

# What's New in SOAP 1.2

*Keywords:* SOAP, Web Services

## **Marc Hadley**

Sun Microsystems

Bagshot

United Kingdom

[marc.hadley@sun.com](mailto:marc.hadley@sun.com)

<http://www.sun.com/xml/>

## *Biography*

Marc Hadley is a Staff Engineer in the XML Technology Centre at Sun Microsystems. He currently represents Sun on the W3C XML Protocol Working Group where he is co-editor of the SOAP 1.2 specification.

Prior to joining Sun Microsystems, Marc worked for Chrystal Software as a principle consultant and chief engineer developing content management and web delivery systems based on XML.

---

## Abstract

---

SOAP is currently the subject of a standardisation effort at the W3C. This paper describes the differences between SOAP 1.1 and SOAP 1.2, SOAP 1.2 being the results of the standardisation process.

This document is based on SOAP 1.2 Last Call Working Draft, May 2002. As the SOAP specification is not yet a W3C Recommendation, further changes are possible.

---

## Table of Contents

---

### Introduction

### "Big Picture" Changes

▣▣▣▣ Acronym

▣▣▣▣ Documents

▣▣▣▣ Infoset

▣▣▣▣ KML Namespaces

### Detailed Changes

▣▣▣▣ Envelope

▣▣▣▣▣▣▣▣ Trailers

▣▣▣▣▣▣▣▣ Actors and Roles

▣▣▣▣▣▣▣▣ Fault Codes

▣▣▣▣▣▣▣▣ Modified Fault Structure

▣▣▣▣▣▣▣▣ Hierarchical Fault Codes

▣▣▣▣▣▣▣▣ Modified Fault Semantics

▣▣▣▣▣▣▣▣ Misunderstood Header

▣▣▣▣ Processing Model

▣▣▣▣ Versioning Mechanism

▣▣▣▣▣▣▣▣ Upgrade Header

▣▣▣▣ Binding Framework

▣▣▣▣ Encoding

▣▣▣▣▣▣▣▣ encodingStyle Attribute

- [Multi-reference Values](#)
- [References](#)
- [Arrays](#)
- [Application to XML Name Mapping](#)
- [Serialisation Roots](#)
- [Omitted Accessors](#)
- [RPC](#)
- [RPC Array Representation](#)
- [RPC Encoding Constraint](#)
- [RPC Fault Codes](#)
- [RPC Return Values](#)
- [HTTP Binding](#)
- [Content Type](#)
- [Miscellaneous Changes](#)
- [Bibliography](#)

This paper outlines the differences between SOAP 1.1 and SOAP 1.2. It is not intended as an in-depth description of new or changed features, rather it aims to highlight the existence of such differences. Readers are referred to the relevant specifications for more in-depth coverage.

The differences between [\[SOAP 1.1\]](#) and SOAP 1.2 (see [Documents](#)) can be broadly categorised into "big picture" and "detailed" changes. The following sections describe the changes in each of these categories.

This section describes the "big picture" changes between SOAP 1.1 and SOAP 1.2. Detailed changes are described in [Detailed Changes](#).

## Acronym

From version 1.2, SOAP is no longer an acronym. The original expansion "Simple Object Access Protocol" is somewhat misleading.

## Documents

[\[SOAP 1.1\]](#) is a single document. The SOAP 1.2 specification comes in three parts:

- [\[SOAP 1.2 - Part 0\]](#) is a non-normative introduction to SOAP.
- [\[SOAP 1.2 - Part 1\]](#) describes the structure of SOAP messages, the SOAP processing model and a framework for binding SOAP to underlying protocols. Conformant SOAP implementations must implement everything in Part 1.
- [\[SOAP 1.2 - Part 2\]](#) describes optional add-ins to the core of SOAP including a data model and encoding, an RPC convention and a binding to HTTP. Conformant SOAP implementations may implement anything in Part 2, if they do so they must conform to the relevant parts of the specification.

