsoft Corporation
1525 Devon Kemp, Canon Inc.

1526 Thomas Kuehnel, Microsoft Corporation
1527 Brad Lovering, Microsoft Corporation
1528 Bryan Roe, Intel
1529 Christopher St. John, Software AG
1530 Jeffrey Schlimmer (Editor), Microsoft Corporation
1531 Guillaume Simonnet, Microsoft Corporation
1532 Doug Walter, Microsoft Corporation
1533 Jack Weast, Intel
1534 Yevgeniy Yarmosh, Intel
1535 Prasad Yendluri, Software AG
1536

1537 **Acknowledgements of the initial contributions:**

1538 The following individuals have provided invaluable input to the original contributions and were
1539 acknowledged in the initial contributions.

1540 Don Box, Microsoft Corporation
1541 Shannon Chan, Microsoft Corporation
1542 Dan Conti, Microsoft Corporation
1543 Ken Cooper, Microsoft Corporation
1544 Mike Fenelon, Microsoft Corporation
1545 Omri Gazitt Microsoft Corporation,
1546 Bertus Greeff, Microsoft Corporation
1547 Rob Hain, Microsoft Corporation
1548 Richard Hasha, Microsoft Corporation
1549 Erin Honeycutt, Microsoft Corporation
1550 Christian Huitema, Microsoft Corporation
1551 Chris Kaler, Microsoft Corporation
1552 Umesh Madan, Microsoft Corporation
1553 Vipul Modi, Microsoft Corporation
1554 Jeff Parham, Microsoft Corporation
1555 Yaniv Pessach, Microsoft Corporation
1556 Stefan Pharies, Microsoft Corporation
1557 Dale Sather, Microsoft Corporation
1558 Matt Tavis, Microsoft Corporation

1559

B. Revision History

1560 [optional; should not be included in OASIS Standards]

Revision	Date	Editor	Changes Made
wd-01	09/16/2008	Vipul Modi	Created the initial working draft by converting the input specification to OASIS template.
wd-01	09/16/2008	Vipul Modi	Authoritative format changed to docx from doc
wd-01	09/19/2008	Vipul Modi	Adjusted the location of the document as per the format decided on 09/18/2008 during F2F meeting day 3.
wd-01	09/24/2008	Vipul Modi	Fixed broken links for cross referencing Table, Figure and Section.
wd-02	09/26/2008	Vipul Modi	<p>Incorporated proposals for the following issues:</p> <p>018 - Move Application Sequencing from Appendix to main specification</p> <p>019 - Combine security section under a single top level heading</p> <p>020 - XSD and WSDL files as separate resources</p> <p>047 - Replace reference to RFC 2396 with RFC 3986</p> <p>048 - Probe requirement in ResolveMatch section</p> <p>050 - The UUIDs URIs do not use UUID URN Namespace defined by RFC 4122</p> <p>054 -Remove support for SOAP 1.1</p> <p>058 - Remove transport specification retransmission notes in ProbeMatch and ResolveMatch sections</p> <p>059 - Follow WSDL naming conventions in naming messages and part names</p> <p>062 - Description of Scopes element for Probe does not mention that whitespace is not allowed</p> <p>063 -Clarify matching behavior for empty <d:Types>, <d:Scopes> element</p> <p>064 -Clarify matching algorithm for @MatchBy, @Scheme and @SequenceId</p> <p>065 - Terminologies should not make normative text like statements</p> <p>066 - RelationshipType attribute is not required</p> <p>067- Define KeyID content in the d:Sig</p> <p>061- Use OASIS assigned namespace</p>
wd-03	10/20/2008	Vipul Modi	<p>Incorporated the proposal for the following issues</p> <p>022 - request-response MEP for communicating with proxy</p> <p>034 - Discovery proxy and multicast suppression requirement</p> <p>035 - define protocol assignment/binding for managed mode</p>

