Reliable Asynchronous Messaging Profile

Version 1.0

Working Draft

2005-08-25

Latest version:
　　http://www.ibm.com/developerworks/webservices/library/specification/ws-b2b/

Authors:
　　Chris Ferris (editor), IBM
　　Tim Fowler, Ford Motor Company

Administrative contact:
　　chrisfer@us.ibm.com

Abstract

This document defines the Reliable Asynchronous Messaging Profile 1.0 (aka RAMP) (hereafter, "Profile"), consisting of a set of Web services specifications, along with clarifications and amendments to those specifications that promote interoperability. Section 1, "Introduction", introduces the Profile, and explains its relationships to other profiles. Section 2, "Profile Conformance," explains what it means to be conformant to the Profile. Each subsequent section addresses a component of the Profile, and consists of two parts; an overview detailing the component specifications and their extensibility points, followed by subsections that address individual parts of the component specifications. Note that there is no relationship between the section numbers in this document and those in the referenced specifications.

Status of this Document

This document is a Working Draft; it reflects the current state of development. It is a work in progress, and should not be considered authoritative or final; other documents may supersede this document.

Notice

terms and conditions, to their respective patent claims that they deem necessary to implement the Reliable Asynchronous Messaging Profile.

Feedback

If you would like to contribute technical comments on this document, please do so through our Feedback page at http://www.alphaworks.ibm.com/wsspec/agreement/basic_B2B.

Table of Contents

1. Introduction

This document defines the Reliable Asynchronous Messaging Profile 1.0 (RAMP) (hereafter, "Profile"), consisting of a set of Web services specifications, along with clarifications and amendments to those specifications which promote interoperability.

1.1 Problem Statement

Inadequate infrastructure to support supply chain integration between business partners is a major cost driver for industry. A 2004 NIST (National Institute of Standards and Technology) Planning Report [http://www.nist.gov/director/prog-ofc/report04-2.pdf] estimates the annual cost of inadequate supply chain infrastructure for just the U.S. automotive and electronics industry at about $9 billion or 1.2 percent of the total value of shipments. At least half of these annual costs are specifically associated with supplier interactions.

Cost, complexity and lack of well adopted standards for B2B exchange have hampered meaningful integration. This profile provides technical guidance, when used in conjunction with architectural best practices and related usage patterns, to support standards-based, cost effective B2B exchange of messages, reliably and securely, between business partners via Web services.

1.2 Relationship to WS-I Profiles

The profiles which have been developed by the Web Services - Interoperability [http://www.ws-i.org] (WS-I) organization have proven to be invaluable at increasing interoperability in the technical areas that they address. Every profile has a purpose or a set of problems that it is designed to address. The WS-I Basic Profile [BP1.1] [http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html] is designed to address basic message exchanges between a service provider and its consumers. Likewise, the WS-I Basic Security Profile [BSP1.0] [http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0-2004-05-12.html] and its related token profiles are designed to address basic security of messages exchanged between a service provider and a consumer at both the message and transport level.

However, even the combination of both the WS-I Basic Profile and the WS-I Basic Security Profile, is insufficient for many real-world business-to-business integration tasks. This Profile is designed to address the problems associated with real-world business-to-business integration by integrating the WS-I profiles, and by adding support for the most significant business-to-business integration requirement which is not addressed by any other profile: reliable asynchronous messaging.

This Profile is not intended to compete with, but rather to complement the work being done by WS-I. WS-I develops profiles of Web services specifications which are widely implemented and deployed, and for which there is broad industry consensus and adoption. This Profile may include certain Web services specifications which may not yet have achieved the necessary levels of industry consensus to be incorporated in a WS-I profile. Our intent is to accelerate industry adoption of those Web services specifications which are required for solving certain classes of business problems as described above in "Problem Statement" [#ProblemStatement] . It

is the stated intent of the authors of this Profile, that this work be contributed to WS-I once the referenced specifications have reached the necessary levels of industry consensus and adoption.
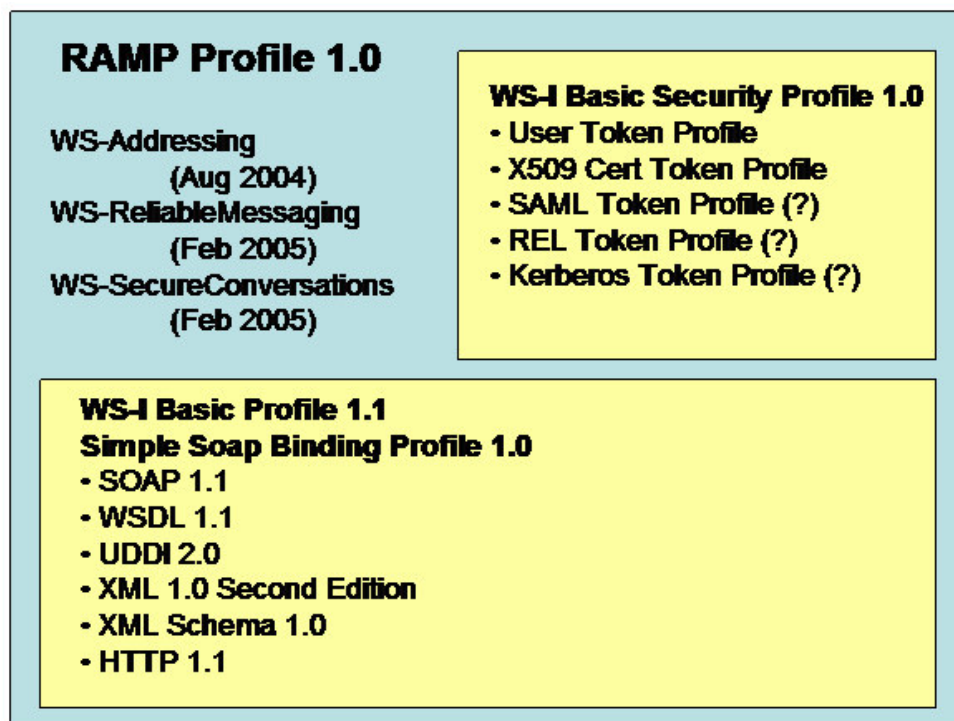
Figure 1. Relationship of Reliable Asynchronous Messaging Profile to WS-I profiles