## Infoset

SOAP 1.1 is based on [\[XML 1.0\]](#), SOAP 1.2 is based on [\[XML Infoset\]](#). In SOAP 1.2 it is left to the specification of a binding to an underlying protocol to specify the XML serialisation used in underlying protocol data units. The HTTP binding specified in [\[SOAP 1.2 - Part 2\]](#) uses XML 1.0 as the serialisation of the SOAP message infoset.

## XML Namespaces

The XML namespaces for the envelope and encoding schemas have changed. This allows SOAP processors to easily distinguish between SOAP 1.1 and SOAP 1.2 messages and allows changes in the SOAP schema without affecting existing implementations.

This section describes detailed changes between SOAP 1.1 and SOAP 1.2. "Big picture" changes are described in ["Big Picture" Changes](#).

## Envelope

This section describes changes to the envelope structure.

## Trailers

SOAP 1.1 allows additional elements to follow the SOAP Body element, SOAP 1.2 disallows these. [Table 1](#) illustrates this.

<pre> &lt;e:Envelope ...&gt;   &lt;e:Header&gt;     0 or more headers   &lt;/e:Header&gt;   &lt;e:Body&gt;     message body   &lt;/e:Body&gt;   other elements &lt;/e:Envelope&gt; </pre>	<pre> &lt;e:Envelope ...&gt;   &lt;e:Header&gt;     0 or more headers   &lt;/e:Header&gt;   &lt;e:Body&gt;     message body   &lt;/e:Body&gt; &lt;/e:Envelope&gt; </pre>
---	--

**Table 1**

## Actors and Roles

SOAP 1.1 has the `actor` attribute. In SOAP 1.2 this attribute is renamed to `role`. The semantics of the attribute are unchanged.

SOAP 1.2 adds two new predefined roles to the existing "Next" role in SOAP 1.1:

- "None" for header blocks that should never be directly processed.
- "Ultimate Receiver" for header blocks that should only be processed by nodes acting as the ultimate receiver of a message.

## Fault Codes

SOAP 1.2 adds new fault codes:

- "DataEncodingUnknown" - generated when a received message uses an unrecognised value of the `encodingStyle` attribute.
- The SOAP 1.1 "Client" fault code is renamed "Sender" in SOAP 1.2.
- The SOAP 1.1 "Server" fault code is renamed "Receiver" in SOAP 1.2.
- Additional faults are added for RPC, see [RPC](#).

## Modified Fault Structure

SOAP 1.2 faults are structured differently to SOAP 1.1. In particular all fault elements are now namespace qualified, many have been renamed and fault codes are now hierarchical (see [Hierarchical Fault Codes](#)). [Table 2](#) shows the mapping between SOAP 1.1 and

SOAP 1.2 faults.

<code>e:Fault</code>	<code>e:Fault</code>
<code>faultcode</code>	<code>e:Code, e:Subcode, e:Value</code>
<code>faultstring</code>	<code>e:Reason</code>
<code>faultactor</code>	<code>e:Node, e:Role</code>
<code>detail</code>	<code>e:Detail</code>

**Table 2**

**Hierarchical Fault Codes**

SOAP 1.1 allows extension of fault codes using a "dot" notation, SOAP 1.2 disallows this and provides a more XML-like representation instead. [Table 3](#) illustrates this.

<pre>&lt;e:Fault&gt;   &lt;faultcode&gt;e:Server.Memory&lt;/faultcode&gt;   &lt;faultstring&gt;Out of memory&lt;/faultstring&gt; &lt;/e:Fault&gt;</pre>	<pre>&lt;e:Fault&gt;   &lt;e:Code&gt;     &lt;e:Value&gt;e:Receiver&lt;/e:Value&gt;     &lt;e:Subcode&gt;       &lt;e:Value&gt;p:Memory&lt;/e:Value&gt;     &lt;/e:Subcode&gt;   &lt;/e:Code&gt;   &lt;e:Reason&gt;Out of memory&lt;/e:Reason&gt; &lt;/e:Fault&gt;</pre>
---	--

**Table 3**

**Modified Fault Semantics**

SOAP 1.2 constrains the body of a SOAP fault message to only contain a single child element, that element being the SOAP `Fault` element. The semantics of a SOAP message containing a SOAP `Fault` element plus sibling elements are not defined.

