INTERNATIONAL STANDARD

ISO/IEC 24824-2

First edition
2006-07-01

Information technology — Generic applications of ASN.1: Fast Web Services

Technologies de l'information — Applications génériques de ASN.1: Services web rapides

Reference number
ISO/IEC 24824-2:2006(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

CONTENTS

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 24824-2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems,* in collaboration with ITU-T. The identical text is published as ITU-T Rec. X.892.

ISO/IEC 24824 consists of the following parts, under the general title *Information technology — Generic applications of ASN.1*:

— *Part 1: Fast Infoset*

— *Part 2: Fast Web Services*

The following parts are under preparation:

— *Part 3: Fast Infoset security*

Introduction

This Recommendation | International Standard specifies the use of ASN.1 (see ITU-T Rec. X.680 | ISO/IEC 8824-1), its Packed Encoding Rules (see ITU-T Rec. X.691 | ISO/IEC 8825-2) and Fast Infoset (see ITU-T Rec. X.891 | ISO/IEC 24824-1) to provide Fast Web Services. (For a general tutorial on Fast Web Services, see Annex C).

Clause 6 specifies the architectural model and the conceptual steps of producing and processing SOAP messages encoded using ASN.1 binary encodings (called "ASN.1 SOAP messages").

Clauses 7 to 9 contain general provisions for the processing of ASN.1 SOAP messages. Clause 7 specifies the mapping of ASN.1 SOAP messages to W3C SOAP messages. Clause 8 specifies the mapping of W3C SOAP messages to ASN.1 SOAP messages. Clause 9 specifies the extended SOAP processing model for processing embedded ASN.1 encoded values present in W3C SOAP messages.

Clause 10 specifies the ASN.1 SOAP HTTP Binding for the transfer of ASN.1 SOAP messages using HTTP as the transport protocol. This binding uses the Multipurpose Internet Mail Extensions (MIME) media type specified in B.1.

Clause 11 specifies the use of the W3C SOAP HTTP Binding for the transfer of W3C SOAP messages encoded as fast infoset documents (fast infoset SOAP messages). This binding uses the Multipurpose Internet Mail Extensions (MIME) media type specified in B.2.

Clause 12 specifies SOAP-oriented service descriptions that support the ASN.1 SOAP binding interface and Fast Web Services.

Clause 13 specifies how a SOAP-oriented service description affects the exchange of ASN.1 SOAP messages that are mapped to and from W3C SOAP messages.

Annex A forms an integral part of this Recommendation | International Standard, and contains the full ASN.1 module for ASN.1 SOAP.

Annex B forms an integral part of this Recommendation | International Standard, and contains the specification of the "**application/fastsoap**" and "**application/soap+fastinfoset**" media types.

Annex C does not form an integral part of this Recommendation | International Standard, and provides tutorial material on Fast Web Services.

Annex D does not form an integral part of this Recommendation | International Standard, and provides tutorial material on the interoperation of Fast Web Services and XML Web services using features of the ASN.1 SOAP HTTP Binding.

Annex E does not form an integral part of this Recommendation | International Standard, and shows how the exchange of ASN.1 SOAP messages can be described by WSDL 1.1 [2] service descriptions.

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – Generic applications of ASN.1: Fast Web Services

1 Scope

This Recommendation | International Standard specifies the messages and encodings that enable the use of Fast Web Services, together with the means of description of such services.

The protocol used to support these services satisfies the requirements of the SOAP processing model (see W3C SOAP Part 1, clause 2) and is based on the transfer of:

a) ASN.1 SOAP messages that contain embedded ASN.1 encoded values and embedded fast infoset documents; and

b) fast infoset SOAP messages.

This Recommendation | International Standard also specifies:

– an ASN.1 module for ASN.1 SOAP that defines the **`Envelope`** type (a value of this type corresponds to an ASN.1 SOAP message);

– a conceptual mapping between ASN.1 SOAP messages and W3C SOAP messages (defined as an instance of the XML Infoset, see W3C SOAP Part 1, clause 5);

– an extension to the W3C SOAP processing model for the processing of embedded ASN.1 encoded values;

– the ASN.1 SOAP HTTP Binding, which is a modification and extension of the W3C SOAP HTTP Binding (see W3C SOAP Part 2, clause 7), for the transfer of ASN.1 SOAP messages;

– support for the transfer of W3C SOAP message infosets serialized as fast infoset documents (fast infoset SOAP messages) using the W3C SOAP HTTP Binding (see W3C SOAP Part 2, clause 7);

– SOAP-oriented service descriptions that define the interface to and the semantics of Fast Web Services.

Two Multipurpose Internet Mail Extensions (MIME) media type names are allocated to identify:

– ASN.1 SOAP messages encoded using Basic Aligned PER;

– fast infoset SOAP messages.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations. The IETF maintains a list of RFCs, together with those that have been obsoleted by later RFCs.

The reference to a document within this Recommendation | International Standard does not give it, as a stand-alone document, the status of a Recommendation or International Standard.

