The SOAP Service Description Language

Savas Parastatidis[1], Jim Webber[2]
Savas@Parastatidis.name, Jim@Webber.name

Abstract

This document introduces and defines the SOAP Service Description Language (SSDL). SSDL is a base framework used to describe contracts for Web Services which communicate by the transfer of SOAP messages. SSDL is extensible via additional protocol frameworks which contract authors define to support additional message exchange patterns, expose service behaviour in terms of multi-message interactions, or to perform higher-level activities like model checking. A number of such plugin frameworks are presented alongside this document.

SSDL documents

An Introduction to the SOAP Service Description Language [1]

SOAP Service Description Language (SSDL) v1.3 [2] (this document)

MEP SSDL Protocol Framework v1.3 [3]

CSP SSDL Protocol Framework v1.3 [4]

Rules SSDL Protocol Framework v1.3 [5]

Sequencing Constraints SSDL Protocol Framework v1.3 [6]

Status of this document

Version: 1.3

Date: April 2005

http://ssdl.org

Disclaimer

[1] School of Computing Science, University of Newcastle, Newcastle upon Tyne, NE1 7RU, UK
[2] ThoughtWorks Australia Pty. Ltd.

Table of Contents

1. Introduction

1.1. Motivation

SOAP [7] is the standard message transfer protocol for Web Services. However, the default description language for Web Services, WSDL [8], does not explicitly target SOAP but provides a generic framework for the description of network-exposed software artefacts. WSDL's protocol independence makes describing SOAP message transfers more complex than if SOAP had been assumed from the outset. While the motivation to define a language that can be used with other underlying transfer and transport technologies is understandable, the cost in terms of complexity of the solution is high. Furthermore, WSDL's focus on the interface abstraction for describing services makes it difficult to escape the object-oriented or remote procedure call mindset and focus on message-orientation as the means through which integration is achieved. Finally, though this is a known issue, it is difficult to use WSDL to describe infrastructure protocols that make use of SOAP headers.

The SOAP Service Description Language (SSDL) is an XML-based vocabulary for writing message-oriented contracts for Web Services. The SOAP Service Description Language focuses on the use of messages combined into protocols (arbitrary message exchange patterns) to describe a SOAP-based Web Service, and is intended to provide a natural fit with the SOAP model.[3]

1.2. Goals

The main goals of SSDL are:

- Define a language for describing Web Services contracts in terms of SOAP messages and protocols (message exchange patterns of arbitrary form);
- Use WS-Addressing [9] so as to enable SOAP message transfer over arbitrary transport protocols and to ensure a consistent addressing mechanism for all SOAP services;
- Make use of XInclude for componentisation;
- Be as simple as possible.

1.3. Example

For comparative purposes, the following example of an SSDL contract describes a SOAP service similar to the one found in the WSDL 2.0 Primer [10].

```
<?xml version="1.0" encoding="utf-8" ?>
<ssdl:contract targetNamespace="http://example.org/service/contract"
               xmlns:ssdl="urn:ssdl:v1">
  <ssdl:schemas>
    <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
               targetNamespace="http://exaxmple.org/service/schema.xsd">

      <xs:element name="AvailabilityCheckRequest">
        <xs:complexType>
          <xs:sequence>
            <xs:element  name="checkInDate" type="xs:date"/>
            <xs:element  name="checkOutDate" type="xs:date"/>
            <xs:element  name="roomType" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

      <xs:element name="AvailabilityCheckResponse" type="xs:double"/>

      <xs:element name="InvalidDataError" type="xs:string"/>
    </xs:schema>
  </ssdl:schemas>
```

[3] From now on it is assumed that a "Web Service" supports SOAP.

```
  <ssdl:messages targetNamespace="http://example.org/service/messages"
                 xmlns:tns="http://exaxmple.org/service/schema.xsd">

    <ssdl:message name="AvailabilityCheckRequestMsg">
      <ssdl:body ref="tns:AvailabilityCheckRequest"/>
    </ssdl:message>

    <ssdl:message name="AvailabilityCheckResponseMsg">
      <ssdl:body ref="tns:AvailabilityCheckResponse"/>
    </ssdl:message>

    <ssdl:fault name="InvalidDataErrorFaultMsg">
      <ssdl:code value="Sender"/>
    </ssdl:fault>

  </ssdl:messages>

  <ssdl:protocols>
    <ssdl:protocol targetNamespace="http://example.org/service/protocol"
                   xmlns:mep="urn:ssdl:mep:v1">
      <mep:in-out>
        <!-- First two messages are the request-response pair.
             Semantics of this MEP require that the value of the "Message ID"
             WS-Addressing header of the incoming message becomes the value
             of the "Relates To" header for the outgoing message. -->
        <ssdl:msgref ref="AvailabilityCheckRequestMsg" direction="in"/>
        <ssdl:msgref ref="AvailabilityCheckResponseMsg" direction="out"/>

        <!-- All other messages are faults -->
        <ssdl:msgref ref="InvalidDataErrorFaultMsg" direction="out"/>
      </mep:in-out>
    </ssdl:protocol>
  </ssdl:protocols>

  <ssdl:endpoints>
    <ssdl:endpoint xmlns:wsa="http://www.w3.org/2004/12/addressing">
      <wsa:Address>http://example.org/service</wsa:Address>
    </ssdl:endpoint>
  </ssdl:endpoints>
</ssdl:contract>
```

Example 1: A simple contract

Example 1 shows a simple contract using only the MEP [3] protocol framework that is delivered as part of this document. Other protocol frameworks are available for describing more complicated exchange patterns/protocols and to enable more sophisticated reasoning about Web Services (e.g. consistency checks, protocol deadlock checks, workflow support, etc). These other protocol frameworks may be standard frameworks or may be developed specifically to support third party integrations, yet each will be able to utilise the underlying SSDL base framework.

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 RFC 2119 [11].

This specification uses properties from the XML Information Set [12]. Such properties are denoted by square brackets and in bold, e.g. **[namespace name]**.

This specification uses properties from the information properties defined by the WS-Addressing specification [9]. Such properties are denoted by square brackets and in bold, e.g. **[property name]**.

This specification uses namespace prefixes throughout; they are listed in Table 1-1. Note that the choice of any namespace prefix is arbitrary and not semantically significant (see [12]).

We use the pseudo-schema notation used in WSDL 2.0 Core [8] as a convenient description of the structure of a component.

1.5. Namespaces

These namespaces and their prefixes are used throughout this document.

