



# Web Services Make Connection (WS-MakeConnection) Version 1.1

## OASIS Standard

2 February 2009

### Specification URIs:

#### This Version:

<http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-os.pdf>  
<http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-os.html>  
<http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-os.doc> (Authoritative)

#### Previous Version:

<http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-cs-02.pdf>  
<http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-cs-02.html>  
<http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-cs-02.doc> (Authoritative)

#### Latest Version:

<http://docs.oasis-open.org/ws-rx/wsmc/v1.1/wsmc.pdf>  
<http://docs.oasis-open.org/ws-rx/wsmc/v1.1/wsmc.html>  
<http://docs.oasis-open.org/ws-rx/wsmc/v1.1/wsmc.doc>

### Technical Committee:

OASIS Web Services Reliable Exchange (WS-RX) TC

### Chairs:

Paul Fremantle <[paul@wso2.com](mailto:paul@wso2.com)>  
Sanjay Patil <[sanjay.patil@sap.com](mailto:sanjay.patil@sap.com)>

### Editors:

Doug Davis, IBM <[dug@us.ibm.com](mailto:dug@us.ibm.com)>  
Anish Karmarkar, Oracle <[Anish.Karmarkar@oracle.com](mailto:Anish.Karmarkar@oracle.com)>  
Gilbert Pilz, BEA <[gpilz@bea.com](mailto:gpilz@bea.com)>  
Steve Winkler, SAP <[steve.winkler@sap.com](mailto:steve.winkler@sap.com)>  
Ümit Yalçınalp, SAP <[umit.yalcinalp@sap.com](mailto:umit.yalcinalp@sap.com)>

### Related Work:

This specification replaces or supercedes:

- WS-MakeConnection v1.0

### Declared XML Namespaces:

<http://docs.oasis-open.org/ws-rx/wsmc/200702>

### Abstract:

This specification (WS-MakeConnection) describes a protocol that allows messages to be transferred between nodes implementing this protocol by using a transport-specific back-channel. The protocol is described in this specification in a transport-independent manner allowing it to be implemented using different network technologies. To support interoperable Web services, a SOAP binding is defined within this specification.

The protocol defined in this specification depends upon other Web services specifications for the identification of service endpoint addresses and policies. How these are identified and retrieved are detailed within those specifications and are out of scope for this document.

By using the XML [XML], SOAP [SOAP 1.1], [SOAP 1.2] and WSDL [WSDL 1.1] extensibility model, SOAP-based and WSDL-based specifications are designed to be composed with each other to define a rich Web services environment. As such, WS-MakeConnection by itself does not define all the features required for a complete messaging solution. WS-MakeConnection is a building block that is used in conjunction with other specifications and application-specific protocols to accommodate a wide variety of requirements and scenarios related to the operation of distributed Web services.

**Status:**

This document was last revised or approved by the WS-RX Technical Committee 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-rx/>.

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-rx/ipr.php>).

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

---

## Notices

Copyright © OASIS® 1993–2009. 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", WS-MakeConnection, WSMC, WSRM, WS-RX are trademarks of [OASIS](http://www.oasis-open.org/who/trademark.php), 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.....	5
1.1	Terminology.....	5
1.2	Normative.....	6
1.3	Non-Normative.....	6
1.4	Namespace.....	7
1.5	Conformance.....	8
2	MakeConnection Model.....	9
2.1	Glossary.....	10
2.2	Protocol Preconditions.....	10
2.3	Example Message Exchange.....	10
3	MakeConnection.....	13
3.1	MakeConnection Anonymous URI.....	13
3.2	MakeConnection Message.....	13
3.3	MessagePending.....	15
3.4	MakeConnection Policy Assertion.....	15
4	Faults.....	17
4.1	Unsupported Selection.....	18
4.2	Missing Selection.....	18
5	Security Considerations.....	20
	Appendix A. Schema.....	21
	Appendix B. WSDL.....	22
	Appendix C. Message Examples.....	23
	Appendix C.1 Example use of MakeConnection.....	23
	Appendix D. Acknowledgments.....	27

---

# 1 Introduction

The primary goal of this specification is to create a mechanism for the transfer of messages between two endpoints when the sending endpoint is unable to initiate a new connection to the receiving endpoint. It defines a mechanism to uniquely identify non-addressable endpoints, and a mechanism by which messages destined for those endpoints can be delivered. It also defines a SOAP binding that is required for interoperability. Additional bindings can be defined.

This mechanism is extensible allowing additional functionality, such as security, to be tightly integrated. This specification integrates with and complements the WS-ReliableMessaging[WS-RM], WS-Security [WS-Security], WS-Policy [WS-Policy], and other Web services specifications. Combined, these allow for a broad range of reliable, secure messaging options.

## 1.1 Terminology

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

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

- The syntax appears as an XML instance, but values in italics indicate data types instead of values.
- Characters are appended to elements and attributes to indicate cardinality:
  - "?" (0 or 1)
  - "\*" (0 or more)
  - "+" (1 or more)
- The character "|" is used to indicate a choice between alternatives.
- The characters "[" and "]" are used to indicate that contained items are to be treated as a group with respect to cardinality or choice.
- An ellipsis (i.e. "...") indicates a point of extensibility that allows other child or attribute content specified in this document. Additional children elements and/or attributes MAY be added at the indicated extension points but they MUST NOT contradict the semantics of the parent and/or owner, respectively. If an extension is not recognized it SHOULD be ignored.
- XML namespace prefixes (see section 1.4) are used to indicate the namespace of the element being defined.

Elements and Attributes defined by this specification are referred to in the text of this document using XPath 1.0 [XPATH 1.0] expressions. Extensibility points are referred to using an extended version of this syntax:

- An element extensibility point is referred to using {any} in place of the element name. This indicates that any element name can be used, from any namespace other than the `wsmc:` namespace.
- An attribute extensibility point is referred to using @{any} in place of the attribute name. This indicates that any attribute name can be used, from any namespace other than the `wsmc:` namespace.