**Misunderstood Header**

SOAP 1.2 adds a new standard header for reporting additional information in "MustUnderstand" faults. In the example shown below, a receiver is reporting that it failed to understand two mandatory headers: `abc:Extension1` and `def:Extension2`.

```

<e:Envelope>
  <e:Header>
    <f:Misunderstood qname='abc:Extension1' />
    <f:Misunderstood qname='def:Extension2' />
  </e:Header>
  <e:Body>
    <e:Fault>
      <e:Code>
        <e:Value>e:MustUnderstand</e:Value>
      </e:Code>
      <e:Reason>Headers not understood</e:Reason>
    </e:Fault>
  </e:Body>
</e:Envelope>

```

## Processing Model

SOAP 1.2 clarifies the processing model of SOAP 1.1. In particular a SOAP node should process an incoming message according to the following rules in the order given:

1. Determine the roles the node is to play. I.e. the values of the `role` attribute for which it will assume processing liability.
2. Identify mandatory headers targetted to the node. I.e. header marked with a `mustUnderstand="true"` attribute and a `role` attribute with a value matching one of the roles determined in step 1.
3. Generate a "MustUnderstand" fault if one or more of the headers identified in step is not understood. I.e. the node does not contain software for processing the header.
4. Process headers and, if the node is the ultimate recipient, the body of the message.
5. If the node is acting as an intermediary remove headers targetted at the node and forward the message.

## Versioning Mechanism

SOAP 1.2 adds semantics for dealing with messages from different versions of SOAP:

- A SOAP 1.2 node that receives a SOAP 1.1 message will respond with a SOAP 1.1 envelope containing a SOAP 1.1 "VersionMismatch" fault.
- A SOAP 1.2 node that receives a message from any other version of SOAP (including future versions) will respond with a SOAP 1.2 envelope containing a SOAP 1.2 "VersionMismatch" fault and an `Upgrade` header with a list of the supported envelope versions.

## Upgrade Header

SOAP 1.2 adds an `Upgrade` header that is used to carry the supported envelope versions when a node reports a "VersionMismatch" fault. An example is shown below.

```

<V:Upgrade xmlns:V="http://www.w3.org/2001/12/soap-envelope">
  <envelope
    qname="ns1:Envelope"
    xmlns:ns1="http://schemas.xmlsoap.org/soap/envelope/" />
  <envelope
    qname="ns2:Envelope"
    xmlns:ns2="http://www.w3.org/2001/12/soap-envelope" />
</V:Upgrade>

```

## Binding Framework

SOAP 1.1 defines a single binding to HTTP. SOAP 1.2 defines an abstract binding framework that:

- Sets out requirements and concepts common to all binding specifications.
- Facilitates homogenous description of bindings that support common features.
- Facilitates homogenous description of optional features.

SOAP 1.2 also defines a concrete binding to HTTP and a non-normative email binding.

## Encoding

SOAP 1.2 provides a similar but simplified version of the encoding provided by SOAP 1.1. The following subsections describes the differences.

### encodingStyle Attribute

In SOAP 1.2 the encodingStyle attribute may only be used on *child* elements of the Body, Header and fault Detail elements. In SOAP 1.1, the attribute is allowed on any element in the envelope.

### Multi-reference Values

In SOAP 1.2 multireference values may be encoded in-place rather than as "top-level" elements. [Table 4](#) illustrates this.