Prefix	Namespace	Notes
ssdl	urn:ssdl:v1	Where elements are not qualified with a namespace prefix, urn:ssdl:v1 is assumed
xs	http://www.w3.org/2001/XMLSchema	
wsa	http://www.w3.org/2004/12/addressing	
xi	http://www.w3.org/2001/XInclude	
soap	http://www.w3.org/2003/05/soap-envelope	

2. General Description

The SOAP Service Description Language (SSDL) is used to describe messages and message exchange patterns that a Web Service supports. SSDL makes the assumption that the distinguishing factor of Web Services, when compared to other distributed computing technologies, is the use of W3C's SOAP [7] distributed message processing model. The assumption that Web Services communicate by exchanging SOAP messages (over arbitrary transport protocols) makes the contract language simpler and easier to use. Moreover, the language does not need to define any bindings to underlying transport or application protocols but, instead, directly leverages the SOAP bindings. SSDL also assumes that all message addressing related requirements are met by W3C's WS-Addressing [9].

2.1. Contract

An SSDL document represents a contract which contains schema declarations, message structure descriptions, protocol definitions, and endpoint reference information about a Web Service.

2.2. Messages

The structure of the messages in a contract is the same as that defined by the SOAP specification [7]. Accordingly the transfer of both ordinary SOAP messages and SOAP faults can be described.

2.3. Protocol frameworks

SSDL provides a container for protocols to be placed. Protocols are defined using one of the externally specified protocol frameworks. A protocol definition represents the behaviour of a Web Service in terms of the message exchange patterns supported by that Web Service. SSDL provides a global element that protocol frameworks can use in order to reference the described messages in the contract. The direction of the message (i.e. "in" or "out") can be defined as well as the Action header from the WS-Addressing specification.

Multiple protocols can be defined using different protocol frameworks. Protocols can be used to define distinct behaviours that a Web Service supports or different protocol frameworks could be used to describe the same behaviours but using a different notation.

2.4. Endpoints

SSDL uses the WS-Addressing endpoint reference (EPR) structure for referencing a Web Service endpoint which supports the defined contract.

2.5. Componentisation

Since an SSDL contract is an XML document, XInclude [13] could be used for componentisation. However, an SSDL-specific mechanism is provided for the inclusion of parts a contract. This mechanism is defined in terms of a collection of XInclude statements that are assumed to exist

before the SSDL contact is processed. As a result, it is not necessary to define additional semantics for the inclusion of contracts or parts of contracts or look to external specifications to understand those semantics.

2.6. SSDL Current Limitations

The following issues are pending:

- The current SSDL design assumes SOAP 1.2. Changes may be required to be backwardly compatible with SOAP 1.1.
- SOAP Features are not considered in the current version of SSDL
- SOAP MEPs are not considered in the current version of SSDL
- Attributes in the soap:Header element cannot be described
- Attribute in the soap:Body element cannot be described
- Text content directly under soap:Body is not supported
- Defining the encoding style of sub-elements of header or body elements is not supported

3. Contract Structure

This section describes the SSDL document using the XML Information Set [12] model.

3.1. contract

```
<contract targetNamespace="xs:anyURI">
  <documentation /> ?
  <include /> *
  <schemas />
  <messages /> +
  <protocols /> ?
  <endpoints /> ?
</contract>
```

An SSDL document MUST contain a `contract` element information item with the following properties:

- A **[local name]** of “contract”
- A **[namespace name]** of “urn:ssdl:v1”
- The following element information items in its **[children]** property in order:
    - An OPTIONAL `documentation` element information item
    - Zero or more `include` element information item
    - A REQUIRED `schemas` element information item
    - One or more `messages` element information item
    - An OPTIONAL `protocols` element information item
    - An OPTIONAL `endpoints` element information item
    - A REQUIRED `targetNamespace` attribute information item

3.1.1. targetNamespace

The **[normalised value]** of the `targetNamespace` attribute information item is a URI (xs:anyURI) that uniquely identifies the contract. This URI value can be used as the **[normalised value]** of the `namespace` attribute information item of the `include` element information item.

The `targetNamespace` attribute information item has the following properties:

- A **[local name]** of “targetNamespace”
- A **[namespace name]** of “urn:ssdl:v1”

3.2. include

```
<contract>
  <include location="xs:anyURI" ?
           namespace="xs:anyURI" ?
  />
</contract>
```

An XInclude [13] `xinclude` information element MAY appear anywhere in an SSDL document with any Values for its Infoset properties, as long as the resulting document after the inclusion is a valid SSDL document. SSDL provides the `include` element information item as a syntactic shortcut. It is expected that SSDL document processors will treat an `include` element information item as if the associated `xi:xinclude` element information items were present instead.

An `include` information element item has the following properties:

- A **[local name]** of “include”
- A **[namespace name]** of “urn:ssdl:v1”
- An OPTIONAL `location` attribute information item
- An OPTIONAL `namespace` attribute information items

If neither the `location` nor the `namespace` attribute information items are present, the `include` element information item has no effect.

The `include` information element is provided as a syntactic convenience and its semantics are defined according to the following rules:

3.2.1. Only the `location` attribute information item is present

When only the `location` attribute information item is present, the `include` element information item is semantically equivalent to a number of `xi:xinclude` element information items before the SSDL Infoset was validated, as defined below. In this case, the following are included from the valid SSDL documented identified by the provided URI:

- All the **[children]** of the `schemas` element information item
- All the `messages` element information items
- All the `protocol` element information items
- All the `endpoint` element information items

The `include` information element item is considered to be semantically equivalent to the following being present before the SSDL is processed:

- An `xi:xinclude` element information item being present as the first child of the `schemas` information element item if there is no `documentation` element information item or immediately after it if there is. The `xi:include` information element item has the following values for its properties:
    - The `href` attribute information item has the same **[normalised value]** as the `location` attribute information item
    - The `parse` attribute information item has a **[normalised value]** of “xml”
    - The `xpointer` attribute information item has a **[normalised value]** of “xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract/ssdl:schemas/*)”