## 39 1.2 Normative

- 40 **[KEYWORDS]** S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," RFC  
41 2119, Harvard University, March 1997  
42 <http://www.ietf.org/rfc/rfc2119.txt>
- 43 **[SOAP 1.1]** W3C Note, "SOAP: Simple Object Access Protocol 1.1," 08 May 2000.  
44 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
- 45 **[SOAP 1.2]** W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging Framework" June  
46 2003.  
47 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>
- 48 **[URI]** T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI):  
49 Generic Syntax," RFC 3986, MIT/LCS, U.C. Irvine, Xerox Corporation, January  
50 2005.  
51 <http://ietf.org/rfc/rfc3986>
- 52 **[UUID]** P. Leach, M. Mealling, R. Salz, "A Universally Unique Identifier (UUID) URN  
53 Namespace," RFC 4122, Microsoft, Refactored Networks - LLC, DataPower  
54 Technology Inc, July 2005  
55 <http://www.ietf.org/rfc/rfc4122.txt>
- 56 **[WSDL 1.1]** W3C Note, "Web Services Description Language (WSDL 1.1)," 15 March 2001.  
57 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>
- 58 **[WS-Addressing]** W3C Recommendation, "Web Services Addressing 1.0 - Core", May 2006.  
59 <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>  
60 W3C Recommendation, "Web Services Addressing 1.0 – SOAP Binding", May  
61 2006.  
62 <http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/>
- 63 **[WS-RM]** OASIS Standard, "Web Services Reliable Messaging (WS-ReliableMessaging),"  
64 February 2009.  
65 <http://docs.oasis-open.org/ws-rx/wsrmp/200702/wsrmp-1.2-spec-os.doc>
- 66 **[WS-RM Policy]** OASIS Standard, "Web Services Reliable Messaging Policy Assertion( WS-RM  
67 Policy)", February 2009.  
68 <http://docs.oasis-open.org/ws-rx/wsrmp/200702/wsrmp-1.2-spec-os.doc>
- 69 **[XML]** W3C Recommendation, "Extensible Markup Language (XML) 1.0 (Fourth  
70 Edition)", September 2006.  
71 <http://www.w3.org/TR/REC-xml/>
- 72 **[XML-ns]** W3C Recommendation, "Namespaces in XML," 14 January 1999.  
73 <http://www.w3.org/TR/1999/REC-xml-names-19990114/>
- 74 **[XML-Schema Part1]** W3C Recommendation, "XML Schema Part 1: Structures," October 2004.  
75 <http://www.w3.org/TR/xmlschema-1/>
- 76 **[XML-Schema Part2]** W3C Recommendation, "XML Schema Part 2: Datatypes," October 2004.  
77 <http://www.w3.org/TR/xmlschema-2/>
- 78 **[XPath 1.0]** W3C Recommendation, "XML Path Language (XPath) Version 1.0," 16 November  
79 1999.  
80 <http://www.w3.org/TR/xpath>

## 81 1.3 Non-Normative

- 82 **[RDDL 2.0]** Jonathan Borden, Tim Bray, eds. "Resource Directory Description Language  
83 (RDDL) 2.0," January 2004  
84 <http://www.openhealth.org/RDDL/20040118/rddl-20040118.html>

- 85 **[RTTM]** V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance",  
86 RFC 1323, May 1992.  
87 <http://www.rfc-editor.org/rfc/rfc1323.txt>
- 88 **[SecurityPolicy]** OASIS Standard, "WS-SecurityPolicy 1.3", February 2009  
89 [http://docs.oasis-open.org/ws-sx/ws-securitypolicy/v1.3/os/ws-securitypolicy-1.3-](http://docs.oasis-open.org/ws-sx/ws-securitypolicy/v1.3/os/ws-securitypolicy-1.3-spec-os.doc)  
90 [spec-os.doc](http://docs.oasis-open.org/ws-sx/ws-securitypolicy/v1.3/os/ws-securitypolicy-1.3-spec-os.doc)
- 91 **[SecureConversation]** OASIS Standard, "WS-SecureConversation 1.4", February 2009  
92 [http://docs.oasis-open.org/ws-sx/ws-secureconversation/v1.4/os/ws-](http://docs.oasis-open.org/ws-sx/ws-secureconversation/v1.4/os/ws-secureconversation-1.4-spec-os.doc)  
93 [secureconversation-1.4-spec-os.doc](http://docs.oasis-open.org/ws-sx/ws-secureconversation/v1.4/os/ws-secureconversation-1.4-spec-os.doc)
- 94 **[Trust]** OASIS Standard "WS-Trust 1.4", February 2009  
95 <http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/os/ws-trust-1.4-spec-os.doc>
- 96 **[WS-Policy]** W3C Recommendation, "Web Services Policy 1.5 - Framework," September  
97 2007.  
98 <http://www.w3.org/TR/2007/REC-ws-policy-20070904>
- 99 **[WS-PolicyAttachment]** W3C Recommendation, "Web Services Policy 1.5 - Attachment,"  
100 September 2007.  
101 <http://www.w3.org/TR/2007/REC-ws-policy-attach-2007004>
- 102 **[WS-Security]** Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS  
103 [Web Services Security: SOAP Message Security 1.0 \(WS-Security 2004\)](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)",  
104 OASIS Standard 200401, March 2004.  
105 [http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)  
106 [security-1.0.pdf](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)
- 107  
108 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS  
109 [Web Services Security: SOAP Message Security 1.1 \(WS-Security 2004\)](http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf)", OASIS  
110 Standard 200602, February 2006.  
111 <http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>

## 112 1.4 Namespace

113 The XML namespace [[XML-ns](#)] URI that MUST be used by implementations of this specification is:

114 <http://docs.oasis-open.org/ws-rx/wsmc/200702>

115 Dereferencing the above URI will produce the Resource Directory Description Language [[RDDL 2.0](#)]  
116 document that describes this namespace.

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

119 Table 1

Prefix	Namespace
S	(Either SOAP 1.1 or 1.2)
S11	<a href="http://schemas.xmlsoap.org/soap/envelope/">http://schemas.xmlsoap.org/soap/envelope/</a>
S12	<a href="http://www.w3.org/2003/05/soap-envelope">http://www.w3.org/2003/05/soap-envelope</a>
wsmc	<a href="http://docs.oasis-open.org/ws-rx/wsmc/200702">http://docs.oasis-open.org/ws-rx/wsmc/200702</a>
wstrm	<a href="http://docs.oasis-open.org/ws-rx/wstrm/200702">http://docs.oasis-open.org/ws-rx/wstrm/200702</a>
wsa	<a href="http://www.w3.org/2005/08/addressing">http://www.w3.org/2005/08/addressing</a>
wsam	<a href="http://www.w3.org/2007/05/addressing/metadata">http://www.w3.org/2007/05/addressing/metadata</a>
wsp	<a href="http://www.w3.org/ns/ws-policy">http://www.w3.org/ns/ws-policy</a>
xs	<a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>

120 The normative schema for WS-MakeConnection can be found linked from the namespace document that  
121 is located at the namespace URI specified above.

122 All sections explicitly noted as examples are informational and are not to be considered normative.

## 123 **1.5 Conformance**

124 An implementation is not conformant with this specification if it fails to satisfy one or more of the MUST or  
125 REQUIRED level requirements defined herein. A SOAP Node MUST NOT use the XML namespace  
126 identifier for this specification (listed in section 1.4) within SOAP Envelopes unless it is conformant with this  
127 specification.

128 Normative text within this specification takes precedence over normative outlines, which in turn take  
129 precedence over the XML Schema [[XML Schema Part 1](#), [Part 2](#)] descriptions.