As depicted in Figure 1 above, the Reliable Asynchronous Messaging Profile is an extension profile of a composition of the WS-I Basic Profile 1.1, Simple Soap Binding Profile 1.0, and Basic Security Profile 1.0. This means it is consistent with each of the referenced profiles but adds additional functionality required for real-world business-to-business integration tasks such as reliable, asynchronous messaging.

As an extension of the WS-I Basic Profile, this Profile is designed to support the interoperable addition of functionality to SOAP messaging, such as is required for real-world business-to-business integration tasks. One example of such functionality is the reliable delivery of SOAP messages via inherently un-reliable network protocols (e.g., HTTP), such as is described in WS-ReliableMessaging [http://schemas.xmlsoap.org/ws/2005/02/rm/], which requires additional SOAP headers be added to the SOAP message. Such a technique does not change the content of the SOAP message body.

Clarifications to Underlying Profiles

With the introduction and adoption of some newer WS-* specifications, new features and options have been made available that unfortunately are not compatible with the current WS-I Basic Profile. Since these new features are becoming required behaviors of SOAP implementations, it becomes necessary to modify some of the WS-I Basic Profile requirements in such a way that the spirit of the requirement is still intact even if the exact implementation details are slightly modified. As the requirements of this Profile are listed, the WS-I Basic Profile requirements that are impacted will be noted.

Constraining Underlying Extensibility Points

This Profile exploits the following extensibility points from underlying profiles:

- from WS-I Basic Profile 1.1 [http://www.ws-i.org/Profiles/BasicProfile-1.1.html], E0001 - Header blocks [http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html#E0001]

  (exploited by WS-Addressing and WS-ReliableMessaging). Use of bp11:E0001 is further constrained by this Profile.

- from BSP 1.0, E0002 - Security Tokens

1.3 Guiding Principles

This Profile was developed according to the same set of principles used in developing the WS-I Basic Profile [http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html#philosophy] .

1.4 Notational Conventions

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 RFC2119 [http://www.ietf.org/rfc/rfc2119.txt] .

Normative statements of requirements in the Profile (i.e., those impacting conformance, as outlined in "Conformance Requirements") are presented in the following manner:

> Rnnnn *Statement text here.*

where "nnnn" is replaced by a number that is unique among the requirements in the Profile, thereby forming a unique requirement identifier.

Requirement identifiers can be considered to be namespace qualified, in such a way as to be compatible with QNames from Namespaces in XML [http://www.w3.org/TR/REC-xml-names/] . If there is no explicit namespace prefix on a requirement's identifier (e.g., "R9999" as opposed to "bp10:R9999"), it should be interpreted as being in the namespace identified by the conformance URI of the document section it occurs in. If it is qualified, the prefix should be interpreted according to the namespace mappings in effect, as documented below.

Some requirements clarify the referenced specification(s), but do not place additional constraints upon implementations. For convenience, clarifications are annotated in the following manner: C

Some requirements are derived from ongoing standardization work on the referenced specification(s). For convenience, such forward-derived statements are annotated in the following manner: xxxx, where "xxxx" is an identifier for the specification (e.g., "WSDL20" for WSDL Version 2.0). Note that because such work was not complete when this document was publiished, the specification that the requirement is derived from may change; this information is included only as a convenience to implementers.

This specification uses a number of namespace prefixes throughout; their associated URIs are listed below. Note that the choice of any namespace prefix is arbitrary and not semantically significant.

- **soap** - "http://schemas.xmlsoap.org/soap/envelope"
- **wsdl** - "http://schemas.xmlsoap.org/wsdl/"
- **soapbind** - "http://schemas.xmlsoap.org/wsdl/soap/"
- **uddi** - "urn:uddi-org:api_v2"
- **wsa** - "http://schemas.xmlsoap.org/ws/2004/08/addressing/"
- **wsrm** - "http://schemas.xmlsoap.org/ws/2005/02/rm/"
- **wssc** - "http://schemas.xmlsoap.org/ws/2005/02/sc/"

1.5 Profile Identification and Versioning

This document is identified by a name (in this case, Reliable Asynchronous Messaging Profile) and a version number (here, 1.0). Together, they identify a particular *profile instance*.