<pre> &lt;e:Books&gt;   &lt;e:Book&gt;     &lt;title&gt;My Life and Work     &lt;/title&gt;     &lt;author href="#henryford" /&gt;   &lt;/e:Book&gt;   &lt;e:Book&gt;     &lt;title&gt;Today and Tomorrow     &lt;/title&gt;     &lt;author href="#henryford" /&gt;   &lt;/e:Book&gt; &lt;/e:Books&gt; &lt;author id="henryford"&gt;   &lt;name&gt;Henry Ford&lt;/name&gt; &lt;/author&gt; </pre>	<pre> &lt;e:Books&gt;   &lt;e:Book&gt;     &lt;title&gt;My Life and Work&lt;/title&gt;     &lt;author id="henryford" &gt;       &lt;name&gt;Henry Ford&lt;/name&gt;     &lt;/author&gt;   &lt;/e:Book&gt;   &lt;e:Book&gt;     &lt;title&gt;Today and Tomorrow&lt;/title&gt;     &lt;author ref="henryford" /&gt;   &lt;/e:Book&gt; &lt;/e:Books&gt; </pre>
---	---

**Table 4**

## References

Graph edges in SOAP encoding are represented differently in SOAP 1.1 and SOAP 1.2. [Table 5](#) illustrates this.

<pre> href uri-reference #henryford, http://example.com/picture.gif </pre>	<pre> ref IDREF henryford </pre>
--	----------------------------------

**Table 5**

## Arrays

SOAP 1.1 partially transmitted and sparse arrays have been removed in SOAP 1.2. In addition the SOAP 1.1 arrayType attribute has been replaced with itemType and arraySize attributes. [Table 6](#) illustrates this.

<pre>&lt;numbers enc:arrayType="xs:int[2]"&gt;   &lt;number&gt;3&lt;/number&gt;   &lt;number&gt;4&lt;/number&gt; &lt;/numbers&gt;</pre>	<pre>&lt;numbers enc:itemType="xs:int" enc:arraySize="2"&gt;   &lt;number&gt;3&lt;/number&gt;   &lt;number&gt;4&lt;/number&gt; &lt;/numbers&gt;</pre>
---	---

**Table 6**

## Application to XML Name Mapping

SOAP 1.2 defines an algorithm for mapping between application and XML names. This is required due to restrictions on allowed element names in XML.

## Serialisation Roots

SOAP 1.1 included a `root` attribute that could be used to mark roots of an encoded graph. SOAP 1.2 removes this attribute. See [RPC Encoding Constraint](#) for details of how to determine the element that represents an RPC invocation or response.

## Omitted Accessors

In SOAP 1.2 an omitted accessor is equivalent to NIL. The semantics of a NIL accessor are explicitly application dependent.

## RPC

This section describes the differences between the SOAP 1.2 and SOAP 1.1 RPC conventions.

## RPC Array Representation

SOAP 1.2 allows the use of either an array or a struct to represent RPC invocations and responses. SOAP 1.1 allows only structs.

The array based serialisation is useful in circumstances where the number of arguments is not known in advance or in languages that support only positional arguments. E.g. the following SOAP Body shows an invocation of a function that takes a variable number of arguments.

```
<e:Envelope xmlns:e='...'>
  <e:Body>
    <f:addNumbers xmlns:f='...' e:encodingStyle='...'>
      <n>10</n>
      <n>15</n>
      ...
      <n>3</n>
    </f:addNumbers>
  </e:Body>
</e:Envelope>
```

## RPC Encoding Constraint

SOAP 1.2 constrains the body of a SOAP message using the SOAP encoding for RPC to only contain a single child element, that child representing the RPC invocation or response struct or array.