## 130 2 MakeConnection Model

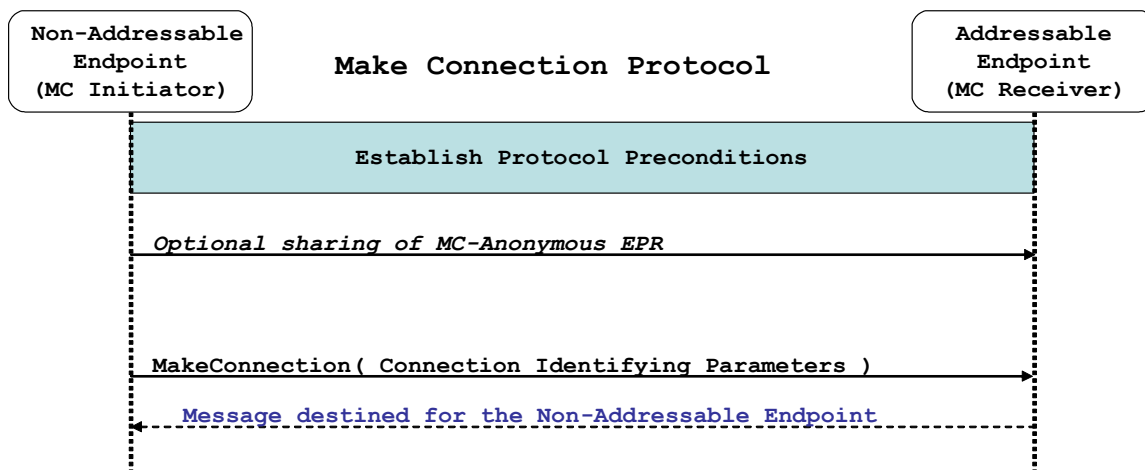
131 The WS-Addressing [WS-Addressing] specification defines the anonymous URI to identify non-  
132 addressable endpoints and to indicate a protocol-specific back-channel is to be used for any messages  
133 destined for that endpoint. For example, when used in the WS-Addressing ReplyTo EPR, the use of this  
134 anonymous URI is meant to indicate that any response message is to be transmitted on the transport-  
135 specific back-channel. In the HTTP case this would mean that any response message is sent back on the  
136 HTTP response flow.

137 In cases where the connection is still available the WS-Addressing URI is sufficient. However, in cases  
138 where the original connection is no longer available, additional mechanisms are needed. Take the situation  
139 where the original connection that carried a request message is broken and therefore is no longer  
140 available to carry a response back to the original sender. Traditionally, non-anonymous (addressable)  
141 EPRs would be used in these cases to allow for the sender of the response message to initiate new  
142 connections as needed. However, if the sender of the request message is unable (or unwilling) to accept  
143 new connections then the only option available is for it to establish a new connection for the purposes of  
144 allowing the response message to be sent. This specification defines a mechanism by which a new  
145 connection can be established.

146 The MakeConnection model consists of two key aspects:

- 147 • An optional anonymous-like URI template is defined that has similar semantics to WS-  
148 Addressing's anonymous, but also allows for each non-addressable endpoint to be uniquely  
149 identified
- 150 • A new message is defined that establishes a connection that can then be used to transmit  
151 messages to these non-addressable endpoints

152 Figure 1 below illustrates the overall flow involved in the use of MakeConnection:



153 Figure 1 – Make Connection Model

154 The `MakeConnection` message is used to establish a new connection between the two endpoints. Within  
155 the message is identifying information that is used to uniquely identify a message that is eligible for  
156 transmission.

## 157 **2.1 Glossary**

158 The following definitions are used throughout this specification:

159 **Back-channel:** When the underlying transport provides a mechanism to return a transport-protocol  
160 specific response, capable of carrying a SOAP message, without initiating a new connection, this  
161 specification refers to this mechanism as a back-channel.

162 **Endpoint:** As defined in the WS-Addressing specification; a Web service Endpoint is a (referenceable)  
163 entity, processor, or resource to which Web service messages can be addressed. Endpoint references  
164 (EPRs) convey the information needed to address a Web service Endpoint.

165 **MC Initiator** The endpoint that transmits the `MakeConnection` message – the destination endpoint for  
166 the messages being sent on the transport-specific back-channel.

167 **MC Receiver:** The endpoint that receives the `MakeConnection` message – the source endpoint for the  
168 messages being sent on the transport-specific back-channel.

169 **Receive:** The act of reading a message from a network connection.

170 **Transmit:** The act of writing a message to a network connection.

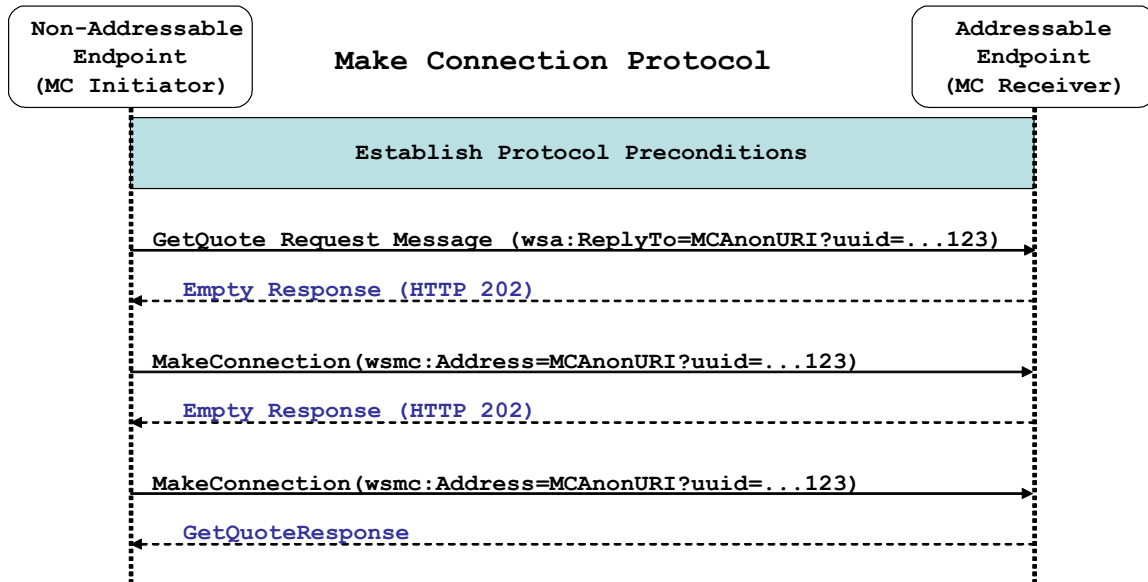
## 171 **2.2 Protocol Preconditions**

172 The correct operation of the protocol requires that a number of preconditions **MUST** be established prior to  
173 the processing of the initial sequenced message:

- 174 • The MC Receiver **MUST** be capable of accepting new incoming connections.
- 175 • The MC Initiator **MUST** be capable of creating new outgoing connections to the MC Receiver, and  
176 those connections **MUST** have a back-channel.
- 177 • If a secure exchange of messages is **REQUIRED**, then the MC Initiator and MC Receiver **MUST**  
178 have a security context.

## 179 **2.3 Example Message Exchange**

180 Figure 2 illustrates a message exchange in which the response message is delivered using  
181 `MakeConnection`.



182 Figure 2: Example WS-MakeConnection Message Exchange

183 1. The protocol preconditions are established. These include policy exchange, endpoint resolution,  
184 and establishing trust.

185 2. The client (MC Initiator) sends a GetQuote request message to the service (MC Receiver). The  
186 WS-Addressing `wsa:ReplyTo` EPR uses the MakeConnection Anonymous URI Template –  
187 indicating that if the GetQuoteResponse message is not sent back on this connection's back-  
188 channel, then the client will use MakeConnection to retrieve it.