- An `xi:xinclude` element information item being present as a child of the `contract` information element item immediately after the `schemas` element information item with the following values for its properties:
    - The `href` attribute information item has the same **[normalised value]** as the `location` attribute information item
    - The `parse` attribute information item has a **[normalised value]** of "xml"
    - The `xpointer` attribute information item has a **[normalised value]** of "xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract/ssdl:messages)"
- An `xi:xinclude` element information item being present as the first child of the `protocols` information element item if there is no `documentation` element information item or immediately after it if there is. The `xi:include` information element item has the following values for its properties:
    - The `href` attribute information item has the same **[normalised value]** as the `location` attribute information item
    - The `parse` attribute information item has a **[normalised value]** of "xml"
    - The `xpointer` attribute information item has a **[normalised value]** of "xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract/ssdl:protocol/ssdl)"
- An `xi:xinclude` element information item being present as the first child of the `endpoints` information element item if there is no `documentation` element information item or immediately after it if there is. The `xi:include` information element item has the following values for its properties:
    - The `href` attribute information item has the same **[normalised value]** as the `location` attribute information item
    - The `parse` attribute information item has a **[normalised value]** of "xml"
    - The `xpointer` attribute information item has a **[normalised value]** of "xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract/ssdl:endpoints/ssdl:endpoint)"

For example, the following `include` element information item

```
<contract>
  <include location="URI" />
  <schemas />
  <messages />
  <protocols />
  <endpoints />
<contract />
```

is semantically equivalent to an SSDL document with the following `xi:xinclude` information element items being present

```
<contract xmlns:xi="http://www.w3.org/2003/XInclude"
  targetNamespace="http://example.org/service/contract">

  <schemas>
    <xi:xinclude href="URI" parse="xml"
                 xpointer="xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract/ssdl:schemas/*)" />
  </schemas>

  <xi:xinclude href="URI" parse="xml"
               xpointer="xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract/ssdl:messages)" />
  <messages />

  <protocols>
    <xi:xinclude href="URI" parse="xml"
                 xpointer="xmlns(ssdl=urn:ssdl:v1)
                           xpointer(/ssdl:contract/ssdl:protocol/ssdl)"/>
  </protocols>
```

```
  <endpoints>
    <xi:xinclude href="URI" parse="xml"
                 xpointer="xmlns(ssdl=urn:ssdl:v1)
                           xpointer(/ssdl:contract/ssdl:endpoints/ssdl:endpoint)"/>
  </endpoints>
</contract>
```

3.2.2. Only the `namespace` attribute information item is present

When only the `namespace` attribute information item is present, the `include` element information item is semantically equivalent to the presence of a number of `xi:xinclude` element information items before the SSDL Infoset was validated, as defined below. Also, it is assumed that a processor provided URI exists which is called for the purposes of this specification 'tmpURI'. The `contract` element information item of the SSDL contract to be included MUST have a `targetNamespace` attribute information item with its **[normalised value]** property equal [14] to the **[normalised value]** of the `namespace` attribute information item. The following element information items are included:

- All the **[children]** of the `schemas` element information item
- All the `messages` element information items
- All the `protocol` element information items
- All the `endpoint` element information items

The `include` information element item is considered to be semantically equivalent to the following being present before the SSDL is processed:

- An `xi:xinclude` element information item being present as the first child of the `schemas` information element item if there is no `documentation` element information item or immediately after it if there is. The `xi:include` information element item has the following values for its properties:
    - The `href` attribute information item has a **[normalised value]** assigned by the processor
    - The `parse` attribute information item has a **[normalised value]** of "xml"
    - The `xpointer` attribute information item has a **[normalised value]** of "xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract[targetNamespace='URI']/ssdl:schemas/*)"
- An `xi:xinclude` element information item being present as a child of the `contract` information element item immediately after the `schemas` element information item with the following values for its properties:
    - The `href` attribute information item has a **[normalised value]** assigned by the processor
    - The `parse` attribute information item has a **[normalised value]** of "xml"
    - The `xpointer` attribute information item has a **[normalised value]** of "xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract[targetNamespace='URI']/ssdl:messages)"
- An `xi:xinclude` element information item being present as the first child of the `protocols` information element item if there is no `documentation` element information item or immediately after it if there is. The `xi:include` information element item has the following values for its properties:
    - The `href` attribute information item has a **[normalised value]** assigned by the processor

- The `parse` attribute information item has a **[normalised value]** of “xml”
- The `xpointer` attribute information item has a **[normalised value]** of “xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract/ssdl:protocol[targetNamespace='URI']/ssdl)”

- An `xi:xinclude` element information item being present as the first child of the `endpoints` information element item if there is no `documentation` element information item or immediately after it if there is. The `xi:include` information element item has the following values for its properties:
  - The `href` attribute information item has a **[normalised value]** assigned by the processor
  - The `parse` attribute information item has a **[normalised value]** of “xml”
  - The `xpointer` attribute information item has a **[normalised value]** of “xmlns(ssdl=urn:ssdl:v1) xpointer(/ssdl:contract[targetNamespace='URI']/ssdl:endpoints/ssdl:endpoint)”

For example, the following `include` element information item

```
<contract>
  <include namespace="URI" />
  <schemas />
  <messages />
  <protocols />
  <endpoints />
<contract />
```

is semantically equivalent to an SSDL document with the following `xi:xinclude` information element items being present

```
<contract xmlns:xi="http://www.w3.org/2003/XInclude"
  targetNamespace="http://example.org/service/contract" >

  <schemas>
    <xi:xinclude href="tmpURI" parse="xml"
                 xpointer="xmlns(ssdl=urn:ssdl:v1)
                 xpointer(/ssdl:contract[targetNamespace='URI']/ssdl:schemas/*)" />
  </schemas>

  <xi:xinclude href="tmpURI" parse="xml"
               xpointer="xmlns(ssdl=urn:ssdl:v1)
                         xpointer(/ssdl:contract[targetNamespace='URI']/ssdl:messages)" />
  <messages />

  <protocols>
    <xi:xinclude href="tmpURI" parse="xml"
              xpointer="xmlns(ssdl=urn:ssdl:v1)
              xpointer(/ssdl:contract[targetNamespace='URI']/ssdl:protocol/ssdl)"/>
  </protocols>

  <endpoints>
    <xi:xinclude href="tmpURI" parse="xml"
              xpointer="xmlns(ssdl=urn:ssdl:v1)
              xpointer(/ssdl:contract[targetNamespace='URI']/ssdl:endpoints/ssdl:endpoint)"/>
  </endpoints>
</contract>
```

3.2.3. Both the `location` and the `namespace` attribute information items are present

When both the `location` and `namespace` attribute information items are present, then the semantics of the `include` element information element item are the same as if only the `namespace` attribute information item was present but the **[normalised value]** of the `location` attribute information item becomes the **[normalised value]** of all the `href` attribute information items.

