3.3 Composition with WS-Addressing

(Add new bullet 4, current 4 becomes 5.)

4. When an Endpoint generates a `MakeConnection` message that has no element content in the SOAP body, then the value of the `wsa:Action` IRI MUST be:

```
http://docs.oasis-open.org/ws-rx/wsrm/200608/MakeConnection
```

3.4 Sequence Creation

(Update description of AcksTo EPR to include being the destination of MakeConnection messages.)

/wsrm:CreateSequence/wsrm:AcksTo

The RM Source MUST include this element in any CreateSequence message it sends. This element is of type wsa:EndpointReferenceType (as specified by WS-Addressing). It specifies the endpoint reference to which messages containing SequenceAcknowledgement, MakeConnection header blocks and faults related to the created Sequence are to be sent, unless otherwise noted in this specification (for example, see Section 3.2).

The RM Source MAY use the AcksTo, including reference parameters, to relate MakeConnection requests to specific sequences.

Implementations MUST NOT use an endpoint reference in the AcksTo element that would prevent the sending of Sequence Acknowledgements back to the RM Source. For example, using the WS-Addressing "http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM Destination to ever send Sequence Acknowledgements.

/wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo

The RM Destination MUST include this element, of type wsa:EndpointReferenceType (as specified by WS-Addressing). It specifies the endpoint reference to which messages containing SequenceAcknowledgement, MakeConnection header blocks and faults related to the created Sequence are to be sent, unless otherwise noted in this specification (for example, see Section 3.2).

The RM Source MAY use the AcksTo, including reference parameters, to relate MakeConnection requests to specific sequences.

Implementations MUST NOT use an endpoint reference in the AcksTo element that would prevent the sending of Sequence Acknowledgements back to the RM Source. For example, using the WS-Addressing "http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM Destination to ever send Sequence Acknowledgements.

3.10 MakeConnection

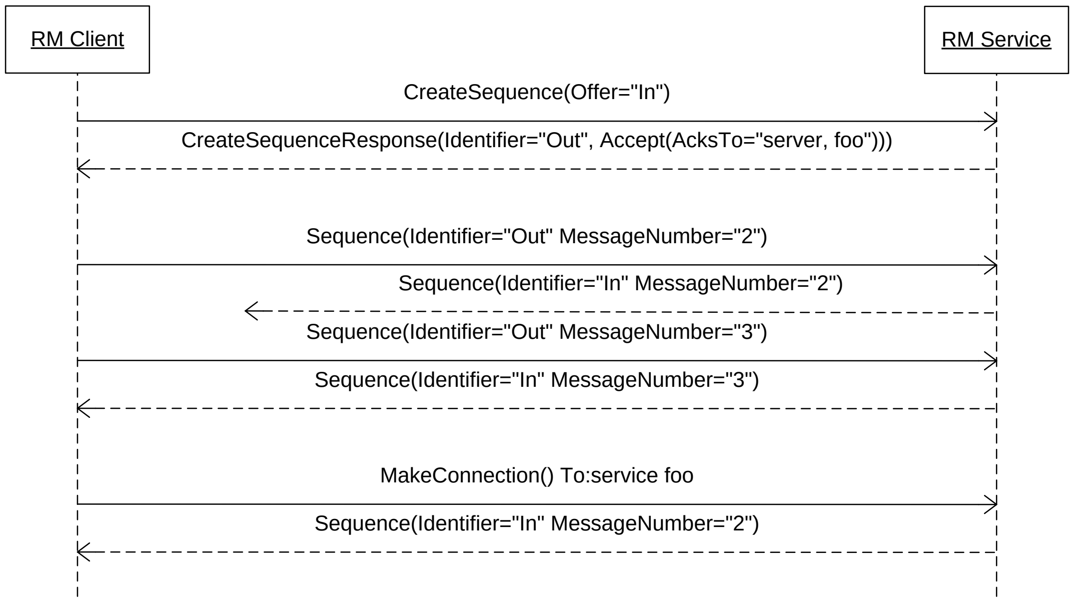
(Replace existing section as below.)

There are situations where clients using WS-RM may be unreachable but have the capability to establish protocol specific back channels. In these situations it may be desirable to establish a back channel in order to transmit Sequence Traffic or Sequence Lifecycle Messages from an established inbound sequence to the client or establish a new inbound sequence.

MakeConnection supports two common scenarios: an unreachable RM Destination wanting to signal a RM Source to use the back channel to transmit Sequence Traffic or Sequence Lifecycle Messages,and an unreachable RM Source requesting an inbound sequence.