2.1 Identical Recommendations | International Standards

– ITU-T Recommendation X.660 (2004) | ISO/IEC 9834-1:2005, *Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: General procedures and top arcs of the ASN.1 Object Identifier tree.*

– ITU-T Recommendation X.680 (2002) | ISO/IEC 8824-1:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*

- ITU-T Recommendation X.681 (2002) | ISO/IEC 8824-2:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification*.
- ITU-T Recommendation X.682 (2002) | ISO/IEC 8824-3:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification*. †
- ITU-T Recommendation X.683 (2002) | ISO/IEC 8824-4:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*. †
- ITU-T Recommendation X.690 (2002) | ISO/IEC 8825-1:2002, *Information technology – ASN.1 encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER), and Distinguished Encoding Rules (DER)*. †
- ITU-T Recommendation X.691 (2002) | ISO/IEC 8825-2:2002, *Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)*.
- ITU-T Recommendation X.692 (2002) | ISO/IEC 8825-3:2002, *Information technology – ASN.1 encoding rules: Specification of Encoding Control Notation (ECN)*. †
- ITU-T Recommendation X.693 (2001) | ISO/IEC 8825-4:2002, *Information technology – ASN.1 encoding rules: XML Encoding Rules (XER)* plus Amendment 1: XER Encoding Instructions and EXTENDED-XER. †
- ITU-T Recommendation X.694 (2004) | ISO/IEC 8825-5:2004, *Information technology – ASN.1 encoding rules: Mapping W3C XML Schema Definitions into ASN.1*.
- ITU-T Recommendation X.891 (2005) | ISO/IEC 24824-1:2005, *Information technology – Generic Applications of ASN.1: Fast Infoset*.

NOTE – The complete set of ASN.1 Recommendations | International Standards are listed above, as they can all be applicable in particular uses of this Recommendation | International Standard. Where these are not directly referenced in the body of this Recommendation | International Standard, a † symbol is added to the reference.

2.2 Additional references

- W3C SOAP:2003, *SOAP Version 1.2 Part 1: Messaging Framework, W3C Recommendation, Copyright © [24 June 2003] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2003/REC-soap12-part1-20030624.*
- W3C SOAP:2003, *SOAP Version 1.2 Part 2: Adjuncts, W3C Recommendation, Copyright © [24 June 2003] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2003/REC-soap12-part2-20030624.*
- W3C XML 1.0:2004, *Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation, Copyright © [4 February 2004] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2000/REC-xml-20040204/.*
- W3C XML Information Set:2004, *XML Information Set (Second Edition), W3C Recommendation, Copyright © [04 February 2004] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2004/REC-xml-infoset-20040204/.*
- W3C XML Namespaces 1.0:1999, Namespaces in XML, W3C Recommendation, Copyright © [14 January 1999] World Wide Web Consortium (*Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/1999/REC-xm-lnames-19990114/.*
- W3C XML Schema:2001, *XML Schema Part 1: Structures, W3C Recommendation, Copyright © [2 May 2001] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/.*
- W3C XML Schema:2001, *XML Schema Part 2: Datatypes, W3C Recommendation, Copyright © [2 May 2001] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/.*

  NOTE – When the reference "W3C XML Schema" is used in this Recommendation | International Standard, it refers to W3C XML Schema Part 1 and W3C XML Schema Part 2.
- IETF RFC 2045 (1996), *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies.*

– IETF RFC 2616 (1999), *Hypertext Transfer Protocol – HTTP/1.1.*

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply.

3.1 Imported definitions

3.1.1 This Recommendation | International Standard uses the following terms defined in ITU-T Rec. X.680 | ISO/IEC 8824-1:

a) abstract value;

b) module;

c) object identifier;

d) relative object identifier;

e) type.

3.1.2 This Recommendation | International Standard also uses the following terms defined in W3C XML Schema:

a) **complex type definition**;

b) **element declaration**;

c) schema;

d) schema component;

e) **simple type definition**.

3.1.3 This Recommendation | International Standard also uses the following terms defined in W3C XML Information Set:

a) abstract information item;

b) character information item;

c) element information item;

d) information item;

e) namespace information item;

f) property (of an information item).

3.1.4 This Recommendation | International Standard also uses the following terms defined in W3C SOAP Part 1, 1.5.1:

a) SOAP;

b) SOAP binding;

c) SOAP message exchange pattern (MEP);

d) SOAP node.

3.1.5 This Recommendation | International Standard also uses the following terms defined in ITU-T Rec. X.891 | ISO/IEC 24824-1:

a) Base64;

b) fast infoset document;

c) XML infoset.

3.2 Additional definitions

3.2.1 **ASN.1 SOAP interface binding**: A concrete interface of a service description (see 12.4) that specifies the semantics of a Fast Web Service that is to be provided through the exchange of ASN.1 SOAP messages.

3.2.2 **ASN.1 SOAP endpoint**: A network location of a Fast Web Service identified in a service description.

3.2.3 **ASN.1 SOAP header block**: A value of the **`HeaderBlock`** type (see Annex A).

3.2.4 **ASN.1 SOAP HTTP binding**: A binding of SOAP to HTTP for the transmission of ASN.1 SOAP messages.