			036 - discovery messages and managed mode 049 - forced managed mode transition for the client
cd-01	10/21/2008	Vipul Modi	Created first committee draft by from working draft 03. Removed all change bars.
wd-04	11/23/2008	Devon Kemp Vipul Modi	Created working draft 04 by applying the proposed resolutions of the following issues to CD-01 version: 007 - Old version of WS-Addressing 009 - Clarify matching rule rfc2396 078 - WS-Discovery - Transport addresses referred to as EPR
wd-05	1/13/2009	Vipul Modi	Applied the resolution of following issues to the document. 023 - Clarify use of AppSequence and related fields 079 - Too many normative statements in Section 2 Terminology and Notations 081 - Use "urn:uuid" scheme for UUID scope matching rule 086 - Example Hello sent in managed mode does not define "i" 087 - Incorrect reference to RFC 5280 088 - Using whitespaces in the expanded signature can result in different digest values 089 - Namespace of a Type can be altered in a secure discovery message 090 - Compact Signature outline does not include the datatype for Refs attribute 091 - Minor Editorial issues in Section 8.2 096 - Clarify meaning of "device leaving the network" 097 - Clarify meaning of "device joining the network" 098 - Assign an OASIS namespace for Committee Draft 2 099 - typo in URI 100 - typo in introduction 101 - typo in section 2.1 102 - typo in section 3.1 103 - typos in section 3.2 104 - clarify case where a TS doesn't specify a Type 105 - editorial changes in section 3.3 106 - redundant mentions of "one way" 107 - typos in section 3.1 (with DP) 108 - clarify "stable identifier" 118 - Add missing text for adding Algorithmsuite attribute in the expanded signature elements (editorial) 119 - Clarify that the DigestValue element inside the Reference element should be populated with the computed digest value (editorial) 120 - References update

			<p>069 - Preventing replay attack using [message id] property is impractical</p> <p>123 - Remove special "ad-hoc" scope</p> <p>141 - Editorial - Remove wildcard from the normative outline descriptions</p> <p>142 - Editorial - Remove wildcard from the normative outline descriptions</p> <p>124 - xAddrs in ResolveMatches issue</p> <p>125 - Finding services require a double roundtrip</p> <p>126 - KeyId complexity in compact signatures</p> <p>144 - replace the >= 1 symbol with the text</p> <p>145 - clarify that the security header should not be signed</p>
cd-02 (candidate)	1/21/2009	Vipul Modi	<p>Created CD-02 candidate draft from working draft 05 by accepting all changes and removing all comments.</p> <p>Applied the resolution of following issues.</p> <p>146 - Conformance must require implementing at least one of the prescribed roles</p> <p>027 - clarification on accepting unicast Probe</p> <p>Following editorial changes were made to be compliant with the OASIS document format.</p> <ul style="list-style-type: none"> * Cover Page: Previous Version was marked as N/A. * Section 2.1 Terminology is moved under Section 1.5 Terminology and named as 1.5.2 Terms and Definitions. Added a line "Defined below are the basic definitions for the terms used in this specifications." before starting the definitions. * Section 2.2 Notational Conventions- The first paragraph is moved to Section 1.5 Terminology and the second paragraph was moved to 1.5.1 and named Notational Conventions * The format of the definitions in section 1.5.2 is changed to have space in-between two definitions. * Section 2.3 XML Namespaces became Section 1.6 * Section 2.4 XSD and WSDL files became Section 1.7 * Section 2.5 Compliance became Section 9 Conformance.
cd-02 (candidate)	1/23/2009	Vipul Modi	<p>Additional editorial changes to comply with the OASIS document format.</p> <ul style="list-style-type: none"> * Corrected errors in hyperlinks in the first page of the document. * Removed "Latest Approved Version" links as suggested by OASIS TC admin. * Appendix. Acknowledgements. In the list of TC participants, removed mention of company name of Individual or Associate members per advice from OASIS TC admin. * Added the Revision History appendix section.
cd-02	1/28/2009	Vipul Modi	<p>Changed the cover page to reflect CD 02 status.</p>

pr-01	1/28/2009	Vipul Modi	Created public review 01 document from CD 02.
wd-06	2/12/2009	Vipul Modi	Includes resolution of following editorial issues. 149 - Update WS-SecureConversation and WS-Trust references to latest version 152 - Move example in Section 1 after the terminology section
wd-07	3/13/2009	Vipul Modi	Includes the resolution of the editorial issue PR-005.
wd-08	4/10/2009	Vipul Modi	Included the resolution of the editorial issue PR-007-Suggested changes to conformance sections and precedence of XSD/WSDL Included the resolution of the editorial clarification issue PR-008- WS-Discovery - Clarifications to ad hoc and managed mode definitions. Added names of 3 new TC members to acknowledgment section.
cd-03	4/14/2009	Vipul Modi	Created Committee Draft 03 document from WD-08.
cd-04	4/28/2009	Vipul Modi	Created Committee Draft 04 document from CD-03 version of the document (no changes). This is to be consistent with the version number of the schema and WSDL files.
cs-01	5/14/2009	Vipul Modi	Created Committee Specification 01 document from CD-04 version of the document.

1561