While these scenarios may be met in a variety of ways that are out of scope of this specification, this specification provides the `MakeConnection` header as one means of addressing these scenarios. The purpose of the `MakeConnection` header block is to signal that the sender is requesting that the transport specific back channel be used to return any un-transmitted Sequence Traffic or Lifecycle Messages.

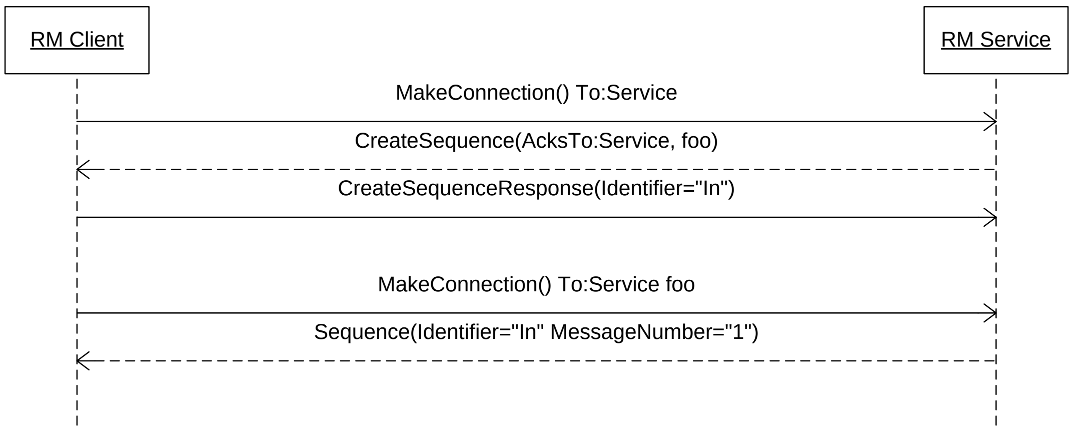
The unreachable RM Destination of a previously established inbound sequence MAY include the `MakeConnection` header in any one-way message targeted at the AcksTo EPR of the RM Source. The `MakeConnection` header may be transmitted independently of any other RM or application message contents. The RM Source MAY return un-transmitted Sequence Traffic or Sequence Lifecycle Messages for sequences it relates to the AcksTo EPR used via the protocol specific back channel of the request. See figure 1 for an example of this message flow.



RM Client
RM Service
CreateSequence(Offer="In")
CreateSequenceResponse(Identifier="Out", Accept(AcksTo="server, foo")))
Sequence(Identifier="Out" MessageNumber="2")
Sequence(Identifier="In" MessageNumber="2")
Sequence(Identifier="Out" MessageNumber="3")
Sequence(Identifier="In" MessageNumber="3")
MakeConnection() To:service foo
Sequence(Identifier="In" MessageNumber="2")

Figure 1 – MakeConnection for with established inbound sequence

When an unreachable client requires a new inbound sequence it MAY send the `MakeConnection` header independently to RM service endpoint. Upon receipt of a `MakeConnection` header block that the RM Source cannot relate to an existing sequence it MUST respond with either a `CreateSequence` message on the protocol specific back channel of the request, or with a MakeConnectionRefused fault. Subsequent MakeConnection messages MUST be transmitted to the AcksTo EPR of the RM Source. See figure 2 for an example of this message flow.



RM Client
RM Service
MakeConnection() To:Service
CreateSequence(AcksTo:Service, foo)
CreateSequenceResponse(Identifier="In")
MakeConnection() To:Service foo
Sequence(Identifier="In" MessageNumber="1")

Figure 2 - MakeConnection to establish new inbound sequence

The `MakeConnection` header MUST NOT be included on a message for which there is a defined response. There are many potential mechanisms (e.g. reference parameters in the AcksTo EPR) that allow a server to relate a new sequence to a client to one previously established from the same client. Mandating a particular mechanism an implementation must use is out of scope of this specification.

If a non mustUnderstand fault occurs when processing the `MakeConnection` header that was piggy-backed on another message, a fault MUST be generated, but the processing of the original message MUST NOT be affected.

It is RECOMMENDED that when using this mechanism to establish a new sequence and composing with WS-Security as described in section 6.1 that a new security token be used. This reduces complexity in determining the relationship of any existing tokens and their applicability to the requested new sequence and any expectations of the client requesting the sequence.

To use this mechanism to establish a new sequence and compose with SSL/TLS as described in section 6.2 the `MakeConnection` message MUST be sent over SSL/TLS. The SSL/TLS session used for this exchange MUST be used for the `CreateSequence` message on the response.