189 3. The service receives the request message and decides to close the connection by sending back  
190 an empty response (in the HTTP case an HTTP 202 Accept is sent).

191 4. The client sends a MakeConnection message to the service. Within the MakeConnection  
192 element is the `wsmc:Address` element containing the same MakeConnection Anonymous URI  
193 used in step 2.

194 5. The service has not completed executing the GetQuote operation and decides to close the  
195 connection by sending back an empty response (in the HTTP case an HTTP 202 Accept)  
196 indicating that no messages destined for this MC Initiator are available at this time.

197 6. The client sends a second MakeConnection message to the service. Within the  
198 MakeConnection element is the `wsmc:Address` element containing the same MakeConnection  
199 Anonymous URI used in step 2.

200 7. The service uses this new connection to transmit the GetQuoteResponse message.

201 The service can assume that because the MakeConnection Anonymous URI Template was used in the  
202 `wsa:ReplyTo` EPR the client will act as an MC Initiator for the purposes of retrieving messages destined  
203 to that EPR (i.e. responses to the GetQuote). This allows the service the option of immediately releasing  
204 resources used by the original connection – knowing that the client will, at some later point in time,  
205 establish a new connection on which the GetQuoteResponse can be transmitted. Likewise, when the first  
206 MakeConnection is received by the service, it again has the option of leaving the connection open until  
207 the GetQuoteResponse is ready to be transmitted, or it can close the connection immediately knowing  
208 that the MC Initiator will retransmit the MakeConnection message at some later point in time. Since the  
209 nature and dynamic characteristics of the underlying transport and potential intermediaries are unknown in  
210 the general case, the timing of re-transmissions cannot be specified. Additionally, over-aggressive re-  
211 transmissions have been demonstrated to cause transport or intermediary flooding which are  
212 counterproductive. Consequently, implementers are encouraged to utilize adaptive mechanisms that

213 dynamically adjust re-transmission time and the back-off intervals that are appropriate to the nature of the  
214 transports and intermediaries envisioned. For the case of TCP/IP transports, a mechanism similar to that  
215 described as RTTM in RFC 1323 [[RTTM](#)] SHOULD be considered.

216 Now that the basic model has been outlined, the details of this protocol are now provided in section 3.

---

## 217 3 MakeConnection

218 The following sub-sections define the various MakeConnection features, and prescribe their usage by a  
219 conformant implementations.

### 220 3.1 MakeConnection Anonymous URI

221 When an Endpoint is not directly addressable (e.g. behind a firewall or not able to allow incoming  
222 connections), an anonymous URI in the EPR address property can indicate such an Endpoint. The WS-  
223 Addressing anonymous URI is one such anonymous URI. This specification defines a URI template (the  
224 WS-MC anonymous URI) which may be used to uniquely identify anonymous Endpoints.

```
225 http://docs.oasis-open.org/ws-rx/wsmc/200702/anonymous?id={unique-String}
```

226 The appearance of an instance of this URI template in the `wsa:Address` value of an EPR indicates a  
227 protocol-specific back-channel will be established through a mechanism such as `MakeConnection`,  
228 defined below. When using this URI template, “{unique-String}” MUST be replaced by a globally unique  
229 string (e.g a UUID value as defined by RFC4122 [UUID]). This specification does not require the use of  
230 one particular string generation scheme. This string uniquely distinguishes the Endpoint. A sending  
231 Endpoint SHOULD Transmit messages at Endpoints identified with the URI template using a protocol-  
232 specific back-channel, including but not limited to those established with a `MakeConnection` message.  
233 Note, this URI template is semantically similar to the WS-Addressing anonymous URI if a protocol-specific  
234 back-channel is available.

### 235 3.2 MakeConnection Message

236 The `MakeConnection` element is sent in the body of a one-way message that establishes a  
237 contextualized back-channel for the transmission of messages according to matching criteria (defined  
238 below). In the non-faulting case, if no matching message is available then no SOAP envelope will be  
239 returned on the back-channel. A common usage will be a client sending `MakeConnection` to a server for  
240 the purpose of receiving asynchronous response messages.

241 When the MC protocol is composed with the WS-Addressing specification, the value of the `wsa:Action`  
242 header would be:

```
243 http://docs.oasis-open.org/ws-rx/wsmc/200702/MakeConnection
```

244 The following exemplar defines the `MakeConnection` syntax:

```
245 <wsmc:MakeConnection ...>  
246   <wsmc:Address ...> xs:anyURI </wsmc:Address> ?  
247   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier> ?  
248   ...  
249 </wsmc:MakeConnection>
```

250 The following describes the content model of the `MakeConnection` element.

251 `/wsmc:MakeConnection`

252 This element allows the sender to create a transport-specific back-channel that can be used to  
253 return a message that matches the selection criteria. Endpoints MUST NOT send this element as  
254 a header block. At least one selection criteria sub-element MUST be specified – if not a  
255 `MissingSelection` fault MUST be generated.

256 `/wsmc:MakeConnection/wsmc:Address`

257 This element specifies the URI (*wsa:Address*) of the initiating Endpoint. Endpoints MUST NOT  
258 return messages on the transport-specific back-channel unless they have been addressed to this  
259 URI. This Address property and a message's WS-Addressing destination property are considered  
260 identical when they are exactly the same character-for-character. Note that URIs which are not  
261 identical in this sense may in fact be functionally equivalent. Examples include URI references  
262 which differ only in case, or which are in external entities which have different effective base URIs.

263 `/wsmc:MakeConnection/wsmc:Address/@{any}`

264 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to  
265 the element.

266 `/wsmc:MakeConnection/wsrn:Identifier`

267 This element specifies the WS-RM Sequence Identifier that establishes the context for the  
268 transport-specific back-channel. The Sequence Identifier should be compared with the Sequence  
269 Identifiers associated with the messages held by the sending Endpoint, and if there is a matching  
270 message it will be returned.

271 `/wsmc:MakeConnection/wsrn:Identifier/@{any}`

272 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to  
273 the element.

274 `/wsmc:MakeConnection/{any}`

275 This is an extensibility mechanism to allow different (extensible) types of information, based on a  
276 schema, to be passed. This allows fine-tuning of the messages to be returned, additional selection  
277 criteria included here are logically ANDed with the *Address* and/or *wsrn:Identifier*. If an  
278 extension is not supported by the Endpoint then it should generate an *UnsupportedSelection*  
279 fault.

280 `/wsmc:MakeConnection/@{any}`

281 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to  
282 the element.

283 If more than one selection criteria element is present, then the MC Receiver processing the  
284 *MakeConnection* message MUST insure that any SOAP Envelope flowing on the back-channel satisfies  
285 all of those selection criteria.

286 The management of messages that are awaiting the establishment of a back-channel to their receiving  
287 Endpoint is an implementation detail that is outside the scope of this specification. Note, however, that  
288 these messages form a class of asynchronous messages that is not dissimilar from "ordinary"  
289 asynchronous messages that are waiting for the establishment of a connection to their destination  
290 Endpoints.