3.2.5 **ASN.1 SOAP message**: A value of the **`Envelope`** type mapped from a W3C SOAP message (see clause 8).

3.2.6 **embedded ASN.1 encoded value**: An abstract value of an ASN.1 type, whose encoding is included in a W3C SOAP message as a Base64 string.

3.2.7 **embedded fast infoset document**: An element information item that, when included in an ASN.1 SOAP message, is encoded as a fast infoset document.

3.2.8 **fast-enabled web service client**: A SOAP node that may send requests and receive responses using both ASN.1 SOAP messages and XML SOAP messages.

3.2.9 **fast infoset SOAP message**: A W3C SOAP message serialized as a fast infoset document.

3.2.10 **fast web services**: Services provided by the exchange of ASN.1 SOAP messages.

3.2.11 **service description**: A set of documents that describe the interface to and the semantics of a Web service.

3.2.12 **W3C SOAP header block**: The "SOAP header block" defined in W3C SOAP Part 1, 1.5.2.

3.2.13 **W3C SOAP message**: The "SOAP message" defined in W3C SOAP Part 1, 1.5.2.

3.2.14 **W3C SOAP namespace**: The namespace whose name is "**http://www.w3.org/2003/05/soap-envelope**" (see W3C SOAP Part 1, 1.1).

3.2.15 **XML web services**: Services provided by the exchange of XML SOAP messages.

3.2.16 **XML SOAP message**: A W3C SOAP message, or a message defined by any previous or subsequent version of SOAP, serialized as an XML document.

4 Abbreviations

For the purposes of this Recommendation | International Standard, the following abbreviations apply:

AII	Attribute Information Item (see W3C XML Information Set, 2.3)
ASN.1	Abstract Syntax Notation One
CII	Character Information Item (see W3C XML Information Set, 2.6)
EII	Element Information Item (see W3C XML Information Set, 2.2)
HTTP	HyperText Transfer Protocol (see IETF RFC 2616)
MIME	Multipurpose Internet Mail Extensions
NII	Namespace Information Item (see W3C XML Information Set, 2.11)
PER	Packed Encoding Rules of ASN.1
RPC	Remote Procedure Call
URI	Uniform Resource Identifier
WSDL	Web Services Description Language
XML	eXtensible Markup Language
XSD	W3C XML Schema

5 Notation

5.1 This Recommendation | International Standard uses the ASN.1 notation defined by ITU-T Rec. X.680 | ISO/IEC 8824-1.

5.2 In this Recommendation | International Standard, **`bold Courier`** is used for ASN.1 notation.

5.3 For the following notations, **bold Arial** is used:

a) XML syntax;

b) the names of EIIs and AIIs;

c) HTTP header fields and parameters of HTTP header fields.

5.4 The names of information items' properties are in **bold Arial** and enclosed between square brackets (for example, **[children]** property).

5.5 MIME media types and URIs are in **bold Arial** and enclosed between normal quotes (for example, the URI "**http://www.w3.org/2003/05/soap-envelope**").

6 The processing of ASN.1 SOAP messages

6.1 ASN.1 SOAP messages are abstract values of the **`Envelope`** type defined in the ASN.1 module **`ASN1SOAP`** (see Annex A). The abstract values of the **`Envelope`** type are semantically equivalent to instances of the XML Infoset specified in W3C SOAP Part 1, clause 5 (referred to as the W3C SOAP message infoset).

NOTE – The **`Envelope`** type enables an optimal binary encoding of the W3C SOAP message infoset.

6.2 ASN.1 SOAP messages may be used either in conjunction with Web service descriptions or independently of any Web service description. A Web service description for XML SOAP messages requires no changes to provide a Fast Web Services description for ASN.1 SOAP messages (see Annex E).

6.3 The SOAP processing model, extensibility model, and binding model (see W3C SOAP Part 1, clauses 2, 3, and 4) shall be applied, by a SOAP node, to the abstract values of the **`Envelope`** type through the mapping specified in 6.4 between the components of the **`Envelope`** type and the information items of the W3C SOAP message infoset.

6.4 The application of these SOAP models to abstract values of the **`Envelope`** type shall be the result of the following conceptual steps:

a) the abstract values of the components of the **`Envelope`** type (an ASN.1 SOAP message) are mapped to information items of a W3C SOAP message infoset as specified in clause 7 and Table 1;

b) the SOAP models are applied to that infoset (see W3C SOAP Part 1, clauses 2, 3, and 4), usually producing a new W3C SOAP message infoset that conforms to W3C SOAP Part 5 and restricted as specified in 6.6; and

c) the information items of the new W3C SOAP message infoset are mapped back to abstract values of the components of the **`Envelope`** type as specified in clause 8 and Table 1, usually producing a new abstract value for the **`Envelope`** type (a new ASN.1 SOAP message).

NOTE – These three steps are only conceptual. There is no requirement for an implementation to actually generate a representation of a W3C SOAP message infoset. Both a W3C SOAP message infoset and an ASN.1 SOAP message are abstract values, independent of any serialization or encoding used for their representation in a computer system or for transfer between systems.