The following exemplar defines the `MakeConnection` syntax:

```
<wsrm:MakeConnection ... >
  ...
</wsrm:MakeConnection>
```

/wsrm:MakeConnection
This element allows the sender to create a transport specific back-channel that can be used to return un-transmitted Sequence Traffic or Sequence Lifecycle Messages. If the recipient is incapable of returning messages on this back channel it MUST throw a MakeConnectionRefused fault. When the recipient of this header block cannot relate the request to an established sequence it MUST return a CreateSequence response. Endpoints MUST send this element as a header block.

/wsrm:MakeConnection/{any}
This is an extensibility mechanism to allow different (extensible) types of information, based on a schema, to be passed.
/wsrm:MakeConnection/@{any}
This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the element.

3.11 MessagePending

The `MessagePending` header SHOULD be included on inbound Sequence Traffic Messages to an unreachable client as an indicator whether there are additional Sequence Traffic or Sequence Lifecycle Messages waiting to be transmitted. Upon receipt of a `MessagePending` header marked “true” the RM Destination MAY use `MakeConnection` to establish a protocol specific back channel for the RM Source to transmit Sequence Traffic or Sequence Lifecycle Messages waiting to be sent on the sequence. See figure 3 for an example of this message flow.

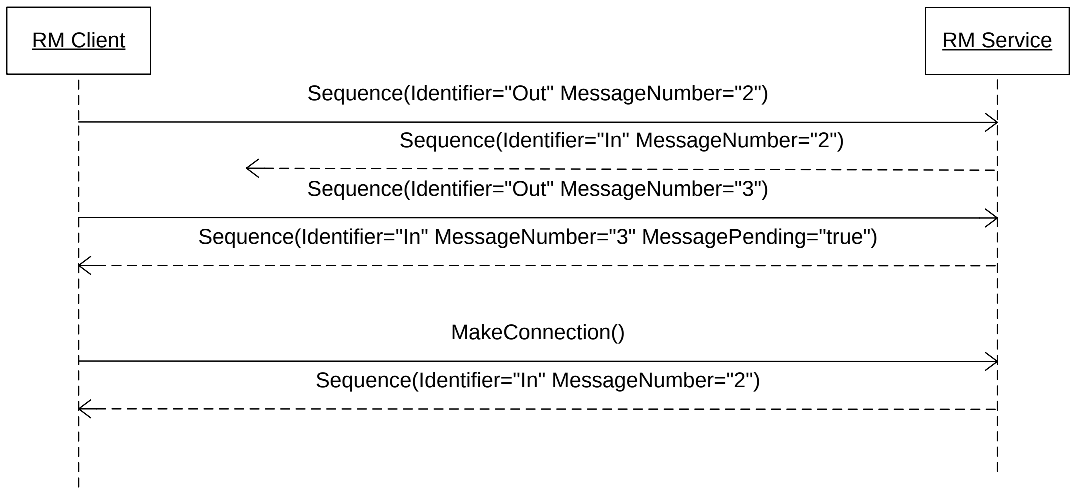

RM Client
RM Service
Sequence(Identifier="Out" MessageNumber="2")
Sequence(Identifier="In" MessageNumber="2")
Sequence(Identifier="Out" MessageNumber="3")
Sequence(Identifier="In" MessageNumber="3" MessagePending="true")
MakeConnection()
Sequence(Identifier="In" MessageNumber="2")

Figure 3 – MessagePending indicates presence of un-transmitted messages on a inbound sequence

The following exemplar defines the `MessagePending` syntax:

```
<wsrm:MessagePending pending="xs:Boolean"...>
  <wsrm:AcksTo> wsa:EndpointReferenceType </wsrm:AcksTo> ?
  ...
</wsrm:MessagePending>
```

/wsrm:MessagePending
This element indicates whether additional messages are waiting to be retrieved.

/wsrm:MessagePending@pending
This attribute, when set to "true", indicates that there is at least one message waiting to be transmitted. When this attribute is set to "false" it indicates there are currently no messages waiting to be transmitted.

/wsrm:MessagePending/wsrm:AcksTo
In the circumstance where an RM Source wishes to initiate a sequence with an anonymous client, the RM Source MAY return a MessagePending header over an existing transport backchannel. The MessagePending header MAY contain an AcksTo EPR, of type wsa:EndpointReferenceType (as specified by WS-Addressing). In that case the receiver MUST use this AcksTo EPR to construct the resulting MakeConnection.

Implementations MUST NOT use an endpoint reference in the AcksTo element that would prevent the sending of Sequence Acknowledgements back to the RM Source. For example, using the WS-Addressing "http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM Destination to ever send Sequence Acknowledgements.