291 This specification places no constraint on the types of messages that can be returned on the transport-  
292 specific back-channel. As in an asynchronous environment, it is up to the recipient of the  
293 *MakeConnection* message to decide which messages are appropriate for transmission to any particular  
294 Endpoint. However, the Endpoint processing the *MakeConnection* message MUST insure that the  
295 messages match the selection criteria as specified by the child elements of the *MakeConnection*  
296 element.

297 Since the message exchange pattern use by *MakeConnection* is untraditional, the following points need  
298 to be reiterated for clarification:

- 299 • The *MakeConnection* message is logically part of a one-way operation; there is no reply  
300 message to the *MakeConnection* itself, and any response flowing on the transport back-channel  
301 is a pending message.

- 302 • Since there is no reply message to `MakeConnection`, the WS-Addressing specific rules in  
303 section 3.4 "Formulating a Reply Message" are not used. Therefore, the value of any  
304 `wsa:ReplyTo` element in the `MakeConnection` message has no effective impact since the WS-  
305 Addressing [`reply endpoint`] property that is set by the presence of `wsa:ReplyTo` is not  
306 used.
- 307 • In the absence of any pending message, there will be no message transmitted on the transport  
308 back-channel. E.g. in the HTTP case just an `HTTP 202 Accepted` will be returned without any  
309 SOAP envelope in the HTTP response message.
- 310 • When there is a message pending, it is sent on the transport back-channel, using the connection  
311 that has been initiated by the `MakeConnection` request.

### 312 3.3 MessagePending

313 When `MakeConnection` is used, and a message is returned on the transport-specific back-channel, the  
314 `MessagePending` header SHOULD be included on the returned message as an indicator whether there  
315 are additional messages waiting to be retrieved using the same selection criteria that was specified in the  
316 `MakeConnection` element.

317 The following exemplar defines the `MessagePending` syntax:

```
318 <wsmc:MessagePending pending="xs:boolean" ...>
319   ...
320 </wsmc:MessagePending>
```

321 The following describes the content model of the `MessagePending` header block.

322 `/wsmc:MessagePending`

323       This element indicates whether additional messages are waiting to be retrieved.

324 `/wsmc:MessagePending/@pending`

325       This attribute, when set to "true", indicates that there is at least one message waiting to be  
326       retrieved. When this attribute is set to "false" it indicates there are currently no messages waiting  
327       to be retrieved.

328 `/wsmc:MessagePending/{any}`

329       This is an extensibility mechanism to allow different (extensible) types of information, based on a  
330       schema, to be passed.

331 `/wsmc:MessagePending/@{any}`

332       This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to  
333       the element.

334 The absence of the `MessagePending` header has no implication as to whether there are additional  
335 messages waiting to be retrieved.

### 336 3.4 MakeConnection Policy Assertion

337 The `MakeConnection` policy assertion indicates that the `MakeConnection` protocol (operation and the use  
338 of the `MakeConnection` URI template in `EndpointReferences`) is required for messages sent from this  
339 endpoint. This assertion has `Endpoint Policy Subject` [[WS-PolicyAttachment](#)].

340 The normative outline for the `MakeConnection` assertion is:

```
341 <wsmc:MCSupported ...> ... </wsmc:MCSupported>
```

342 The following describes the content model of the `MCSupported` element.

343 `/wsmc:MCSupported`

344 A policy assertion that specifies that the `MakeConnection` protocol is required for messages sent  
345 from this endpoint.

346 `/wsmc:MCSupported/{any}`

347 This is an extensibility mechanism to allow different (extensible) types of information, based on a  
348 schema, to be passed.

349 `/wsmc:MCSupported/@{any}`

350 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to  
351 the element.



---

## 352 4 Faults

353 Entities that generate WS-MakeConnection faults MUST include as the [action] property the default fault  
354 action IRI defined below. The value from the W3C Recommendation is below for informational purposes:

355 <http://docs.oasis-open.org/ws-rx/wsmc/200702/fault>

356 The faults defined in this section are generated if the condition stated in the preamble is met. Fault  
357 handling rules are defined in section 6 of WS-Addressing SOAP Binding.

358 The definitions of faults use the following properties:

359 [Code] The fault code.

360 [Subcode] The fault subcode.

361 [Reason] The English language reason element.

362 [Detail] The detail element(s). If absent, no detail element is defined for the fault. If more than one detail  
363 element is defined for a fault, implementations MUST include the elements in the order that they are  
364 specified.

365 Entities that generate WS-MakeConnection faults MUST set the [Code] property to either "Sender" or  
366 "Receiver". These properties are serialized into text XML as follows:

SOAP Version	Sender	Receiver
SOAP 1.1	S11:Client	S11:Server
SOAP 1.2	S:Sender	S:Receiver

367 The properties above bind to a SOAP 1.2 fault as follows:

```
368 <S:Envelope>  
369 <S:Header>  
370 <wsa:Action>  
371 http://docs.oasis-open.org/ws-rx/wsmc/200702/fault  
372 </wsa:Action>  
373 <!-- Headers elided for brevity. -->  
374 </S:Header>  
375 <S:Body>  
376 <S:Fault>  
377 <S:Code>  
378 <S:Value> [Code] </S:Value>  
379 <S:Subcode>  
380 <S:Value> [Subcode] </S:Value>  
381 </S:Subcode>  
382 </S:Code>  
383 <S:Reason>  
384 <S:Text xml:lang="en"> [Reason] </S:Text>  
385 </S:Reason>  
386 <S:Detail>  
387 [Detail]  
388 ...  
389 </S:Detail>  
390 </S:Fault>  
391 </S:Body>  
392 </S:Envelope>
```

393 The properties bind to a SOAP 1.1 fault as follows when the fault is generated as a result of processing a  
394 MakeConnection message:

```

395 <S11:Envelope>
396 <S11:Body>
397 <S11:Fault>
398 <faultcode> [Subcode] </faultcode>
399 <faultstring> [Reason] </faultstring>
400 </S11:Fault>
401 </S11:Body>
402 </S11:Envelope>

```

## 403 4.1 Unsupported Selection

404 The QName of the unsupported element(s) are included in the detail.

405 Properties:

406 [Code] Receiver

407 [Subcode] wsmc:UnsupportedSelection

408 [Reason] The extension element used in the message selection is not supported by the MakeConnection receiver

410 [Detail]

```

411 <wsmc:UnsupportedSelection> xs:QName </wsmc:UnsupportedSelection>+

```

Generated by	Condition	Action Upon Generation	Action Upon Receipt
MakeConnection receiver	In response to a MakeConnection message containing a selection criteria in the extensibility section of the message that is not supported	Unspecified.	Unspecified.

## 412 4.2 Missing Selection

413 The MakeConnection element did not contain any selection criteria.

414 Properties:

415 [Code] Receiver

416 [Subcode] wsmc:MissingSelection

417 [Reason] The MakeConnection element did not contain any selection criteria.

418 [Detail]

Generated by	Condition	Action Upon Generation	Action Upon Receipt
--------------	-----------	------------------------	---------------------

Generated by	Condition	Action Upon Generation	Action Upon Receipt
MakeConnection receiver	In response to a MakeConnection message that does not contain any selection criteria	Unspecified.	Unspecified.