3.2.4. location

The **[normalised value]** of the `location` attribute information item is the URI (xs:anyURI) of the SSDL document to include.

The `location` attribute information item has the following properties:

- A **[local name]** of "location"
- A **[namespace name]** of "urn:ssdl:v1"

3.2.5. namespace

The **[normalised value]** of the `namespace` attribute information item is the URI (xs:anyURI) namespace in which the messages to be included in the SSDL document are defined.

The `namespace` attribute information item has the following properties:

- A **[local name]** of "namespace"
- A **[namespace name]** of "urn:ssdl:v1"

3.3. schemas

```
<contract>
  <schemas>
    <documentation /> ?
    [extension elements] *
  </schemas>
</contract>
```

A `schemas` information element item is used as a container for schema documents.

A `schemas` information element item has the following properties:

- A **[local name]** of "schemas"
- A **[namespace name]** of "urn:ssdl:v1"
- The following element information items under **[children]:**
    - Zero or one `documentation` information element item
    - Zero or more element information items. The **[namespace]** property of these information elements must not be "urn:ssdl:v1"

3.4. messages

```
<contract>
  <messages targetNamespace="xs:anyURI">
    <documentation /> ?
    [<message> | <fault>] *
  </messages>
</contract>
```

The `messages` element information item is used to group `message` and `fault` element information items representing messages and faults which are declared in the same `targetNamespace`.

A `messages` information element item has the following properties:

- A **[local name]** of "messages"
- A **[namespace name]** of "urn:ssdl:v1"
- A REQUIRED `targetNamespace` attribute information item in its **[attributes]** property
- The following element information items in its **[children]** property in order**:**
    - Zero or one `documentation` information element item

- Zero or more `message` or `fault` element information items

3.4.1. targetNamespace

The **[normalised value]** of the `targetNamespace` attribute information item, when the **[owner element]** property is the `messages` element information item, is the URI (xs:anyURI) representing the namespace in which the `message` and `fault` element information items define the messages and faults respectively. Two `messages` element information items MAY have the same **[normalised value]** for their `targetNamespace` attribute information item.

The `targetNamespace` attribute information item has the following properties:

- A **[local name]** of "targetNamespace"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2. message

```
<contract>
  <messages>
    <message name="xs:string"
             headerOrdering="strict | lax" ?
             bodyOrdering="string | lax" ? >
      <documentation /> ?
      <header /> *
      <body /> *
    </message>
  </messages>
</contract>
```

A `message` element information item defines the name and structure of a SOAP message.

The `message` element information item has the following properties:

- A **[local name]** of "message"
- A **[namespace name]** of "urn:ssdl:v1"
- A REQUIRED `name` attribute information attribute under **[attributes]**
- An OPTIONAL `headerOrdering` attribute information attribute under **[attributes]**
- An OPTIONAL `bodyOrdering` attribute information attribute under **[attributes]**
- The following element information items in its **[children]** property in order:
    - Zero or one `documentation` information element item
    - Zero or more `header` element information items
    - Zero or more `body` element information items

3.4.2.1 name

The **[normalised value]** of the `name` attribute information item is the name of the defined message. The `name` attribute information item has the following properties:

- A **[local name]** of "name"
- A **[namespace name]** of "urn:ssdl:v1"

A `messages` element information item MUST NOT contain two `message` element information items with the same **[normalised value]** for the `name` attribute information item.

3.4.2.2 headerOrdering

The **[normalised value]** of the `headerOrdering` attribute information item specifies whether the order in which the `header` element information items appear is significant with respect to each other. The `name` attribute information item has the following properties:

- A **[local name]** of "headerOrdering"
- A **[namespace name]** of "urn:ssdl:v1"
- A **[normalised value]** of either "strict" or "lax"

If the attributed is omitted, the **[normalised value]** is assumed to be "lax".

3.4.2.3 bodyOrdering

The **[normalised value]** of the `bodyOrdering` attribute information item specifies whether the order in which the `body` element information items appear is significant with respect to each other. The `name` attribute information item has the following properties:

- A **[local name]** of "bodyOrdering"
- A **[namespace name]** of "urn:ssdl:v1"
- A **[normalised value]** of either "strict" or "lax"

If the attributed is omitted, the **[normalised value]** is assumed to be "lax".

3.4.2.4 header

```
<contract>
  <messages>
    <message>
      <header ref="xs:QName"
              role="xs:anyURI" ?
              mustUnderstand="xs:boolean" ?
              relay="xs:boolean" ?
              encodingStyle="xs:anyURI" ?
              minOccurs="xs:positiveInteger" ?
              maxOccurs="xs:positiveInteger | unbounded" ?
      />
    </message>
  </messages>
</contract>
```

The `header` element information item describes a SOAP header. It has the following properties:

- A **[local name]** of "header"
- A **[namespace name]** of "urn:ssdl:v1"
- The following attribute information items in its **[attributes]** property:
    - A REQUIRED `ref` attribute information item
    - An OPTIONAL `role` attribute information item
    - An OPTIONAL `mustUnderstand` attribute information item
    - An OPTIONAL `relay` attribute information item
    - An OPTIONAL `encodingStyle` attribute information item
    - An OPTIONAL `minOccurs` attribute information item
    - An OPTIONAL `maxOccurs` attribute information item

3.4.2.4.1 ref

The **[normalised value]** of the `ref` attribute information item, when its **[owner element]** property is the `header` element information item, identifies a qualified element name (xs:QName) to be used as a child of the `soap:Header` element information item. The `ref` attribute information item has the following properties:

- A **[local name]** of "ref"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.4.2 role

The **[normalised value]** of the `role` attribute information item SHOULD be used as the **[normalised value]** for the `soap:role` attribute information item of the SOAP message and it is a URI (xs:anyURI). It has the following properties:

- A **[local name]** of "role"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.4.3 mustUnderstand

The **[normalised value]** of the `mustUnderstand` attribute information item SHOULD be used as the **[normalised value]** for the `soap:mustUnderstand` attribute information item of the SOAP message and it is a Boolean (xs:boolean). It has the following properties:

- A **[local name]** of "mustUnderstand"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.4.4 relay

The **[normalised value]** of the `relay` attribute information item SHOULD be used as the **[normalised value]** for the `soap:relay` attribute information item of the SOAP message and it is a Boolean (xs:boolean). It has the following properties:

- A **[local name]** of "relay"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.4.5 encodingStyle