6.5 The application of the SOAP models to the W3C SOAP message infoset (see 6.4 b) shall include the extended processing of embedded ASN.1 encoded values as specified in clause 9.

6.6 The following restrictions apply to the W3C SOAP message infoset resulting from the transformation referred to in 6.4 b:

a) no AIIs shall be present among the members of the **[attributes]** property of the **Body** EII and **Detail** EII; and

b) at most one EII shall be present among the members of the **[children]** property of the **Body** EII and **Detail** EII.

6.7 A component of the **`Envelope`** type (at any depth up to the presence of a value of the **`Content`** type) shall be mapped to an information item (or conversely) as specified in Table 1. Column 1 of Table 1 lists the components of the **`Envelope`** type. Column 2 gives reference to the subclause of W3C SOAP Part 1 that specifies the semantically equivalent information item(s). Column 3 lists the clause and subclauses of this Recommendation | International Standard that specify the mapping from the component to the semantically equivalent information item(s). Column 4 lists the clause and subclauses of this Recommendation | International Standard that specify the mapping from the information item(s) to the component.

Table 1 – Mapping between components of the `Envelope` type and information items of a W3C SOAP message infoset

ASN.1 module for ASN.1 SOAP	W3C SOAP Part 1 reference	Mapping from ASN.1	Mapping to ASN.1
`Envelope ::= SEQUENCE {`	5.1	Clause 7	Clause 8
`header Header,`	5.2	Subclause 7.1.3	Subclause 8.1.2

`body-or-fault CHOICE {`
`body Body,`
`fault Fault`
	5.3, 5.4	Subclauses 7.1.4 & 7.1.5	Subclauses 8.1.3 & 8.1.4

`}` `}`			
`Header ::= SEQUENCE OF HeaderBlock`	5.2	Subclause 7.2	Subclause 8.2
`HeaderBlock ::= SEQUENCE {`	5.2.1	Subclause 7.2.2	Subclause 8.2.2
`mustUnderstand BOOLEAN OPTIONAL,`	5.2.2	Subclause 7.2.2.1	Subclause 8.2.2.1
`relay BOOLEAN OPTIONAL,`	5.2.3	Subclause 7.2.2.2	Subclause 8.2.2.2
`role XSD.AnyURI` `DEFAULT ultimateReceiver,`	5.2.4	Subclause 7.2.2.3	Subclause 8.2.2.3
`content Content`		Subclause 7.2	Subclause 8.2
`}`			
`Body ::= SEQUENCE {`	5.3	Subclause 7.3	Subclause 8.3
`content Content OPTIONAL`	5.3	Subclause 7.3.2	Subclause 8.3.2
`}`			
`Fault ::= SEQUENCE {`	5.4	Subclause 7.4	Subclause 8.4
`code Code,`	5.4.1	Subclause 7.4.1.2	Subclause 8.4.1.2
`reason SEQUENCE SIZE(1..MAX) OF Text,`	5.4.2	Subclause 7.4.1.3	Subclause 8.4.1.3
`node XSD.AnyURI OPTIONAL,`	5.4.3	Subclause 7.4.1.4	Subclause 8.4.1.4
`role XSD.AnyURI OPTIONAL,`	5.4.4	Subclause 7.4.1.5	Subclause 8.4.1.5
`detail Content`	5.4.5	Subclause 7.4.1.6	Subclause 8.4.1.6
`}`			
`Code ::= SEQUENCE {`	5.4.1	Subclause 7.4.2	Subclause 8.4.2
`value Value,`	5.4.1.1	Subclause 7.4.2.2	Subclause 8.4.2.2
`subcodes SEQUENCE OF XSD.QName`	5.4.1.2, 5.4.1.3	Subclauses 7.4.2.3 & 7.4.2.4	Subclauses 8.4.2.3 & 8.4.2.4
`}`			

`Value ::= ENUMERATED {`
`versionMismatch,`
`mustUnderstand,`
`dataEncodingUnknown,`
`sender,`
`receiver`
	5.4.1.1, 5.4.8	Subclause 7.4.3	Subclause 8.4.3

`}`			
`Text ::= SEQUENCE {`	5.4.2.1	Subclause 7.4.4	Subclause 8.4.4
`lang XSD.Language,`		Subclause 7.4.4.2	Subclause 8.4.4.2
`text UTF8String`		Subclause 7.4.4.3	Subclause 8.4.4.3
`}`			
`Content ::= ....`	N/A	Subclause 7.5	Subclause 8.5

7 Mapping components of the `Envelope` type to information items

7.1 General

7.1.1 An **Envelope** EII shall be generated from a value of the `Envelope` type.

7.1.2 A unique **[prefix]** property of an NII with a **[namespace name]** property equal to the name of the W3C SOAP namespace among the members of the **[in-scope namespaces]** property of the **Envelope** EII shall be generated with its value chosen by the SOAP node.

NOTE 1 – The prefix "**env**" is conventionally used in W3C SOAP Part 1, 1.1, but any prefix can be used.

NOTE 2 – All EIIs and AIIs defined in SOAP have a **[namespace name]** property equal to the name of the W3C SOAP namespace as specified in W3C SOAP 1, 1.1.