---

## 419 5 Security Considerations

420 It is strongly RECOMMENDED that the communication between Web services be secured using the  
421 mechanisms described in WS-Security. In order to properly secure messages, the body and all relevant  
422 headers need to be included in the signature. Specifically, any standard messaging headers, such as  
423 those from WS-Addressing, need to be signed with the body in order to "bind" the two together.

424 Different security mechanisms may be desired depending on the frequency of messages. For example, for  
425 infrequent messages, public key technologies may be adequate for integrity and confidentiality. However,  
426 for high-frequency events, it may be more performant to establish a security context for the events using  
427 the mechanisms described in WS-Trust [[Trust](#)] and WS-SecureConversation [[SecureConversation](#)]. It  
428 should be noted that if a shared secret is used it is RECOMMENDED that derived keys be used to  
429 strengthen the secret as described in WS-SecureConversation.

430 Requests for messages which are not available to anonymous parties are strongly RECOMMENDED to  
431 require usage of WS-Security so that the requestor can be authenticated and authorized to access the  
432 indicated messages. Similarly, integrity and confidentiality SHOULD be used whenever messages have  
433 restricted access.

434 Recipients of messages are RECOMMENDED to validate the signature to authenticate and verify the  
435 integrity of the data. Specifically, recipients SHOULD verify that the sender has the right to "speak" for the  
436 message.

437 The following list summarizes common classes of attacks that apply to this protocol and identifies the  
438 mechanism to prevent/mitigate the attacks:

- 439 • Message alteration - Alteration is prevented by including signatures of the message information  
440 using WS-Security.
- 441 • Message disclosure - Confidentiality is preserved by encrypting sensitive data using WS-Security.
- 442 • Key integrity - Key integrity is maintained by using the strongest algorithms possible (by comparing  
443 secured policies - see WS-Policy and WS-SecurityPolicy [[SecurityPolicy](#)]).
- 444 • Authentication - Authentication is established using the mechanisms described in WS-Security and  
445 WS-Trust. Each message is authenticated using the mechanisms described in WS-Security.
- 446 • Accountability - Accountability is a function of the type of and strength of the key and algorithms  
447 being used. In many cases, a strong symmetric key provides sufficient accountability. However, in  
448 some environments, strong PKI signatures are required.
- 449 • Availability - All reliable messaging services are subject to a variety of availability attacks. Replay  
450 detection is a common attack and it is RECOMMENDED that this be addressed by the  
451 mechanisms described in WS-Security. Other attacks, such as network-level denial of service  
452 attacks are harder to avoid and are outside the scope of this specification. That said, care should  
453 be taken to ensure that minimal state is saved prior to any authenticating sequences.
- 454 • Replay - Messages may be replayed for a variety of reasons. To detect and eliminate this attack,  
455 mechanisms should be used to identify replayed messages such as the timestamp/nonce outlined  
456 in WS-Security. Alternatively, and optionally, other technologies, such as sequencing, can also be  
457 used to prevent replay of application messages.

458 Service endpoints SHOULD scope its searching of messages to those that were processed under the  
459 same security context as the requesting `MakeConnection` message.

---

## 460 Appendix A. Schema

461 The normative schema that is defined for WS-MakeConnection using [XML-Schema Part1] and [XML-  
462 Schema Part2] is located at:

463 <http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-schema-200702.xsd>

464 The following copy is provided for reference.

```
465 <?xml version="1.0" encoding="UTF-8"?>
466 <!-- Copyright (C) OASIS (R) 1993-2007. All Rights Reserved.
467 OASIS trademark, IPR and other policies apply. -->
468 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
469 xmlns:wsmc="http://docs.oasis-open.org/ws-rx/wsmc/200702"
470 targetNamespace="http://docs.oasis-open.org/ws-rx/wsmc/200702"
471 elementFormDefault="qualified" attributeFormDefault="unqualified">
472 <!-- Protocol Elements -->
473 <xs:complexType name="MessagePendingType">
474 <xs:sequence>
475 <xs:any namespace="##other" processContents="lax" minOccurs="0"
476 maxOccurs="unbounded"/>
477 </xs:sequence>
478 <xs:attribute name="pending" type="xs:boolean"/>
479 <xs:anyAttribute namespace="##other" processContents="lax"/>
480 </xs:complexType>
481 <xs:element name="MessagePending" type="wsmc:MessagePendingType"/>
482 <xs:element name="Address">
483 <xs:complexType>
484 <xs:simpleContent>
485 <xs:extension base="xs:anyURI">
486 <xs:anyAttribute namespace="##other" processContents="lax"/>
487 </xs:extension>
488 </xs:simpleContent>
489 </xs:complexType>
490 </xs:element>
491 <xs:complexType name="MakeConnectionType">
492 <xs:sequence>
493 <xs:element ref="wsmc:Address" minOccurs="0" maxOccurs="1"/>
494 <xs:any namespace="##other" processContents="lax" minOccurs="0"
495 maxOccurs="unbounded"/>
496 </xs:sequence>
497 <xs:anyAttribute namespace="##other" processContents="lax"/>
498 </xs:complexType>
499 <xs:element name="MakeConnection" type="wsmc:MakeConnectionType"/>
500 <xs:complexType name="MCSupportedType">
501 <xs:sequence>
502 <xs:any namespace="##other" processContents="lax" minOccurs="0"
503 maxOccurs="unbounded"/>
504 </xs:sequence>
505 <xs:anyAttribute namespace="##other" processContents="lax"/>
506 </xs:complexType>
507 <xs:element name="MCSupported" type="wsmc:MCSupportedType"/>
508 <xs:element name="UnsupportedSelection">
509 <xs:simpleType>
510 <xs:restriction base="xs:QName"/>
511 </xs:simpleType>
512 </xs:element>
513 </xs:schema>
```

---

## 514 Appendix B. WSDL

515 This WSDL describes the WS-MC protocol from the point of view of the endpoint that receives the  
516 MakeConnection message.

517 Also note that this WSDL is intended to describe the internal structure of the WS-MC protocol, and will not  
518 generally appear in a description of a WS-MC-capable Web service. See section 3.4 Policy for a higher-  
519 level mechanism to indicate that WS-MC is supported.