The **[normalised value]** of the `encodingStyle` attribute information item, when its **[owner element]** property is the `header` element information item, SHOULD be used as the **[normalised value]** for the `soap:encodingStyle` attribute information item of the defined `header` and it is a URI (xs:anyURI). It has the following properties:

- A **[local name]** of "encodingStyle"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.4.6 minOccurs

The **[normalised value]** of the `minOccurs` attribute information item, when its **[owner element]** property is the `header` element information item represents the minimum possible occupancies in a SOAP envelope expected by this protocol of the defined header and it is a positive integer (xs:positiveInteger). It has the following properties:

- A **[local name]** of "minOccurs"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.4.7 maxOccurs

The **[normalised value]** of the `maxOccurs` attribute information item, when its **[owner element]** property is the `header` element information item represents the maximum possible occupancies in a SOAP envelope expected by this protocol of the defined header and it is a positive integer (xs:positiveInteger) or has the value "unbounded". It has the following properties:

- A **[local name]** of "maxOccurs"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.5 body

```
<contract>
  <messages>
    <message>
      <body ref="xs:QName"
            encodingStyle="xs:anyURI" ?
            minOccurs="xs:positiveInteger" ?
            maxOccurs="xs:positiveInteger | unbounded" ?
      />
    </message>
  </messages>
</contract>
```

The body element information item describes a child element information item of a SOAP Body. It has the following properties:

- A **[local name]** of "body"
- A **[namespace name]** of "urn:ssdl:v1"
- The following attribute information items in its **[attributes]** property
    - A REQUIRED `ref` attribute information item
    - An OPTIONAL `encodingStyle` attribute information item
    - An OPTIONAL `minOccurs` attribute information item
    - An OPTIONAL `maxOccurs` attribute information item

3.4.2.5.1 ref

The **[normalised value]** of the `ref` attribute information item, when its **[owner element]** property is the `body` element information item, identifies a qualified element name (xs:QName) to be used as a child of the `soap:Body` element information item. The `ref` attribute information item has the following properties:

- A **[local name]** of "ref"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.5.2 encodingStyle

The **[normalised value]** of the `encodingStyle` attribute information item, when its **[owner element]** property is the `body` element information item, SHOULD be used as the **[normalised value]** for the `soap:encodingStyle` attribute information item of the defined `body` and it is a URI (xs:anyURI). It has the following properties:

- A **[local name]** of "encodingStyle"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.5.3 minOccurs

The **[normalised value]** of the `minOccurs` attribute information item, when its **[owner element]** property is the `body` element information item represents the minimum possible occupancies in a SOAP envelope expected by this protocol of the defined body element and it is a positive integer (xs:positiveInteger). It has the following properties:

- A **[local name]** of "minOccurs"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.2.5.4 maxOccurs

The **[normalised value]** of the `maxOccurs` attribute information item, when its **[owner element]** property is the `body` element information item represents the maximum possible occupancies in a SOAP envelope expected by this protocol of the defined body element and it is a positive integer (xs:positiveInteger) or has the value 'unbounded'. It has the following properties:

- A **[local name]** of "maxOccurs"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.3. fault

```
<contract>
  <messages>
    <fault name="xs:string">
      <documentation /> ?
      <code />
      <reason />
      <node /> ?
      <role /> ?
      <detail /> ?
    </fault>
  </messages>
</contract>
```

A `fault` element information item describes a SOAP fault message. It has the following properties:

- A **[local name]** of "fault"
- A **[namespace name]** of "urn:ssdl:v1"
- A `name` attribute information item in its **[attributes]** property
- The following element information items in its **[children]** property in order:
    - An OPTIONAL `documentation` element information item
    - A REQUIRED `code` element information item
    - A REQUIRED `reason` element information item
    - An OPTIONAL `node` element information item
    - An OPTIONAL `role` element information item
    - An OPTIONAL `detail` element information item

3.4.3.1 name

The **[normalised value]** of the `name` attribute information item is the name of the defined fault message. It has the following properties:

- A **[local name]** of "name"
- A **[namespace name]** of "urn:ssdl:v1"

A `messages` element information item MUST NOT contain two `fault` element information items with the same **[normalised value]** for the `name` attribute information item.

3.4.3.2 code

```
<contract>
  <messages>
    <fault>
      <code value="xs:string">
        <subcode /> ?
      </code>
    </fault>
  </messages>
</contract>
```

The contents of the `code` element information item SHOULD be used as the contents of the `soap:code` attribute information item in the defined fault message. It has the following properties:

- A **[local name]** of “code”
- A **[namespace name]** of “urn:ssdl:v1”
- A REQUIRED `value` attribute information item in its **[attributes]** property
- An OPTIONAL `subcode` element information item in its **[children]** property

3.4.3.2.1 *value* (with code as parent)

The **[normalised value]** of the `value` attribute information item, when the **[owner element]** property is the `code` element information item, SHOULD be used as the **[normalised value]** of the `soap:value` attribute information item (with the `soap:code` element information item as parent) for the defined fault message. Its type is *soap:faultCodeEnum*, as defined in the SOAP specification [7], and it has the following properties:

- A **[local name]** of “value”
- A **[namespace name]** of “urn:ssdl:v1”

3.4.3.2.2 subcode

```
<contract>
  <messages>
    <fault>
      <code>
        <subcode value="xs:string">
          <subcode /> ?
        </subcode>
      </code>
    </fault>
  </messages>
</contract>
```

The contents of the `subcode` element information item SHOULD be used as the contents of the `soap:subcode` element information item for the defined fault message. It has the following properties:

- A **[local name]** of “subcode”
- A **[namespace name]** of “urn:ssdl:v1”
- A REQUIRED `value` attribute information element item in its **[attributes]** property.
- An OPTIONAL `subcode` element information element item with the same Infoset as the one defined in this subsection.

3.4.3.2.3 value (with subcode as parent)

The **[normalised value]** of the `value` attribute information item, when the **[owner element]** property is the `subcode` element information item, SHOULD be used as the **[normalised value]** of the `soap:value` attribute information item (with the `soap:subcode` element information item as parent) for the defined fault message. Its type is *soap:faultCodeEnum*, as defined in the SOAP specification [7], and it has the following properties:

- A **[local name]** of "value"
- A **[namespace name]** of "urn:ssdl:v1"

3.4.3.3 reason

```
<contract>
  <messages>
    <fault>
      <reason>
        <text /> +
      </reason>
    </fault>
  </messages>
</contract>
```

The contents of the `reason` element information item SHOULD be used as the contents of the `soap:reason` element information item for the defined fault message. It has the following properties:

- A **[local name]** of "reason"
- A **[namespace name]** of "urn:ssdl:v1"
- One or more `text` element information items in its **[children]** property.