/wsrm:MessagePending/{any}
This is an extensibility mechanism to allow different (extensible) types of information, based on a schema, to be passed.

/wsrm:MessagePending/@{any}
This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the element.

4.9 MakeConnectionRefused

(Change current Unsupported Selection fault to MakeConnectionRefused section as current fault is no longer required.)

[Code] Sender

[Subcode] wsrm:MakeConnectionRefused

[Reason] The MakeConnection request has been refused by the RM Source.

[Detail]

```
xs:any
```

Generated by	Condition	Action upon generation	Action upon receipt
RM Source	In response to a MakeConnection request when the RM Source is incapable of returning messages on this back channel.	Unspecified.	Unspecified.

Appendix C.6 MakeConnection

(Replace current section C.6 with the following.)

Below are example message exchanges for MakeConnection and MessagePending headers from the diagrams in sections 3.10 and 3.11.

Example 1
MakeConnection message from figure 1.

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200608"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>
      http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546893
    </wsa:MessageID>
    <wsa:Action>
      http://docs.oasis-open.org/wsrx/wsrm/200608/MakeConnection
    </wsa:Action>
    <!- foo is represented here as a reference parameter returned as a
        header. Implementations may make their own choice in how to
        distinguish MakeConnection messages sent to the AcksTo endpoint. -->
    <wsa:To>http://example.com/serviceB/123</wsa:To>
    <rms:foo/>
    <wsrm:MakeConnection/>
</S:Header>
<S:Body/>
```

The message 2 in figure 1 is returned on the underlying protocol back channel upon receipt of this message.

Example 2
Empty MakeConnection message from figure 2.

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200608"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>
      http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546823
    </wsa:MessageID>
    <wsa:Action>
      http://docs.oasis-open.org/wsrx/wsrm/200608/MakeConnection
    </wsa:Action>
    <wsa:To>http://example.com/serviceB/123</wsa:To>
    <wsa:From>
     <wsa:Address>http://www.w3.org/2005/08/addressing/anonymous</wsa:Address>
    </wsa:From>
    <wsrm:MakeConnection/>
</S:Header>
<S:Body/>
```

CreateSequence is returned on the underlying protocol back channel upon receipt of this message as illustrated in figure 2 as this request could not be related to a specific sequence. From here subsequent MakeConnection messages appear as illustrated in Example 1.

Example 3
MessagePending header returned in response to sequence Out message number 3 in Figure 3.

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200608"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>
      http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546813
    </wsa:MessageID>
    <wsa:To>http://www.w3.org/2005/08/addressing/anonymous</wsa:To>
    <wsa:From>
      <wsa:Address>http://example.com/serviceB/123</wsa:Address>
    </wsa:From>
    <wsa:Action>http://example.com/serviceB/123/response</wsa:Action>
    <wsrm:Sequence>
      <wsrm:Identifier>http://Business456.com/RM/In</wsrm:Identifier>
      <wsrm:MessageNumber>3</wsrm:MessageNumber>
    </wsrm:Sequence>
    <wsrm:MessagePending pending="true"/>
  </S:Header>
  <S:Body>
    <!-- Some Application Data -->
  </S:Body>
</S:Envelope>
```

From here MakeConnection is transmitted as illustrated in Example 1.

Appendix D. State Tables

(Only new rows for Table 1 and 2 are covered below. Tables 3 and 4 are removed.)

Table 1 RM Source Sequence State Transition Table

Events	Sequence States					
	None	Creating	Created	Closing	Closed	Terminating

Events	Sequence States					
	None	Creating	Created	Closing	Closed	Terminating
<MakeConnection> [msg] {3.10}	Xmit Create Sequence [Creating] {3.10}	N/A	Xmit message [same] {3.10}	N/A	Xmit Terminate Sequence [Terminating] {3.10}	N/A
Process un-transmitted messages [int] {3.11}			Prepare MessagePending header [same] {3.11}	N/A	Prepare MessagePending header [same] {3.11}	N/A

Table 2 RM Destination Sequence State Transition Table

Events	Sequence States		
	None	Created	Closed
MessagePending (true) [msg] {3.11}	N/A	<Xmit MakeConnection r> [same] {3.11}	<Xmit MakeConnection > [same] {3.11}
<Xmit MakeConnection> [msg/int] {3.10}	CreateSequence (successful) [Created] {3.10}	Accept Message; <Xmit SeqAck> [same] {3.10}	Accept Message; <Xmit SeqAck> [same] {3.10}