Version numbers are composed of a major and minor portion, in the form "major.minor". They can be used to determine the precedence of a profile instance; a higher version number (considering both the major and minor components) indicates that an instance is more recent, and therefore supersedes earlier instances.

Instances of profiles with the same name (e.g., "Example Profile 1.1" and "Example Profile 5.0") address interoperability problems in the same general scope (although some developments may require the exact scope of a profile to change between instances).

One can also use this information to determine whether two instances of a profile are backwards-compatible; that is, whether one can assume that conformance to an earlier profile instance implies conformance to a later one. Profile instances with the same name and major version number (e.g., "Example Profile 1.0" and "Example Profile 1.1") MAY be considered compatible. Note that this does not imply anything about compatibility in the other direction; that is, one cannot assume that conformance with a later profile instance implies conformance

to an earlier one.

2 Profile Conformance

Conformance to the Profile is defined by adherence to the set of *requirements* defined for a specific *target*, within the *scope* of the Profile. This section explains these terms and describes how conformance is defined and used.

2.1 Conformance Requirements

Requirements state the criteria for conformance to the Profile. They typically refer to an existing specification and embody refinements, amplifications, interpretations and clarifications to it in order to improve interoperability. All requirements in the Profile are considered normative, and those in the specifications it references that are in-scope (see "Conformance Scope") should likewise be considered normative. When requirements in the Profile and its referenced specifications contradict each other, the Profile's requirements take precedence for purposes of Profile conformance.

Requirement levels, using RFC2119 [http://www.ietf.org/rfc/rfc2119.txt] language (e.g., MUST, MAY, SHOULD) indicate the nature of the requirement and its impact on conformance. Each requirement is individually identified (e.g., R9999) for convenience.

For example;

> R9999   *WIDGETs SHOULD be round in shape.*

This requirement is identified by "R9999", applies to the target WIDGET (see below), and places a conditional requirement upon widgets; i.e., although this requirement must be met to maintain conformance in most cases, there are some situations where there may be valid reasons for it not being met (which are explained in the requirement itself, or in its accompanying text).

Each requirement statement contains exactly one requirement level keyword (e.g., "MUST") and one conformance target keyword (e.g., "MESSAGE"). Additional text may be included to illuminate a requirement or group of requirements (e.g., rationale and examples); however, prose surrounding requirement statements must not be considered in determining conformance.

Definitions of terms in the Profile are considered authoritative for the purposes of determining conformance.

None of the requirements in the Profile, regardless of their conformance level, should be interpreted as limiting the ability of an otherwise conforming implementation to apply security countermeasures in response to a real or perceived threat (e.g., a denial of service attack).

2.2 Conformance Targets

Conformance targets identify what artifacts (e.g., SOAP message, WSDL description, and UDDI registry data) or parties (e.g., SOAP processor, end user) requirements apply to.

This allows for the definition of conformance in different contexts, to assure unambiguous interpretation of the applicability of requirements, and to allow conformance testing of artifacts (e.g., SOAP messages and WSDL descriptions) and the behavior of various parties to a Web service (e.g., clients and service instances).

Requirements' conformance targets are physical artifacts wherever possible, to simplify testing and avoid ambiguity.

The following conformance targets are used in the Profile:

- **MESSAGE** - protocol elements that transport the ENVELOPE (e.g., SOAP/HTTP messages) (from WS-I Basic Profile 1.1 [http://www.ws-i.org/Profiles/BasicProfile-1.1.html])
- **ENVELOPE** - the serialization of the soap:Envelope element and its content (from WS-I Basic Profile 1.1 [http://www.ws-i.org/Profiles/BasicProfile-1.1.html])
- **DESCRIPTION** - descriptions of types, messages, interfaces and their concrete protocol and data format bindings, and the network access points associated with Web services (e.g., WSDL descriptions) (from WS-I Basic Profile 1.0 [http://www.ws-i.org/Profiles/BasicProfile-1.0.html])
- **INSTANCE** - software that implements a wsdl:port or a uddi:bindingTemplate (from WS-I Basic Profile 1.0

[http://www.ws-i.org/Profiles/BasicProfile-1.0.html])

- **CONSUMER** - software that invokes an INSTANCE (from WS-I Basic Profile 1.0 [http://www.ws-i.org/Profiles/BasicProfile-1.0.html])
- **SENDER** - software that generates a message according to the protocol(s) associated with it (from WS-I Basic Profile 1.0 [http://www.ws-i.org/Profiles/BasicProfile-1.0.html])
- **RECEIVER** - software that consumes a message according to the protocol(s) associated with it (e.g., SOAP processors) (from WS-I Basic Profile 1.0 [http://www.ws-i.org/Profiles/BasicProfile-1.0.html])

2.3 Conformance Scope

The scope of the Profile delineates the technologies that it addresses; in other words, the Profile only attempts to improve interoperability within its own scope. Generally, the Profile's scope is bounded by the specifications referenced by it.