7.1.3 A value of the `header` component shall be mapped as specified in 7.2.

7.1.4 If a value of the `body-or-fault` component has the `body` alternative present, then that alternative shall be mapped to a **Body** EII as specified in 7.3.

7.1.5 If a value of the `body-or-fault` component has the `fault` alternative present, then a `Body` EII shall be generated and the alternative shall be mapped to a `Fault` EII as specified in 7.4.

NOTE – A W3C SOAP message containing fault information may only have one **Fault** EII as a child of the **Body** EII (and can have no other child EIIs). The ASN.1 schema reflects these constraints by providing separate body and fault alternatives of the `body-or-fault` choice.

7.2 Mapping of the `Header` type

7.2.1 A **Header** EII shall be generated from a value of the `Header` type. If the `Header` type contains one or more occurrences of `HeaderBlock`, then each occurrence of `HeaderBlock` shall be mapped, in order, to a child EII of the **Header** EII as specified in 7.2.2. If there are no occurrences of `HeaderBlock`, then no **Header** EII shall be generated.

7.2.2 A value of the `content` component shall be mapped to a W3C SOAP header block as specified in 7.5. Additional AIIs, among the members of the **[attributes]** property of the EII generated in 7.5, shall be generated as specified in 7.2.2.1 to 7.2.2.3.

7.2.2.1 The **mustUnderstand** AII shall be generated from a value of the `mustUnderstand` component if the value is present and is not `FALSE`, and the **[normalized value]** property of the **mustUnderstand** AII shall be "**1**". Otherwise no **mustUnderstand** AII shall be generated.

7.2.2.2 The **relay** AII shall be generated from a value of the `relay` component if the value is present and is not `FALSE`, and the **[normalized value]** property of the **relay** AII shall be "**1**". Otherwise no **relay** AII shall be generated.

7.2.2.3 The **role** AII shall be generated from a value of the `role` component if that value is different from `ultimateReceiver`, and the **[normalized value]** property of the **role** AII shall be the character string value of the `role` component. Otherwise no role AII shall be generated.

7.3 Mapping of the `Body` type

7.3.1 A **Body** EII shall be generated from a value of the `Body` type.

7.3.2 The value of the `content` component (if present) shall be mapped as specified in 7.5.

7.4 Mapping of the `Fault` type

7.4.1 General

7.4.1.1 A **Fault** EII shall be generated from a value of the `Fault` type.

7.4.1.2 A value of the `code` component shall be mapped as specified in 7.4.2.

7.4.1.3 The **Reason** EII shall be generated from a value of the `reason` component. Each occurrence of `Text` in the sequence-of shall be mapped, in order, to a child **Text** EII of the **Reason** EII as specified in 7.4.4.

NOTE – It is recommended that all occurrences of `Text` in the sequence-of have unique **lang** component values (see W3C SOAP Part 1, 5.4.2).

7.4.1.4 The **Node** EII shall be generated from a value of the `node` component (if present), and the **Node** EII shall have as its **child** CIIs the characters of the character string value of the `node` component.

7.4.1.5 The **Role** EII shall be generated from a value of the **role** component (if present), and the **Role** EII shall have as its child CIIs the characters of the character string value of the **role** component.

7.4.1.6 The **Detail** EII shall be generated from the **detail** component (if present) as specified in 7.5.

7.4.2 Mapping of the Code type

7.4.2.1 A **Code** EII shall be generated from a value of the **Code** type.

7.4.2.2 A value of the **value** component shall be mapped as specified in 7.4.3 to provide the first (or the only, if the **subcodes** component is empty) child EII of the **Code** EII.

7.4.2.3 The first **XSD.QName** (if any) of the **subcodes** component shall generate:

a) a **Subcode** EII as the second child EII of the **Code** EII; and

b) a **Value** EII (child of a Subcode EII and generated from the value of the first occurrence of **XSD.QName** as specified in 7.4.2.5 to 7.4.2.6) as the first child EII of the **Subcode** EII generated in a).

7.4.2.4 Each of the following **XSD.QName**s (if any) of the **subcodes** component shall generate:

a) a **Subcode** EII as the second child of the **Subcode** EII that was generated from the value of the previous occurrence of **XSD.QName**; and

b) a **Value** EII (child of a **Subcode** EII and generated from the value of the current occurrence of **XSD.QName** as specified in 7.4.2.5 to 7.4.2.6) as the first child EII of the **Subcode** EII generated in a).

NOTE – Each **Subcode** EII has a second (**Subcode**) child EII if and only if there is a subsequent XSD.QName in the **subcodes**.

7.4.2.5 A **Value** EII (child of a **Subcode** EII) shall be generated from an occurrence of **XSD.QName** (with its **uri** component present) with:

a) an NII among the members of its **[in-scope namespaces]** property with a **[namespace name]** property equal to the value of the **uri** component and a **[prefix]** property chosen by the SOAP node; and

b) a sequence of CIIs that shall be the concatenation of:

1) the **[prefix]** property in a);

2) a COLON (":"), and

3) the character string value of the **name** component.