3.4.3.3.1 text

```
<contract>
  <messages>
    <fault>
      <reason>
        <text xml:lang>xs:string</text>
      </reason>
    </fault>
  </messages>
</contract>
```

The contents of the `text` attribute information item SHOULD be used as the contents of the `soap:text` element information item for the defined fault message. It has the following properties:

- A **[local name]** of "text"
- A **[namespace name]** of "urn:ssdl:v1"
- The `lang` attribute information item from the XML Namespace (http://www.w3.org/XML/1998/namespace) and with its **[prefix]** property set to "xml". Each `text` element information item SHOULD have a different **[normalised value]** for `lang`.
- Any number of character information items in its **[children]** property.

3.4.3.4 node

```
<contract>
  <messages>
    <fault>
      <node>anyURI</node>
    </fault>
  </messages>
```

```
</contract>
```

The contents of the `node` element information item SHOULD be used as the contents of the `soap:node` element information item. It has the following properties:

- A **[local name]** of “node”
- A **[namespace name]** of “urn:ssdl:v1”
- Any number of character information items in its **[children]** property that make up a URI (xs:anyURI).

3.4.3.5 role

```
<contract>
  <messages>
    <fault>
      <role>anyURI</role>
    </fault>
  </messages>
</contract>
```

The contents of the `role` element information item SHOULD be used as the contents of the `soap:role` element information item. It has the following properties:

- A **[local name]** of “role”
- A **[namespace name]** of “urn:ssdl:v1”
- Any number of character information items in its **[children]** property that make up a URI (xs:anyURI).

3.4.3.6 detail

```
<contract>
  <messages>
    <fault>
      <detail anyAttribute>any mixed content</detail>
    </fault>
  </messages>
</contract>
```

The contents of the `detail` element information item SHOULD be used as the contents of the `soap:detail` element information item. It has the following properties:

- A **[local name]** of “detail”
- A **[namespace name]** of “urn:ssdl:v1”
- Zero or more attribute information items in its **[attributes]** property
- Zero or more element information items in its **[children]** property
- Any number of character information items in its **[children]** property

3.5. protocols

```
<contract>
  <protocols>
    <documentation /> ?
    <protocol> *
  </protocols>
</contract>
```

The `protocols` element information item is a container for the protocol framework specific elements. It has the following properties:

- A **[local name]** of “protocols”
- A **[namespace name]** of “urn:ssdl:v1”

- Zero or one `documentation` element information items
- Zero or more `protocol` element information items

3.5.1. protocol

```
<contract>
  <protocols>
    <protocol targetNamespace="xs:anyURI"
              name="xs:string" ? >
      <documentation /> ?
      [extension elements] *
    </protocol>
  </protocols>
</contract>
```

The `protocol` element information item contains the definition of a protocol using a protocol framework. It has the following properties:

- A **[local name]** of "protocol"
- A **[namespace name]** of "urn:ssdl:v1"
- Zero or more element information items defined in a namespace other than "urn:ssdl:v1"
- A REQUIRED `targetNamespace` attribute information item
- An OPTIONA `name` attribute information item

3.5.1.1 targetNamespace

The **[normalised value]** of the `targetNamespace` attribute information item, when the **[owner element]** property is the `protocol` element information item, is the URI (xs:anyURI) representing the namespace in which the `protocol` element information item defines the protocol.

The `targetNamespace` attribute information item has the following properties:

- A **[local name]** of "targetNamespace"
- A **[namespace name]** of "urn:ssdl:v1"

More than one `protocol` element information items MAY have the same **[normalised value]** for their `targetNamespace` attribute information item. If two or more `protocol` element information items have the same **[normalised value]** for their `targetNamespace` attribute information item, they MUST semantically describe the same protocol.

3.5.1.2 name

The **[normalised value]** of the `name` attribute information item, when the **[owner element]** property is the `protocol` element information item represents the name of the defined protocol, which can be used for reference purposes.

The `name` attribute information item has the following properties:

- A **[local name]** of "name"
- A **[namespace name]** of "urn:ssdl:v1"

3.6. endpoints

```
<contract>
  <endpoints>
    <documentation /> ?
    <endpoint /> *
  </endpoints>
</contract>
```

The `endpoints` element information item contains the `endpoint` element information items of the described service. It has the following properties:

- A **[local name]** of “endpoints”
- A **[namespace name]** of “urn:ssdl:v1”
- Zero or more `endpoint` element information items in its **[children]** property

3.6.1. endpoint

An `endpoint` element information item contains the Endpoint Reference (EPR) information elements as defined in the WS-Addressing specification. The Web Service at each `endpoint` SHOULD support all the described messages and protocols in the contract. The `endpoint` element information item has the following properties:

- A **[local name]** of “endpoint”
- A **[namespace name]** of “urn:ssdl:v1”
- Zero or more element information items in its **[children]** property that represent an Endpoint Reference, as defined in the WS-Addressing specification

3.7. msgref

```
<msgref ref="xs:Qname"
        direction="in" | "out"
        action="anyURI" ?
        [ extension attributes ] * />
  [ extension elements ] *
</msgref>
```

A global `msgref` element information item is defined by SSDL that defines the semantics of referencing messages and faults in an SSDL contract. It is expected that protocol framework designers will use the `msgref` element information item when referring to a message or a fault. It has the following properties:

- A **[local name]** of “msgref”
- A **[namespace name]** of “urn:ssdl:v1”
- A REQUIRED `ref` attribute information item
- A REQUIRED `direction` attribute information item
- An OPTIONAL `action` attribute information item
- Zero or more attribute information items with their **[namespace]** different from urn:ssdl:v1
- Zero or more element information items with their **[namespace]** different from urn:ssdl:v1

3.7.1. ref

The **[normalised value]** of the `ref` attribute information item, when its **[owner element]** property is the `msgref` element information item, identifies the qualified element name (xs:QName) of the referenced message or fault. The `ref` attribute information item has the following properties:

- A **[local name]** of “ref”
- A **[namespace name]** of “urn:ssdl:v1”

3.7.2. direction

The **[normalised value]** of the `direction` attribute information item identifies the direction (“in” or “out”) of the referenced message or fault. “in” means that the message is received while “out”

means that the message is sent. The `direction` attribute information item has the following properties:

- A **[local name]** of “direction”
- A **[namespace name]** of “urn:ssdl:v1”

Only “in” or “out” are valid values for the `direction` attribute information item.