The Profile's scope is further refined by extensibility points. Referenced specifications often provide extension mechanisms and unspecified or open-ended configuration parameters; when identified in the Profile as an extensibility point, such a mechanism or parameter is outside the scope of the Profile, and its use or non-use is not relevant to conformance.

Note that the Profile may still place requirements on the use of an extensibility point. Also, specific uses of extensibility points may be further restricted by other profiles, to improve interoperability when used in conjunction with the Profile.

Because the use of extensibility points may impair interoperability, their use should be negotiated or documented in some fashion by the parties to a Web service; for example, this could take the form of an out-of-band agreement.

The Profile's scope is defined by the referenced specifications in Appendix A, as refined by the extensibility points in Appendix B.

2.4 Claiming Conformance

Claims of conformance to the Profile can be made using the following mechanisms, as described in Conformance Claim Attachment Mechanisms [http://www.ws-i.org/Profiles/ConformanceClaims-1.0.html] , when the applicable Profile requirements associated with the listed targets have been met:

- **WSDL 1.1 Claim Attachment Mechanism for Web Services Instances** - MESSAGE DESCRIPTION INSTANCE RECEIVER
- **WSDL 1.1 Claim Attachment Mechanism for Description Constructs** - DESCRIPTION
- **UDDI Claim Attachment Mechanism for Web Services Instances** - MESSAGE DESCRIPTION INSTANCE RECEIVER

The conformance claim URI for this Profile is "urn:uuid:B61CC2B0-A4A7-4E95-9A13-8BA10C134A05".

3. Sending and Receiving Messages

This section of the Profile incorporates the following specifications by reference, and defines extensibility points within them:

- WS-I Basic Profile (BP) 1.1 [http://www.ws-i.org/Profiles/BasicProfile-1.1.html]
- WS-I Simple SOAP Binding Profile (SSBP) 1.0 [http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html]

3.1 Constraining the Basic Profile

This profile further constrains certain of the options permitted by the WS-I Basic Profile to improve the prospect for interoperability.

3.1.1 Use of Document-Literal WSDL

The WS-I Basic Profile defines the terms [http://ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html#WSDLMSGS] "document-literal" and "rpc-literal" as applies to the permitted WSDL authoring styles. This profile limits the choice of WSDL authoring style permitted by requirement R2705

[http://ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html#R2705] to "document-literal".

> R3001 *A `wsdl:port` in a DESCRIPTION MUST use a "document-literal binding".*

While the profile requires use of "document-literal" style WSDL, it should be noted that the convention known as "document-literal wrapped" may be used to capture RPC semantics.

The characteristics of the document/literal wrapped pattern are as follows:

- The input and output `wsdl:message`s each have a single `part`
- The `wsdl:part` for the input and output messages each have an `@element` attribute
- The referenced `xsd:element` declaration of the input message's `wsdl:part/@element` has a `name` attribute with the same value as the `wsdl:operation` for which the `wsdl:message` will be used
- The referenced `xsd:element` declaration of the output message's `wsdl:part/@element` has a `name` attribute with the same value as the `wsdl:operation` for which the `wsdl:message` will be used, appended with "Response"
- The element's complex type has no attributes
- The WSDL binding is a document-literal binding

For example,

```
CORRECT:
<types>
    <schema>
        <element name="myMethod"/>
            <complexType>
                <sequence>
                    <element name="x" type="xsd:int"/>
                </sequence>
            </complexType>
        </element>
        <element name="myMethodResponse"/>
            <complexType>
                <sequence>
                    <element name="y" type="xsd:string"/>
                </sequence>
            </complexType>
        </element>
    </schema>
</types>
<message name="myMethodRequest">
    <part name="body" element="myMethod"/>
</message>
<message name="myMethodResponse">
    <part name="body" element="myMethodResponse"/>
</message>
<message name="empty"/>

<portType name="myPortType">
    <operation name="myMethod">
        <input message="myMethodRequest"/>
        <output message="myMethodResponse"/>
    </operation>
</portType>
```

3.2 Overriding Requirements of the WS-I Basic Profile 1.1

Normally, a profile would NEVER contradict a referenced specification's requirements. However, in this case, it has proven to be a necessary evil in order to accommodate composition of the WS-I Basic Profile with the WS-ReliableMessaging and WS-Addressing specifications. The Profile authors hope that WS-I will take note and address the composition issues in a subsequent revision of the WS-I Basic Profile.

3.2.1 SOAP Envelope in HTTP Response Message

Typically, the HTTP Response Code of a one-way WSDL would be "202 Accepted" with no SOAP envelope in the entity body of the HTTP Response Message. However, certain Web services specifications such as WS-ReliableMessaging can cause the generation of SOAP messages that are not considered to be application-level "responses" yet need to be sent back to the originator of the message carried on the HTTP Request Message. For example, when using WS-ReliableMessaging, if the `wsrm:AcksTo` EPR is the anonymous URI, then the RM Destination would send back `SequenceAcknowledgement` messages to the RM Source on the HTTP Response Message. However, if the request message is a one-way message, there is no response message to flow back - meaning the RM Destination must generate a new SOAP envelope for the specific

purpose of delivering the `SequenceAcknowledgement`. This would violate the WS-I Basic Profile requirements R2714 and R2750, which state that the response to a one-way message must include an HTTP response code of "2xx" without a SOAP envelope, and that the receiver of this flow must ignore any envelope if there is one. However, since the anonymous EPR was chosen by the RM Source, one would assume that it would be expecting this behavior. Therefore, the Profile overrides the requirements R2714 and R2750 of the WS-I Basic Profile 1.1.