7.4.2.6 A **Value** EII (child of a **Subcode** EII) shall be generated from the value of an occurrence of **XSD.QName** (with its **uri** component absent) with a sequence of child CIIs that is the value of the **name** component.

7.4.3 Mapping of the Value type

A **Value** EII (child of a **Code** EII) shall be generated from a value of the **Value** type with a sequence of CIIs that shall be generated from the value of the enumeration as the characters of the character string that is the concatenation of:

a) the **[prefix]** property as specified in 7.1.2;

b) a COLON (":"); and

c) a local name as specified in Table 2.

Table 2 – Mapping the Value type to a local name

Enumeration value of Value	Local name
versionMismatch	VersionMismatch
mustUnderstand	MustUnderstand
dataEncodingUnknown	DataEncodingUnknown
sender	Sender
receiver	Receiver

7.4.4 Mapping of the Text type

7.4.4.1 A **Text** EII shall be generated from a value of the **Text** type.

7.4.4.2 An AII shall be generated from the **lang** component with:

a) a **[local name]** property of "**lang**"; and

b) a **[namespace name]** property of "**http://www.w3.org/XML/1998/namespace**"; and

c) a **[prefix]** property of "**xml**"; and

d) a **[normalized value]** property equal to the value of the **lang** component.

7.4.4.3 The sequence of **child** CIIs of the **Text** EII shall be the character string value of the **text** component.

7.5 Mapping of the Content type

7.5.1 General

7.5.1.1 A content EII shall be generated from a value of the **Content** type in 7.5.2, 7.5.3 or 7.5.4 for the mapping of fast infoset documents, ASN.1 encoded values and "not understood" ASN.1 SOAP header blocks (see 7.5.4), respectively, to XML infoset.

7.5.1.2 If the **fast-infoset-document** alternative of the **Content** type is present, 7.5.2 shall apply.

7.5.1.3 If the **encoded-value** alternative of the **Content** type is present and the **encoded-value.id** is not equal to the **notUnderstoodIdentifier** value, 7.5.3 shall apply.

7.5.1.4 If the **encoded-value** alternative of the **Content** type is present and the **encoded-value.id** is equal to the **notUnderstoodIdentifier** value, 7.5.4 shall apply.

7.5.2 Fast infoset document content

7.5.2.1 The octets of the **fast-infoset-document** component will be a fast infoset document specified in ITU-T Rec. X.891 | ISO/IEC 24824-1.

7.5.2.2 The content EII shall be generated by application of the following:

a) decode the octets of the **fast-infoset-content** to generate an XML infoset that is a root EII (as specified in ITU-T Rec. X.891 | ISO/IEC 24824-1); and

b) apply 7.5.2.3 to the root EII to generate the content EII.

7.5.2.3 The following AIIs (among the members of the **[attributes]** property of the root EII), if generated from the mapping of the fast infoset value to a root EII, shall be removed from the **[attributes]** property of the root EII:

a) the **role** AII;

b) the **mustUnderstand** AII;

c) the **relay** AII.

NOTE – The **role**, **mustUnderstand** and **relay** AIIs are mapped from **role**, **mustUnderstand** and **relay** components of the **HeaderBlock** type respectively (see 7.2.2). The removal of these AIIs from the **[attributes]** property of the root EII ensures that only components of the **HeaderBlock** will be used for the processing of W3C SOAP header block by a SOAP node.

7.5.3 Embedded ASN.1 encoded value content

7.5.3.1 An **encodingStyle** AII (see W3C SOAP Part 1, 5.1.1), among the members of the **[attributes]** property of the content EII, shall be generated with a **[normalized value]** property of:

"**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope:encoding-style:aper**".

7.5.3.2 If the **encoded-value.id** component has the **qName** alternative present, then the **[namespace name]** and **[local name]** properties of the content EII shall be set from the **qName**.

7.5.3.3 If the **encoded-value.id** component has the **roid** alternative present, then the content EII shall be generated with:

a) a **[local name]** property of "roid";

b) a **[namespace name]** property of:
"**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope**";

c) a **roid** AII among the members of the **[attributes]** property as specified in 7.5.3.4.

7.5.3.4 An AII, among the members of the **[attributes]** property of the content EII, shall be generated from a value of the **Content** type (if the **encoded-value.id** component has the **roid** alternative present) with:

a) a **[local name]** property of "roid";

b) a **[namespace name]** property of:
"**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope**";

c) a **[specified]** property of "true";

d) a **[normalized value]** property that shall be the value of the **`roid`** component encoded as an "XMLRelativeOIDValue" using only the "XMLNumberForm" (see ITU-T Rec. X.680 | ISO/IEC 8824-1, clause 32).

7.5.3.5 A sequence of child CIIs of the content EII shall be generated from the Base64 encoding of an octet string (as specified in IETF RFC 2045, 6.8) that is the value of the **`encoded-value.encoding`** component.