520 The normative WSDL 1.1 definition for WS-MakeConnection is located at:

521 <http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.0-wsdl-200702e1.wsdl>

522 The following non-normative copy is provided for reference.

```
523 <?xml version="1.0" encoding="utf-8"?>
524 <!-- Copyright (C) OASIS (R) 1993-2007. All Rights Reserved.
525 OASIS trademark, IPR and other policies apply. -->
526 <wSDL:definitions xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
527 xmlns:xs="http://www.w3.org/2001/XMLSchema"
528 xmlns:wsa="http://www.w3.org/2005/08/addressing"
529 xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
530 xmlns:wsmc="http://docs.oasis-open.org/ws-rx/wsmc/200702"
531 xmlns:tns="http://docs.oasis-open.org/ws-rx/wsmc/200702/wsdl"
532 targetNamespace="http://docs.oasis-open.org/ws-rx/wsmc/200702/wsdl">
533
534   <wSDL:types>
535     <xs:schema
536       <xs:import namespace="http://docs.oasis-open.org/ws-rx/wsmc/200702"
537       schemaLocation="http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.0-schema-
538       200702.xsd"/>
539     </xs:schema>
540   </wSDL:types>
541
542   <wSDL:message name="MakeConnection">
543     <wSDL:part name="makeConnection" element="wsmc:MakeConnection"/>
544   </wSDL:message>
545
546   <wSDL:portType name="MCAbstractPortType">
547     <wSDL:operation name="MakeConnection">
548       <wSDL:input message="tns:MakeConnection" wsam:Action="http://docs.oasis-
549 open.org/ws-rx/wsmc/200702/MakeConnection"/>
550       <!-- As described in the WS-MakeConnection specification, the
551 MakeConnection operation establishes a connection. If a matching
552 message is available then the back-channel of the connection will
553 be used to carry the message. In SOAP terms the returned message
554 is not a response, so there is no WSDL output message. -->
555     </wSDL:operation>
556   </wSDL:portType>
557
558 </wSDL:definitions>
```

---

## 559 Appendix C. Message Examples

### 560 Appendix C.1 Example use of MakeConnection

561 To illustrate how a `MakeConnection` message exchange can be used to deliver messages to an  
562 Endpoint that is not addressable, consider the case of a pub/sub scenario in which the Endpoint to which  
563 notifications are to be delivered (the "event consumer") is not addressable by the notification sending  
564 Endpoint (the "event producer"). In this scenario the event consumer must initiate the connections in order  
565 for the notifications to be delivered. One possible set of message exchanges (using HTTP) that  
566 demonstrate how this can be achieved using `MakeConnection` is shown below.

567 **Step 1** – During a "subscribe" operation, the event consumer's EPR specifies the MC anonymous URI and  
568 the WS-RM Policy Assertion [[WS-RM Policy](#)] to indicate whether or not RM is required:

```
569 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"  
570 xmlns:wsmc="http://docs.oasis-open.org/ws-rx/wsmc/200702"  
571 xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200702"  
572 xmlns:wsa="http://www.w3.org/2005/08/addressing">  
573   <S:Header>  
574     <wsa:To> http://example.org/subscriptionService </wsa:To>  
575     <wsa:MessageID> http://client456.org/id-a6d8-a7c2eb546813</wsa:MessageID>  
576     <wsa:ReplyTo>  
577       <wsa:To> http://client456.org/response </wsa:To>  
578     </wsa:ReplyTo>  
579   </S:Header>  
580   <S:Body>  
581     <sub:Subscribe xmlns:sub="http://example.org/subscriptionService">  
582       <!-- subscription service specific data -->  
583       <targetEPR>  
584         <wsa:Address>http://docs.oasis-open.org/ws-  
585 rx/wsmc/200702/anonymous?id=550e8400-e29b-11d4-a716-446655440000</wsa:Address>  
586         <wsa:Metadata>  
587           <wsp:Policy wsu:Id="MyPolicy">  
588             <wsrmp:RMAssertion/>  
589           </wsp:Policy>  
590         </wsa:Metadata>  
591       </targetEPR>  
592     </sub:Subscribe>  
593   </S:Body>  
594 </S:Envelope>
```

595 In this example the `subscribe` and `targetEPR` elements are simply examples of what a subscription  
596 request message might contain. Note: the `wsa:Address` element contains the MC anonymous URI  
597 indicating that the notification producer needs to queue the messages until they are requested using the  
598 `MakeConnection` message exchange. The EPR also contains the WS-RM Policy Assertion indicating the  
599 RM must be used when notifications related to this subscription are sent.

600

601 **Step 2** – Once the subscription is established, the event consumer checks for a pending message:

```
602 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"  
603 xmlns:wsmc="http://docs.oasis-open.org/ws-rx/wsmc/200702"  
604 xmlns:wsa="http://www.w3.org/2005/08/addressing">  
605   <S:Header>  
606     <wsa:Action>http://docs.oasis-open.org/ws-  
607 rx/wsmc/200702/MakeConnection</wsa:Action>  
608     <wsa:To> http://example.org/subscriptionService </wsa:To>
```

```

609     </S:Header>
610     <S:Body>
611         <wsmc:MakeConnection>
612             <wsmc:Address>http://docs.oasis-open.org/ws-
613 rx/wsmc/200702/anonymous?id=550e8400-e29b-11d4-a716-446655440000</wsmc:Address>
614         </wsmc:MakeConnection>
615     </S:Body>
616 </S:Envelope>

```

617 **Step 3** – If there are messages waiting to be delivered then a message will be returned back to the event  
618 consumer. However, because WS-RM is being used to deliver the messages, the first message returned  
619 is a CreateSequence:

```

620 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
621 xmlns:wsmc="http://docs.oasis-open.org/ws-rx/wsmc/200702"
622 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200702"
623 xmlns:wsa="http://www.w3.org/2005/08/addressing">
624     <S:Header>
625         <wsa:Action>http://docs.oasis-open.org/ws-
626 rx/wsmr/200702/CreateSequence</wsa:Action>
627         <wsa:To>http://docs.oasis-open.org/ws-rx/wsmc/200702/anonymous?id=550e8400-
628 e29b-11d4-a716-446655440000</wsa:To>
629         <wsa:ReplyTo> http://example.org/subscriptionService </wsa:ReplyTo>
630         <wsa:MessageID> http://example.org/id-123-456 </wsa:MessageID>
631         <wsmc:MessagePending pending="true"/>
632     </S:Header>
633     <S:Body>
634         <wsmr:CreateSequence>
635             <wsmr:AcksTo>
636                 <wsa:Address> http://example.org/subscriptionService </wsa:Address>
637             </wsmr:AcksTo>
638         </wsmr:CreateSequence>
639     </S:Body>
640 </S:Envelope>

```

641 Notice from the perspective of how the RM Source on the event producer interacts with the RM Destination  
642 of those messages, nothing new is introduced by the use of the MakeConnection, the use of RM  
643 protocol is the same as the case where the event consumer is addressable. Note the message contains a  
644 wsmc:MessagePending header indicating that additional message are waiting to be delivered.

645

646 **Step 4** – The event consumer will respond with a CreateSequenceResponse message per normal WS-  
647 Addressing rules:

```

648 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
649 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200702"
650 xmlns:wsa="http://www.w3.org/2005/08/addressing">
651     <S:Header>
652         <wsa:Action>http://docs.oasis-open.org/ws-
653 rx/wsmr/200702/CreateSequenceResponse</wsa:Action>
654         <wsa:To> http://example.org/subscriptionService </wsa:To>
655         <wsa:RelatesTo> http://example.org/id-123-456 </wsa:RelatesTo>
656     </S:Header>
657     <S:Body>
658         <wsmr:CreateSequenceResponse>
659             <wsmr:Identifier> http://example.org/rmid-456 </wsmr:Identifier>
660         </wsmr:CreateSequenceResponse>
661     </S:Body>
662 </S:Envelope>

```



663 Note, this message is carried on an HTTP request directed to the `wsa:ReplyTo` EPR, and the HTTP  
664 response will be an HTTP 202.