> R3002 *An HTTP Response MESSAGE corresponding to a WSDL one-way operation MAY contain a SOAP envelope in its entity body.*

4. Addressing Messages

In order to effect the types of message exchange patterns (MEPs) required to enable most business-to-business usage scenarios (such as asynchronous request/response using HTTP), some form of addressing is required. This profile mandates the use of the WS-Addressing submission to the W3C. At such time as the W3C WS Addressing WG produces a Proposed Recommendation (PR), consideration will be given as to adopting its use.

This section of the Profile incorporates the following specifications by reference, and defines extensibility points within them:

- WS-Addressing (W3C Member Submission 10 August 2004)
  [http://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810/]

4.1 Use of Addressing Header Blocks

4.1.1 Presence of Message Header Blocks

> R0001 *An ENVELOPE MUST contain exactly one `wsa:To` header.*

> R0002 *An ENVELOPE MUST contain exactly one `wsa:From` header.*

> R0003 *An ENVELOPE MUST contain exactly one `wsa:MessageId` header.*

> R0004 *An ENVELOPE MUST contain exactly one `wsa:Action` header.*

For example,

```
CORRECT:
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
  <soap:Header>
   <wsa:To>http://example.com/service</wsa:To>
   <wsa:From>http://example.org/client</wsa:From>
   <wsa:ReplyTo soap:mustUnderstand='1'>
     <wsa:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</wsa:Address>
   </wsa:ReplyTo>
   <wsa:MessageId>uuid:aaaabbbb-cccc-dddd-eeee-ffffffffffff</wsa:MessageId>
   <wsa:Action>http://example.com/service/debit</wsa:Action>
  </soap:Header>
  <soap:Body>
     <!-- body contents -->
  </soap:Body>
</soap:Envelope>
```

4.1.2 Relating Output Operation to Input Operation

In the context of asynchronous messaging, it is necessary that the envelope contain correlation metadata sufficient to enable the receiver of a response message to correlate that response with the original request message.

> R0010 *Any ENVELOPE MAY contain a `wsa:ReplyTo` header block.*

> R0011 *If, in a DESCRIPTION, an operation is described with a `wsdl:output`, the message corresponding to the `wsdl:input` of that same `wsdl:operation` MUST contain a `wsa:ReplyTo` header block.*

> R0012 *If, in a DESCRIPTION, an operation is described with a `wsdl:output`, the message corresponding to the `wsdl:output` of that same `wsdl:operation` MUST contain a*

wsa:RelatesTo header block containing the value of the *wsa:MessageId* of the corresponding input message.

A CONSUMER that sends a request message whose response is expected to be delivered to the endpoint specified in the *wsa:ReplyTo* SOAP header block needs to ensure that the receiver of that message understands the processing semantics of the *wsa:ReplyTo* header block.

> R0020 *If an ENVELOPE contains a *wsa:ReplyTo* header block, that header block MUST have a *soap:MustUnderstand* attribute with a value of '1'.*

4.2 Asynchronous Request/Response Considerations

In the context of the WS-I Basic Profile, the only reason the response message to an HTTP request would be empty is when the request message is a one-way message. However, the WS-Addressing specification introduces a *wsa:ReplyTo* SOAP header block that changes the nature of a request/response WSDL operation as it relates to the HTTP binding. If a request message includes a *wsa:ReplyTo* header that does not use the WS-Addressing Anonymous URI, then the response message is expected to be sent to the *wsa:ReplyTo* endpoint reference (EPR) rather than in the HTTP response message. Since the response message will not be sent in the HTTP response message, the HTTP response code on the original connection would be "202 Accepted". While this doesn't violate any WS-I Basic Profile requirements it is a change in behavior.

> R0100 *If an envelope corresponding to a *wsdl:input* contains a *wsa:ReplyTo* EPR with an address other than the WS-Addressing Anonymous URI, then the corresponding HTTP response MESSAGE MAY have an HTTP return code of "202 Accepted" and an empty entity body.*

4.3 Composition with WS-Security

The requirements in this section apply whenever WS-Security is being used in conjunction with WS-Addressing.

> R2000 *When present in an ENVELOPE, each of the following SOAP header blocks MUST be included in the signature whenever the *soap:Body* is being signed: *wsa:To, wsa:From, wsa:Action, wsa:ReplyTo, wsa:FaultsTo, wsa:MessageId, wsa:RelatesTo.*

5. Reliable Delivery of Messages

Many business-to-business usage scenarios require reliable message delivery as a quality of service characteristic such as that provided by the WS-ReliableMessaging (WS-RM) specification.

Use of the WS-RM protocol is optional. By no means do all usage scenarios require the kind of reliable delivery semantics that WS-RM provides; but where such capability is needed, WS-RM is the sanctioned mechanism.