7.5.3.6 The **`schema-identifier`** component, if present, shall be ignored and not mapped.

7.5.4 Not understood W3C SOAP header block content

7.5.4.1 The **`notUnderstoodIdentifier`** will identify the ASN.1 type **`NotUnderstood`**, a value of which is encoded, using Basic Aligned PER, to an octet string that is the value of the **`encoded-value.encoding`** component.

7.5.4.2 A value of the **`NotUnderstood`** type shall be generated by decoding, using Basic Aligned PER, the octets of **`encoded-value.encoding`** component.

7.5.4.3 A **NotUnderstood** EII (see W3C SOAP Part 1, 5.4.8.1) shall be generated as the content EII with:

a) a NII among the members of its **[in-scope namespaces]** property with a **[namespace name]** property equal to the value of the **`NotUnderstood.uri`** component and a unique **[prefix]** property chosen by the SOAP node; and

b) a **qname** AII (see W3C SOAP Part 1, 5.4.8.2) with a **[normalized value]** property that is the concatenation of the **[prefix]** property in a), a COLON (":"), and the character string value of the **`NotUnderstood.name`** component.

8 Mapping W3C SOAP message infosets to abstract values of the `Envelope` type

8.1 General

8.1.1 A value of the **`Envelope`** type shall be generated from an **Envelope** EII.

8.1.2 The **Header** EII (if present) shall be mapped to the **`header`** component as specified in 8.2.

8.1.3 If a **Body** EII does not contain a **Fault** EII as the only child EII, then a value of the **`body-or-fault`** component with the **`body`** alternative present shall be generated and the **Body** EII shall be mapped to the **`body`** alternative as specified in 8.3.

8.1.4 If a **Body** EII contains a **Fault** EII as the only child EII, then a value of the **`body-or-fault`** component with the **`fault`** alternative present shall be generated and the **Fault** EII shall be mapped to the **`fault`** alternative as specified in 8.4.

8.2 Mapping of a Header EII

8.2.1 A value of the **`Header`** type shall be generated from a **Header** EII, and each child EII (a W3C SOAP header block) shall be mapped, in order, to an occurrence of **`Content`** in the sequence-of as specified in 8.2.2.

8.2.2 A value of the **`HeaderBlock`** type shall be generated from a W3C SOAP header block, and the W3C SOAP header block shall be mapped to a value of the content component as specified in 8.5. Additional components of the **`HeaderBlock`** type shall be generated as specified in 8.2.2.1 to 8.2.2.3.

8.2.2.1 A value of the **`mustUnderstand`** component shall be generated from the **mustUnderstand** AII (if present), and shall be **`TRUE`** if the **[normalized value]** property of the **mustUnderstand** AII is "**1**". Otherwise, the component shall be absent.

8.2.2.2 A value of the **`relay`** component shall be generated from the **relay** AII (if present), and shall be **`TRUE`** if the **[normalized value]** property of the **relay** AII is "**1**". Otherwise, the component shall be absent.

8.2.2.3 A value of the **`role`** component shall be generated from the **role** AII (if present), and shall be the **[normalized value]** property of the **role** AII.

8.3 Mapping of a Body EII

8.3.1 A value of the **Body** type shall be generated from a **Body** EII.

8.3.2 The child EII of the **Body** EII (if present) shall be mapped to a value of the **content** component as specified in 8.5.

8.4 Mapping of a Fault EII

8.4.1 General

8.4.1.1 A value of the **Fault** type shall be generated from a **Fault** EII.

8.4.1.2 A **Code** EII shall be mapped to a value of the **code** component as specified in 8.4.2.

8.4.1.3 A value of the **reason** component shall be generated from the **Reason** EII. Each child **Text** EII shall be mapped, in order, to an occurrence of **Text** in the sequence-of as specified in 8.4.4.

8.4.1.4 A value of the **node** component shall be generated from the **Node** EII (if present), and shall have as its character string value the sequence of child CIIs of the **Node** EII.

8.4.1.5 A value of the **role** component shall be generated from the **Role** EII (if present), and shall have as its character string value the sequence of child CIIs of the **Role** EII.

8.4.1.6 A value of the **detail** component shall be generated from the **Detail** EII (if present), and the child EII shall be mapped as specified in 8.5.

8.4.2 Mapping of a Code EII

8.4.2.1 A value of the **Code** type shall be generated from a **Code** EII.

8.4.2.2 The **Value** EII (child of a **Code** EII) shall be mapped to a value of the **value** component as specified in 8.4.3.

8.4.2.3 The first **Subcode** EII (if present) shall generate a value of the **XSD.QName** type as the first item of the **subcodes** component. The value shall be generated from the first child **Value** EII as specified in 8.4.2.5 and 8.4.2.6.

8.4.2.4 The second child **Subcode** EII (if present) of each **Subcode** EII shall generate a value of the **XSD.QName** type as the next item of the **subcodes** component. The value shall be generated from the first child **Value** EII of the second child **Subcode** EII as specified in 8.4.2.5 or in 8.4.2.6.

8.4.2.5 A value of the **XSD.QName** type shall be generated from a **Value** EII (child of a **Subcode** EII), having a sequence of child CIIs that is the concatenation of a prefix (P, say), a COLON (":"), and a local name, with:

a) a value of the **uri** component that is the **[namespace name]** property of the NII, among the members of the **[in-scope namespaces]** property of the **Value** EII (child of a Subcode EII) with **[prefix]** property P;

b) a value of the **name** component that is the local name.