3.7.3. action

The **[normalised value]** of the `action` attribute information item SHOULD be used as the **[normalised value]** of the `wsa:Action` header information element. If the `action` attribute information item is not defined, then its **[normalised value]** is assumed to be “urn:ssdl:v1:ProcessMessage”. The `action` attribute information item has the following properties:

- A **[local name]** of “action”
- A **[namespace name]** of “urn:ssdl:v1”

3.8. documentation

A `documentation` element information item is used to capture human readable information. It has the following properties:

- A **[local name]** of “documentation”
- A **[namespace name]** of “urn:ssdl:v1”
- The following child element information items under **[children]:**
    - Zero or more child element information items. The **[namespace]** property of these information elements must not be “urn:ssdl:v1”
    - Zero or more child character information items

References

[1] S. Parastatidis, J. Webber, S. Woodman, D. Kuo, and P. Greenfield, "An Introduction to the SOAP Service Description Language," School of Computing Science, University of Newcastle, Newcastle upon Tyne CS-TR-898, 2005.

[2] S. Parastatidis, J. Webber, S. Woodman, D. Kuo, and P. Greenfield, "SOAP Service Description Language (SSDL)," School of Computing Science, University of Newcastle, Newcastle upon Tyne CS-TR-899, 2005.

[3] S. Parastatidis and J. Webber, "MEP SSDL Protocol Framework," School of Computing Science, University of Newcastle, Newcastle upon Tyne CS-TR-900, 2005.

[4] S. Parastatidis and J. Webber, "CSP SSDL Protocol Framework," School of Computing Science, University of Newcastle, Newcastle upon Tyne CS-TR-901, 2005.

[5] D. Kuo, S. Parastatidis, and J. Webber, "Rules SSDL Protocol Framework," School of Computing Science, University of Newcastle, Newcastle upon Tyne CS-TR-902, 2005.

[6] S. Woodman, S. Parastatidis, and J. Webber, "Sequencing Constraints SSDL Protocol Framework," School of Computing Science, University of Newcastle, Newcastle upon Tyne CS-TR-903, 2005.

[7] W3C, "SOAP 1.2 Part 1: Messaging Framework," M. Gudgin, M. Hadley, N. Mendelsohn, J. J. Moreau, and H. F. Nielsen, Eds. http://www.w3.org/TR/2003/REC-soap12-part1-20030624/: W3C, 2003.

[8] W3C, "Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language," R. Chinnici, M. Gudgin, J.-J. Moreau, J. Schlimmer, and S. Weerawarana, Eds. http://www.w3.org/TR/2004/WD-wsdl20-20040803/, 2004.

[9] W3C, "Web Services Addressing (WS-Addressing)." http://www.w3.org/2002/ws/addr/.

[10] W3C, "Web Services Description Language (WSDL) Version 2.0 Part 0: Primer," R. Chinnici, M. Gudgin, J.-J. Moreau, J. Schlimmer, and S. Weerawarana, Eds. http://www.w3.org/TR/2004/WD-wsdl20-primer-20041221/, 2004.

[11] S. Bradner, "IETF RFC 2119: Key words for use in RFCs to Indicate Requirement Levels." http://www.ietf.org/rfc/rfc2119.txt: Internet Engineering Task Force, 1999.

[12] W3C, "XML Information Set." http://www.w3.org/TR/xml-infoset/, 2004.

[13] W3C, "XML Inclusions (XInclude) Version 1.0." http://www.w3.org/TR/xinclude/, 2004.

[14] T. Berners-Lee, W3C/MIT, R. Fielding, D. Software, L. Masinter, and Adobe, "Uniform Resource Identifier (URI): Generic Syntax." http://www.ietf.org/internet-drafts/draft-fielding-uri-rfc2396bis-07.txt, 2004.

Acknowledgments

The authors would like to thank Simon Woodman (Simon.Woodman@newcastle.ac.uk, School of Computing Science, University of Newcastle upon Tyne, UK) for his significant contributions to the editing of this specification.