This section of the Profile incorporates the following specifications by reference, and defines extensibility points within them:

- WS-ReliableMessaging (WS-RM) [http://schemas.xmlsoap.org/ws/2005/02/rm/]

5.1 Sequence Creation

The WS-RM specification incorrectly identifies the value of 'P0S' as representing a duration of zero seconds for the various *wsrm:Expires* elements. The correct value of an *xs:duration* representing zero seconds is 'PT0S'.

> R1001 *In an ENVELOPE, the value of the *wsrm:Expires* element MUST NOT be 'P0S'.* [C]

For example,

```
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope"
    xmlns:wsrm="http://schemas.xmlsoap.org/ws/2005/02/rm"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <soap:Header>
    <wsa:MessageID>
      http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
    </wsa:MessageID>
    <wsa:To>http://example.org/service</wsa:To>
    <wsa:From>http://example.com/client/rm</wsa:From>
    <wsa:Action>http://schemas.xmlsoap.org/ws/2005/02/rm/CreateSequence</wsa:Action>
    <wsa:ReplyTo soap:mustUnderstand='1'>
      <wsa:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</wsa:Address>
    </wsa:ReplyTo>
  </soap:Header>
  <soap:Body>
    <wsrm:CreateSequence>
      <wsrm:AcksTo>
        <wsa:Address>http://example.com/client/rm</wsa:Address>
      </wsrm:AcksTo>
      <wsrm:Expires>P0S</wsrm:Expires>
    </wsrm:CreateSequence>
  </soap:Body>
</soap:Envelope>
```

```
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope"
    xmlns:wsrm="http://schemas.xmlsoap.org/ws/2005/02/rm"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <soap:Header>
    <wsa:MessageID>
      http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
    </wsa:MessageID>
    <wsa:To>http://example.org</wsa:To>
    <wsa:From>http://example.com/client/rm</wsa:From>
    <wsa:Action>http://schemas.xmlsoap.org/ws/2005/02/rm/CreateSequence</wsa:Action>
    <wsa:ReplyTo soap:mustUnderstand='1'>
      <wsa:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</wsa:Address>
    </wsa:ReplyTo>
  </soap:Header>
  <soap:Body>
    <wsrm:CreateSequence>
      <wsrm:AcksTo>
        <wsa:Address>http://example.com/client/rm</wsa:Address>
      </wsrm:AcksTo>
      <wsrm:Expires>PT0S</wsrm:Expires>
    </wsrm:CreateSequence>
  </soap:Body>
</soap:Envelope>
```

5.1.2 Changing Expires Element Value

The specification states that the value of the `wsrm:Expires` element in a `CreateSequenceResponse` can be less than the corresponding value in the `CreateSequence`. However, since there is no (positive) value less than 'PT0S' then the destination has no choice but accept an unlimited duration `Sequence` or return a `CreateSequenceRefused` fault.

> R1002 *The value of the `wsrm:Expires` element in a `CreateSequenceResponse` ENVELOPE MUST NOT have a value other than 'PT0S' when the value of the wsrm:Expires in the corresponding `CreateSequence` request was 'PT0S'.*

5.2 Sequence Header Block

5.2.1 Use of soap:mustUnderstand attribute

When a message is sent reliably, using the WS-RM protocol, it is imperative to the success of the protocol that the recipient of that message understand the processing semantics of the `wsrm:Sequence` header block.

> R1010 *If an ENVELOPE contains a `wsrm:Sequence` header block, that header block MUST have a `soap:MustUnderstand` attribute with a value of '1'.*

For example,

```
INCORRECT:
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope"
    xmlns:wsrm="http://schemas.xmlsoap.org/ws/2005/02/rm"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <soap:Header>
    <wsa:MessageID>
      http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
    </wsa:MessageID>
    <wsa:To>http://example.org/service</wsa:To>
    <wsa:From>http://example.com/client/rm</wsa:From>
    <wsa:Action>http://schemas.xmlsoap.org/ws/2005/02/rm/CreateSequence</wsa:Action>
    <wsa:ReplyTo soap:mustUnderstand='1'>
       <wsa:Address>http://example.com/client</wsa:Address>
    </wsa:ReplyTo>
    <wsrm:Sequence>
      <wsrm:Identifier>http://example.org/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817</wsrm:Identifier>
      <wsrm:MessageNumber>42</wsrm:MessageNumber>
    </wsrm:Sequence>
  </soap:Header>
  <soap:Body>
     <!-- content of body -->
  </soap:Body>
</soap:Envelope>
```

```
CORRECT:
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope"
    xmlns:wsrm="http://schemas.xmlsoap.org/ws/2005/02/rm"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <soap:Header>
    <wsa:MessageID>
      http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
    </wsa:MessageID>
    <wsa:To>http://example.org/service</wsa:To>
    <wsa:From>http://example.com/client/rm</wsa:From>
    <wsa:Action>http://example.org/service/credit</wsa:Action>
    <wsa:ReplyTo soap:mustUnderstand='1'>
        <wsa:Address>http://example.com/client</wsa:Address>
    </wsa:ReplyTo>
    <wsrm:Sequence soap:mustUnderstand='1'>
      <wsrm:Identifier>http://example.org/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817</wsrm:Identifier>
      <wsrm:MessageNumber>42</wsrm:MessageNumber>
    </wsrm:Sequence>
  </soap:Header>
  <soap:Body>
     <!-- content of body -->
  </soap:Body>
</soap:Envelope>
```