665

666 **Step 5** – The event consumer checks for another message pending:

```
667 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"  
668 xmlns:wsmc="http://docs.oasis-open.org/ws-rx/wsmc/200702"  
669 xmlns:wsa="http://www.w3.org/2005/08/addressing">  
670   <S:Header>  
671     <wsa:Action>http://docs.oasis-open.org/ws-  
672 rx/wsmc/200702/MakeConnection</wsa:Action>  
673     <wsa:To> http://example.org/subscriptionService </wsa:To>  
674   </S:Header>  
675   <S:Body>  
676     <wsmc:MakeConnection>  
677       <wsmc:Address>http://docs.oasis-open.org/ws-  
678 rx/wsmc/200702/anonymous?id=550e8400-e29b-11d4-a716-446655440000</wsmc:Address>  
679     </wsmc:MakeConnection>  
680   </S:Body>  
681 </S:Envelope>
```

682 Notice this is the same message as the one sent in step 2.

683

684 **Step 6** – Since there is a message pending for this destination then it is returned on the HTTP response:

```
685 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"  
686 xmlns:wsmc="http://docs.oasis-open.org/ws-rx/wsmc/200702"  
687 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200702"  
688 xmlns:wsa="http://www.w3.org/2005/08/addressing">  
689   <S:Header>  
690     <wsa:Action> http://example.org/eventType1</wsa:Action>  
691     <wsa:To>http://docs.oasis-open.org/ws-rx/wsmc/200702/anonymous?id=550e8400-  
692 e29b-11d4-a716-446655440000</wsa:To>  
693     <wsmr:Sequence>  
694       <wsmr:Identifier> http://example.org/rmid-456 </wsmr:Identifier>  
695     </wsmr:Sequence>  
696     <wsmc:MessagePending pending="true"/>  
697   </S:Header>  
698   <S:Body>  
699     <!-- event specific data -->  
700   </S:Body>  
701 </S:Envelope>
```

702 As noted in step 3, the use of the RM protocol does not change when using `MakeConnection`. The  
703 format of the messages, the order of the messages sent and the timing of when to send it remains the  
704 same.

705

706 **Step 7** – At some later interval, or immediately due to the `MessagePending` header's "pending"  
707 attribute being set to "true", the event consumer will poll again:

```
708 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"  
709 xmlns:wsmc="http://docs.oasis-open.org/ws-rx/wsmc/200702"  
710 xmlns:wsa="http://www.w3.org/2005/08/addressing">  
711   <S:Header>  
712     <wsa:Action> http://docs.oasis-open.org/ws-rx/wsmc/200702/MakeConnection  
713   </wsa:Action>  
714     <wsa:To> http://example.org/subscriptionService </wsa:To>  
715   </S:Header>
```

```
716 <S:Body>
717 <wsmc:MakeConnection>
718 <wsmc:Address>http://docs.oasis-open.org/ws-
719 rx/wsmc/200702/anonymous?id=550e8400-e29b-11d4-a716-446655440000</wsmc:Address>
720 </wsmc:MakeConnection>
721 </S:Body>
722 </S:Envelope>
```

723 Notice this is the same message as the one sent in steps 2 and 5. As in steps 3 and 6, the response to the  
724 `MakeConnection` can be any message destined to the specified Endpoint. This allows the event  
725 producer to send not only application messages (events) but RM protocol messages (e.g.  
726 `CloseSequence`, `TerminateSequence` or even additional `CreateSequence` messages) as needed.

727

728 **Step 8** – If at any point in time there are no messages pending, in response to a `MakeConnection` the  
729 event producer returns an HTTP 202 back to the event consumer. The process then repeats (back to step  
730 7) until the subscription ends.

---

## 731 Appendix D. Acknowledgments

732 The following individuals have provided invaluable input into the initial contribution:

733	Keith Ballinger, Microsoft	747	Efim Hudis, Microsoft
734	Stefan Batres, Microsoft	748	David Ingham, Microsoft
735	Rebecca Bergersen, Iona	749	Gopal Kakivaya, Microsoft
736	Allen Brown, Microsoft	750	Johannes Klein, Microsoft
737	Kyle Brown, IBM	751	Frank Leymann, IBM
738	Michael Conner, IBM	752	Martin Nally, IBM
739	George Copeland, Microsoft	753	Peter Niblett, IBM
740	Francisco Curbera, IBM	754	Jeffrey Schlimmer, Microsoft
741	Paul Fremantle, IBM	755	James Snell, IBM
742	Steve Graham, IBM	756	Keith Stobie, Microsoft
743	Pat Helland, Microsoft	757	Satish Thatte, Microsoft
744	Rick Hill, Microsoft	758	Stephen Todd, IBM
745	Scott Hinkelman, IBM	759	Sanjiva Weerawarana, IBM
746	Tim Holloway, IBM	760	Roger Wolter, Microsoft

761 The following individuals were members of the committee during the development of this specification:

762	Abbie Barbir, Nortel	792	Paul Knight, Nortel
763	Charlton Barreto, Adobe	793	Dan Leshchiner, Tibco
764	Stefan Batres, Microsoft	794	Mark Little, JBoss
765	Hamid Ben Malek, Fujitsu	795	Lily Liu, webMethods
766	Andreas Bjarlestam, Ericsson	796	Matt Lovett, IBM
767	Toufic Boubez, Layer 7	797	Ashok Malhotra, Oracle
768	Doug Bunting, Sun	798	Jonathan Marsh, Microsoft
769	Lloyd Burch, Novell	799	Daniel Millwood, IBM
770	Steve Carter, Novell	800	Jeff Mischkin, Oracle
771	Martin Chapman, Oracle	801	Nilo Mitra, Ericsson
772	Dave Chappell, Sonic	802	Peter Niblett, IBM
773	Paul Cotton, Microsoft	803	Duane Nickull, Adobe
774	Glen Daniels, Sonic	804	Eisaku Nishiyama, Hitachi
775	Doug Davis, IBM	805	Dave Orchard, BEA
776	Blake Dournaee, Intel	806	Chouthri Palanisamy, NEC
777	Jacques Durand, Fujitsu	807	Sanjay Patil, SAP
778	Colleen Evans, Microsoft	808	Gilbert Pilz, BEA
779	Christopher Ferris, IBM	809	Martin Raeppe, SAP
780	Paul Fremantle, WSO2	810	Eric Rajkovic, Oracle
781	Robert Freund, Hitachi	811	Stefan Rossmann, SAP
782	Peter Furniss, Erebor	812	Tom Rutt, Fujitsu
783	Marc Goodner, Microsoft	813	Rich Salz, IBM
784	Alastair Green, Choreology	814	Shivajee Samdarshi, Tibco
785	Mike Grogan, Sun	815	Vladimir Videlov, SAP
786	Ondrej Hrebicek, Microsoft	816	Claus von Riegen, SAP
787	Kazunori Iwasa, Fujitsu	817	Pete Wenzel, Sun
788	Chamikara Jayalath, WSO2	818	Steve Winkler, SAP
789	Lei Jin, BEA	819	Umit Yalçinalp, SAP
790	Ian Jones, BTplc	820	Nobuyuki Yamamoto, Hitachi
791	Anish Karmarkar, Oracle		

821