Appendix A – XML Schema

The non-normative XML Schema for the SSDL contract document.

```
<?xml version="1.0" encoding="utf-8" ?>
<!--
    $Modtime: 5/03/05 15:27 $
    $Revision: 17 $
-->
<xs:schema
      elementFormDefault="qualified"
      targetNamespace="urn:ssdl:v1"
      xmlns:ssdl="urn:ssdl:v1"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:wsa="http://www.w3.org/2004/12/addressing">

   <xs:import namespace="http://www.w3.org/2004/12/addressing"/>
   <xs:import namespace="http://www.w3.org/XML/1998/namespace"/>

   <xs:element name="contract">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="documentation" type="xs:string"
                        minOccurs="0" maxOccurs="1" />
            <xs:element name="include" type="ssdl:include-type"
                        minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="schemas" type="ssdl:schemas-type"
                        minOccurs="1" maxOccurs="1"/>
            <xs:element name="messages" type="ssdl:messages-type"
                        minOccurs="1" maxOccurs="unbounded"/>
            <xs:element name="protocols" type="ssdl:protocols-type"
                        minOccurs="0" maxOccurs="1"/>
            <xs:element name="endpoints" type="ssdl:endpoints-type"
                        minOccurs="0" maxOccurs="1"/>
         </xs:sequence>
         <xs:attribute name="targetNamespace" type="xs:anyURI" use="required" />
      </xs:complexType>
   </xs:element>

   <xs:complexType name="include-type">
      <xs:attribute name="location" use="required" />
      <xs:attribute name="namespace" use="optional" />
   </xs:complexType>

   <xs:complexType name="schemas-type">
      <xs:sequence>
         <xs:element name="documentation" type="xs:string"
                     minOccurs="0" maxOccurs="1" />
         <xs:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
```

```
   </xs:complexType>

   <xs:complexType name="messages-type">
      <xs:sequence>
         <xs:element name="documentation" type="xs:string"
                     minOccurs="0" maxOccurs="1" />
         <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="message" type="ssdl:message-type" />
            <xs:element name="fault" type="ssdl:fault-type" />
         </xs:choice>
      </xs:sequence>
      <xs:attribute name="targetNamespace" type="xs:anyURI" use="required" />
   </xs:complexType>

   <xs:complexType name="message-type">
      <xs:sequence>
         <xs:element name="documentation" type="xs:string"
                     minOccurs="0" maxOccurs="1" />
         <xs:element name="header" type="ssdl:header-type"
                     minOccurs="0" maxOccurs="unbounded" />
         <xs:element name="body" type="ssdl:body-type"
                     minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="name" type="xs:string" use="required" />
      <xs:attribute name="headerOrdering" type="ssdl:ordering-type" use="optional" />
      <xs:attribute name="bodyOrdering" type="ssdl:ordering-type" use="optional" />
   </xs:complexType>

   <xs:simpleType name="ordering-type">
      <xs:restriction base="xs:string">
         <xs:enumeration value="strict" />
         <xs:enumeration value="lax" />
      </xs:restriction>
   </xs:simpleType>

   <xs:complexType name="header-type">
      <xs:attribute name="ref" type="xs:QName" use="required" />
      <xs:attribute name="role" type="ssdl:role-type" use="optional" />
      <xs:attribute name="mustUnderstand" type="xs:boolean" use="optional" />
      <xs:attribute name="relay" type="xs:boolean" use="optional" />
      <xs:attribute name="encodingStyle" type="xs:anyURI" use="optional" />
      <xs:attribute name="minOccurs" type="xs:positiveInteger" use="optional" />
      <xs:attribute name="maxOccurs" type="xs:positiveInteger" use="optional" />
   </xs:complexType>

   <xs:complexType name="body-type">
      <xs:attribute name="ref" type="xs:QName" use="required" />
      <xs:attribute name="encodingStyle" type="xs:anyURI" use="optional" />
      <xs:attribute name="minOccurs" type="xs:positiveInteger" use="optional" />
      <xs:attribute name="maxOccurs" type="ssdl:maxOccurs-type" use="optional" />
   </xs:complexType>

   <xs:simpleType name="maxOccurs-type">
      <xs:union memberTypes="xs:nonNegativeInteger">
         <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
               <xs:enumeration value="unbounded"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:union>
   </xs:simpleType>

   <xs:simpleType name="role-type" final="#all">
      <xs:union>
         <xs:simpleType>
            <xs:list itemType="xs:anyURI" />
         </xs:simpleType>
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:enumeration
                  value="http://www.w3.org/2003/05/soap-envelope/role/next" />
               <xs:enumeration
                  value="http://www.w3.org/2003/05/soap-envelope/role/ultimateReceiver" />
               <xs:enumeration
                  value="http://www.w3.org/2003/05/soap-envelope/role/none" />
            </xs:restriction>
         </xs:simpleType>
```

```
      </xs:union>
   </xs:simpleType>

   <xs:complexType name="fault-type">
      <xs:sequence>
         <xs:element name="documentation" type="xs:string"
                     minOccurs="0" maxOccurs="1" />
         <xs:element name="code" type="ssdl:fault-code-type"
                     minOccurs="1" maxOccurs="1" />
         <xs:element name="reason" type="ssdl:fault-reason-type"
                     minOccurs="1" maxOccurs="1" />
         <xs:element name="node" type="xs:anyURI"
                     minOccurs="0" maxOccurs="1"/>
         <xs:element name="role" type="ssdl:role-type"
                     minOccurs="0" maxOccurs="1"/>
         <xs:element name="detail" type="ssdl:fault-detail-type"
                     minOccurs="0" maxOccurs="1" />
      </xs:sequence>
      <xs:attribute name="name" type="xs:string" use="required" />
   </xs:complexType>

   <xs:complexType name="fault-code-type">
      <xs:sequence>
         <xs:element name="subcode" type="ssdl:fault-subcode-type"
                     minOccurs="0" maxOccurs="1" />
      </xs:sequence>
      <xs:attribute name="value" use="required">
         <xs:simpleType>
            <xs:restriction base="xs:string">
               <xs:enumeration value="VersionMismatch" />
               <xs:enumeration value="MustUnderstand" />
               <xs:enumeration value="DataEncodingUnknown" />
               <xs:enumeration value="Sender" />
               <xs:enumeration value="Receiver" />
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
   </xs:complexType>

   <xs:complexType name="fault-subcode-type">
      <xs:sequence>
         <xs:element name="subcode" type="ssdl:fault-subcode-type"
                     minOccurs="0" maxOccurs="1" />
      </xs:sequence>
      <xs:attribute name="value" type="xs:QName" use="required" />
   </xs:complexType>

   <xs:complexType name="fault-reason-type">
      <xs:sequence>
         <xs:element name="text" type="xs:string"
                     minOccurs="1" maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute ref="xml:lang" use="required" />
   </xs:complexType>

   <xs:complexType name="fault-detail-type" mixed="true">
      <xs:sequence>
         <xs:any namespace="##other" processContents="skip"
                 minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
      <xs:anyAttribute namespace="##other" processContents="skip" />
   </xs:complexType>

   <xs:element name="msgref">
      <xs:complexType>
         <xs:sequence>
            <xs:any namespace="##other" processContents="skip"
                    minOccurs="0" maxOccurs="unbounded" />
         </xs:sequence>
         <xs:attribute name="ref" type="xs:QName" use="required" />
         <xs:attribute name="direction" use="required">
            <xs:simpleType>
               <xs:restriction base="xs:string">
                  <xs:enumeration value="in" />
                  <xs:enumeration value="out" />
               </xs:restriction>
            </xs:simpleType>
```

```
         </xs:attribute>
         <xs:attribute name="action" type="xs:anyURI" default="urn:ssdl:v1:ProcessMessage" />
         <xs:anyAttribute namespace="##other" processContents="skip" />
      </xs:complexType>
   </xs:element>

   <xs:complexType name="protocols-type">
      <xs:sequence>
         <xs:element name="documentation" type="xs:string"
                     minOccurs="0" maxOccurs="1" />
         <xs:element name="protocol" type="ssdl:protocol-type"
                     minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="protocol-type">
      <xs:sequence>
         <xs:element name="documentation" type="xs:string"
                     minOccurs="0" maxOccurs="1" />
         <xs:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="targetNamespace" type="xs:anyURI" use="required" />
      <xs:attribute name="name" type="xs:string" use="optional" />
   </xs:complexType>

   <xs:complexType name="endpoints-type">
      <xs:sequence>
         <xs:element name="documentation" type="xs:string"
                     minOccurs="0" maxOccurs="1" />
         <xs:element name="endpoint" type="wsa:EndpointReferenceType"
                     minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
   </xs:complexType>

</xs:schema>
```