5.3 SequenceAcknowledgement Header Block

5.3.1 Piggy-backing SequenceAcknowledgements

If a SOAP envelope happens to be going to the same endpoint that was specified by a
`wsrm:CreateSequence/wsrm:AcksTo` EPR, then the RM destination can add a `wsrm:SequenceAcknowledgement`
SOAP header (for each relevant Sequence) to the envelope instead of sending them in separate messages to
the RM source. This practice is commonly referred to as "piggy-backing" of acknowledgements.

> R1021 *A `wsrm:SequenceAcknowledgement` header, for each relevant Sequence, MAY be*
> *included in any ENVELOPE destined for the endpoint specified by the*
> *`wsrm:CreateSequence/wsrm:AcksTo` EPR.*

5.3.2 Carrying SequenceAcknowledgement on HTTP response for one-way operations

The Basic Profile requirements R2714 [http://ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html#R2714] and R2750
[http://ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html#R2750] conflict with the ability to leverage the HTTP response
message to carry `wsrm:SequenceAcknowledgement`s for a reliable one-way operation. As this requirement is likely
to be commonplace in such cases where the RM Source is hidden behind a firewall, this profile overrides the
Basic Profile requirements R2714 and R2750.

> R1022 *A MESSAGE corresponding to the HTTP response message for a one-way operation*
> *MAY contain a SOAP envelope with a `wsrm:SequenceAcknowledgement` SOAP header*
> *block and no child element of the `soap:Body` in its entity body.*

R1023 *A CONSUMER that specifies the WS-Addressing Anonymous URI as the address in a `wsrm:CreateSequence/wsrm:AcksTo` EPR MUST be prepared to receive and process HTTP response messages for a one-way operation that contain a SOAP envelope in their entity bodies.*

5.4 SequenceFault Header Block

5.4.1 Mapping [Detail] Property in SOAP1.1

The WS-RM specification omits the mapping of the `[Detail]` property for the SOAP1.1 binding. It is supposed to be mapped to the `wsrm:SequenceFault` element as a child element directly following the `wsrm:FaultCode` element.

R1060 *If an ENVELOPE contains a `wsrm:SequenceFault` header block, that header block MUST have the `[Detail]` property as a child element directly following the `wsrm:FaultCode` element.* ^C