E.g. the following SOAP Body would be valid according to the SOAP encoding, but not valid when used in conjunction with the RPC convention.

```
<e:Envelope xmlns:e='... '>
  <e:Body>
    <f:addNumbers xmlns:f='...' e:encodingStyle='... '>
      <n ref='aNum' />
      <n ref='aNum' />
    </f:addNumbers>
    <n id='aNum'>10</num>
  </e:Body>
</e:Envelope>
```

Whilst the following SOAP Body (which is an alternative serialisation of the same struct) would be valid according to the SOAP encoding and also valid when used in conjunction with the RPC convention.

```
<e:Envelope xmlns:e='... '>
  <e:Body>
    <f:addNumbers xmlns:f='...' e:encodingStyle='... '>
      <n id='aNum'>10</n>
      <n ref='aNum' />
    </f:addNumbers>
  </e:Body>
</e:Envelope>
```

## RPC Fault Codes

SOAP 1.2 adds new RPC specific fault subcodes:

- "ProcedureNotPresent"
- "BadArguments"

## RPC Return Values

SOAP 1.2 adds a new convention for representing RPC return values. [Table 7](#) illustrates this.

<pre>&lt;e:Body&gt;   &lt;m:GetTradePriceResponse&gt;     &lt;xxx&gt;34.5&lt;/xxx&gt;   &lt;/m:GetTradePriceResponse&gt; &lt;/e:Body&gt;</pre>	<pre>&lt;e:Body&gt;   &lt;m:GetTradePriceResponse&gt;     &lt;rpc:result&gt;34.5&lt;/rpc:result&gt;   &lt;/m:GetTradePriceResponse&gt; &lt;/e:Body&gt;</pre>
--	--

**Table 7**

## HTTP Binding

This section discusses differences between the SOAP 1.2 and SOAP 1.1 HTTP binding.

### Content Type

The media type for SOAP 1.2 has changed from `text/xml` used in SOAP 1.1 to `application/soap+xml`. The `application/soap+xml` media type is described in [\[application/soap+xml\]](#).

The SOAP 1.1 mandatory `SOAPAction` HTTP header has been removed in SOAP 1.2. In its place is an optional `action` parameter on the `application/soap+xml` media type.

### Miscellaneous Changes

SOAP 1.2 does not provide a binding to the experimental HTTP extension framework.

SOAP 1.2 provides a finer grained mapping between SOAP errors and HTTP status codes than SOAP 1.1.

**[SOAP 1.1]**

W3C Note "Simple Object Access Protocol (SOAP) 1.1", Don Box, David Ehnebuske, Gopal Kakivaya, Andrew Layman, Noah Mendelsohn, Henrik Nielsen, Satish Thatte, Dave Winer, 8 May 2000. (See <http://www.w3.org/TR/SOAP/>).

**[SOAP 1.2 - Part 0]**

W3C Working Draft "SOAP Version 1.2 Part 0: Primer", Nilo Mitra, Dec 2001. (See <http://www.w3.org/TR/soap12-part0/>).

**[SOAP 1.2 - Part 1]**

W3C Working Draft "SOAP Version 1.2 Part 1: Messaging Framework", Martin Gudgin, Marc Hadley, Jean-Jacques Moreau, Henrik Frystyk Nielsen, Dec 2001. (See <http://www.w3.org/TR/soap12-part1/>).

**[SOAP 1.2 - Part 2]**

W3C Working Draft "SOAP Version 1.2 Part 2: Adjuncts", Martin Gudgin, Marc Hadley, Jean-Jacques Moreau, Henrik Frystyk Nielsen, Dec 2001. (See <http://www.w3.org/TR/soap12-part2/>).

**[XML 1.0]**

W3C Recommendation "Extensible Markup Language (XML) 1.0 (Second Edition)", Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, 6 October 2000. (See <http://www.w3.org/TR/2000/REC-xml-20001006/>).

**[XML Infoset]**

W3C Recommendation "XML Information Set", John Cowan, Richard Tobin, 24 October 2001. (See <http://www.w3.org/TR/2001/REC-xml-infoset-20011024/>).

**[application/soap+xml]**

IETF "INTERNET DRAFT: The 'application/soap+xml' media type", M. Baker, M. Nottingham, January 14, 2002. (Work in progress). (See <http://www.ietf.org/internet-drafts/draft-baker-soap-media-reg-00.txt>).