For example,

```
CORRECT:
<soap:Envelope>
  <soap:Header>
    <wsrm:SequenceFault>
      <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
      [Detail]
      ...
    </wsrm:SequenceFault>
    <!-- Headers elided for clarity. -->
  </soap:Header>
  <soap:Body>
    <soap:Fault>
      <faultcode> [Code] </faultcode>
      <faultstring> [Reason] </faultstring>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

5.5 Composition with WS-Addressing

5.5.1 WS-Addressing

R1050 *If the WS-Addressing Anonymous URI is used in the `wsrm:AcksTo` element of a `wsrm:CreateSequence` ENVELOPE, all SequenceAcknowledgement messages for the created Sequence MUST be sent back on the response flow of the HTTP connection on which a message is received containing a `wsrm:Sequence` header block with a `wsrm:Identifier` element containing the same value as the created Sequence.*

5.5.2 Use of WS-Addressing Anonymous URI

The ability to retry the transmission of messages is a key component of WS-ReliableMessaging. However, this requires that the sender of a message must be able to establish a connection with the receiving endpoint. As such, using the anonymous URI in the `wsa:ReplyTo` EPR of the request message violates this because the sender of the response can not initiate new connections back to this EPR as needed to support WS-ReliableMessaging.

R1051 *In a message exchange in which both the input (request) and output (response) messages are to be delivered reliably using WS-RM, the input ENVELOPE's `wsa:ReplyTo` EPR MUST NOT be the WS-Addressing Anonymous URI.*

5.6 Composition with WS-Security

In order to effect end-to-end secure, reliable messaging between business partners, the typical usage scenario will compose the use of WS-RM, WS-Addressing, and WS-Security, as constrained by the WS-I Basic Security Profile 1.0. The requirements in this section apply whenever WS-Security is being used in conjunction with WS-RM.

5.6.1 Signing Header Blocks

In order to ensure the integrity of the WS-RM protocol elements (e.g. that they have not been tampered-with by a "man-in-the-middle"), they need to be digitally signed.

> R2010 *When present in an ENVELOPE, each of the following SOAP header blocks MUST be included in the signature whenever the `soap:Body` is being signed: `wsrm:Sequence`, `wsrm:SequenceAcknowledgement.`*

5.6.2 Enabling Detection of Replay Attacks

The value of the wsu:Timestamp element must be changed for retransmissions so that the security layer at the RM Destination can use that information to distinguish between a valid retransmission of an unacknowledged message and a potential replay attack mounted by a third party.

> 2011 *The wsse:Security header block in an ENVELOPE MUST contain a wsu:Timestamp child element.*

> 2012 *The value of the wsu:Timestamp element in an ENVELOPE MUST be the system time of the RM Source at time of transmission for each transmission of a message (the initial and each successive transmission of an unacknowledged message).*

5.7 Delivery Assurance

The WS-RM protocol itself defines an *AtLeastOnce* quality of service between the RM Source and RM Destination. How the respective endpoints choose to provide the other supported delivery assurances (AtMostOnce, ExactlyOnce, and InOrder) is up to the implementation of the participating endpoints.

For instance, a resource constrained RM Destination might only be capable of storing a limited number of unprocessed messages. When its store is full, it might employ an algorithm such as First In First Out (FIFO), Last In First Out (LIFO) or LargestOneOutOfThePool when it receives additional messages. Such an endpoint might then offer its Application Destinations the following choices of Quality of Service (QoS) contract: AtMostOnce, which means that it would be free to drop some messages on the floor; or AtLeastOnce, ExactlyOnce, or InOrder, in which case it would have to fault, possibly terminating the Sequence, upon receiving a message that it could not accommodate in its store of unprocessed messages.

Similarly, on the sending side, the RM Source might be resource constrained with regards to how many unacknowledged messages it was capable of storing. Once its capacity had been reached, it might have to fault back to the Application Source sending another message, indicating that it could not accept responsibility for delivering any more messages.

6. Security

Many business-to-business usage scenarios require end-ro-end message security as a quality of service characteristic such as that provided by the WS-I Basic Security Profile 1.0 (WS-I BSP) specification.

Use of the WS-I BSP is optional. By no means do all usage scenarios require the levels of security WS-I BSP provides; but where such capability is needed, WS-I BSP is the sanctioned mechanism.

> **Editors' note:** At the time of this publication, the BSP1.0 is still a Working Group Draft. As the BSP progresses towards WS-I Final Material, this profile will be updated to reflect any necessary changes.

This section of the Profile incorporates the following specifications by reference, and defines extensibility points within them:

- WS-I Basic Security Profile (BSP) 1.0 [http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html]
- Web Services Secure Conversation (WS-SC) [http://schemas.xmlsoap.org/ws/2005/02/sc/]

6.1 Constraining the Basic Security Profile

This profile further constrains certain of the options permitted by the WS-I Basic Security Profile to improve the

prospect for interoperability.

6.1.1 Use of RSA1.5

This profile overrides the Basic Security Profile 1.0 requirement R5621
[http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0-2005-01-20.html#R5621] by limiting the choice of encryption algorithm
to the widely implemented RSA-1_5 to improve the prospects for interoperability.

> R5001 *When used for Key Transport, any `xenc:EncryptionMethod/@Algorithm` attribute in an*
> *ENCRYPTED_KEY SHOULD have a value of*
> *"http://www.w3.org/2001/04/xmlenc#rsa-1_5".*

6.1.2 Applying Security Policy

Some runtime platforms do not provide for the ability to apply a different security policy to selected messages
within a given port. The profile therefore recommends that all messages within a given port have the same
security policy.

> R5002 *When applying security policy, all MESSAGEs within a given `wsdl:port` SHOULD have*
> *the same security policy characteristics.*

6.2 Constraining WS-Secure Conversation

> **Editors' note:** We need to include requirements derived from the ws-rm+sc/t interop workshop
> scenario.

6.2.1 Applying Security Policy

> R6001

Appendix A: Referenced Specifications

The following specifications' requirements are incorporated into the Profile by reference, except where
superseded by the Profile:

- WS-I Basic Profile (BP) 1.1 [http://www.ws-i.org/Profiles/BasicProfile-1.1.html]
- WS-I Simple SOAP Binding Profile (SSBP) 1.0 [http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html]
- WS-Addressing (W3C Member Submission 10 August 2004)
  [http://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810/]
- WS-ReliableMessaging (WS-RM) [http://schemas.xmlsoap.org/ws/2005/02/rm/]
- WS-I Basic Security Profile (BSP) 1.0 [http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html]
- Web Services Secure Conversation (WS-SC) [http://schemas.xmlsoap.org/ws/2005/02/sc/]

Appendix B: Extensibility Points

This section identifies extensibility points, as defined in "Scope of the Profile," for the Profile's component
specifications.

These mechanisms are out of the scope of the Profile; their use may affect interoperability, and may require
private agreement between the parties to a Web service.

Appendix C: Acknowledgements

The authors wish to extend their thanks and appreciation for the valuable comments and feedback from the
following individuals:

Richard Allen (IBM), Rob Cutlip (IBM), Doug Davis (IBM), Ed Esker (Ford Motor Company), Chris Ferris (IBM),
Tom Glover (IBM), Tony Greening (Ford Motor Company), John Jackson (General Motors), Brian LeBlanc
(DaimlerChrysler) Michael McIntosh (IBM), Barbara McKee (IBM), David Murrell (Ford Motor Company),

Anthony Nadalin (IBM), Ray Reilly (Ford Motor Company), Rimas Rekasius (IBM), Faisal Waris (Ford Motor Company).