8.4.2.6 A value of the **XSD.QName** type shall be generated from a **Value** EII (child of a **Subcode** EII) having a sequence of child CIIs that does not contain a COLON (":"), with a value of the **name** component that is the character string value of the sequence of child CIIs.

8.4.3 Mapping of a Value EII that is a child of a Code EII

A value of the **Value** type shall be generated from a **Value** EII (child of a **Code** EII) with a local name, as specified in Table 2, and shall be a substring of the sequence of child CIIs that is a concatenation of the following:

a) the **[prefix]** property as specified in 7.1.2;

b) a COLON (":"); and

c) the local name.

8.4.4 Mapping of a Text EII

8.4.4.1 A value of the **Text** type shall be generated from a **Text** EII.

8.4.4.2 A value of the **lang** component shall be generated from an AII with a **[local name]** property of "**lang**" and a **[namespace name]** property of "**http://www.w3.org/XML/1998/namespace**", and shall be the **[normalized value]** property of the AII.

8.4.4.3 A value of the **text** component shall be generated from the **Text** EII and shall be the sequence of the child CIIs of the **Text** EII.

8.5 Mapping of a content EII to a value of the Content type

8.5.1 General

8.5.1.1 A value of the **Content** type shall be generated from a content EII as specified in 8.5.2, 8.5.3 or 8.5.4 for the mapping from XML infoset to fast infoset documents, ASN.1 encoded values and not understood ASN.1 SOAP header blocks, respectively.

8.5.1.2 Subclause 8.5.2 shall apply if:

a) the **encodingStyle** AII (see W3C SOAP Part 1, 5.1.1) is not among the members of the **[attributes]** property of the content EII and the content EII is not a **NotUnderstood** EII (see W3C SOAP Part 1, 5.4.8.1); or

b) the **encodingStyle** AII is among the members of the **[attributes]** property of the content EII and the **encodingStyle** AII has a **[normalized value]** property that is not equal to "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope:encoding-style:aper**" (see 7.5.3.1).

8.5.1.3 If the **encodingStyle** AII (see W3C SOAP Part 1, 5.1.1) is among the members of the **[attributes]** property of the content EII and the **encodingStyle** AII has a **[normalized value]** property of "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope:encoding-style:aper**", 8.5.3 shall apply.

8.5.1.4 If the content EII is a **NotUnderstood** EII (see W3C SOAP Part 1, 5.4.8.1), 8.5.4, shall apply.

NOTE – An **encodingStyle** AII cannot be among the members of the **[attributes]** property of a **NotUndestood** EII (see W3C SOAP Part 1, 5.4.8.1).

8.5.2 Embedded fast infoset document

8.5.2.1 A value of the **Content** type with the **fast-infoset-document** alternative present shall be generated.

8.5.2.2 The octets of the **fast-infoset-document** component will be a fast infoset document and shall be generated by application of the following:

a) apply 8.5.2.3 to the content EII to generate the root EII of an XML infoset;

b) encode the XML infoset as a fast infoset document (as specified in ITU-T Rec. X.891 | ISO/IEC 24824-1).

8.5.2.3 The following AIIs, if present among the members of the **[attributes]** property of the content EII, shall be removed from the **[children]** property of the EII:

a) the **role** AII;

b) the **mustUnderstand** AII;

c) the **relay** AII.

NOTE – The removal of these AIIs from the **[attributes]** property of the content EII ensures that only components of the **HeaderBlock** will used for the processing of W3C SOAP header block by a SOAP node.

8.5.3 Embedded ASN.1 encoded value

8.5.3.1 A value of the **Content** type with the **encoded-value** alternative present shall be generated.

8.5.3.2 If the **roid** AII (see 7.5.3.4) is not among the members of the **[attributes]** property of the content EII, then:

a) an **encoded-value.id** with the **qName** alternative present shall be generated; and

b) its value shall be set from the **[local name]** property and **[namespace name]** property of the content EII.

8.5.3.3 If **roid** AII (see 7.5.3.4) is among the members of the **[attributes]** property of the content EII**,** then:

a) an **encoded-value.id** with the **roid** alternative present shall be generated; and

b) its value shall be set from the **[normalized value]** property of the **roid** AII encoded as an "XMLRelativeOIDValue", using only the "XMLNumberForm" (see ITU-T Rec. X.680 | ISO/IEC 8824-1, clause 32).

NOTE – A relative object identifier, instead of a qualified name, may be used when there are constraints on the size of ASN.1 SOAP messages.

8.5.3.4 A value of the **encoded-value.encoding** component shall be generated from the sequence of child CIIs of the content EII that is the Base64 encoding of an octet string, and shall be the octet string.

8.5.3.5 The **schema-identifier** component shall not be mapped and is omitted.

8.5.4 Not understood W3C SOAP header block content

8.5.4.1 An **encoded-value.id** with the **qName** alternative present shall be generated; and its value shall be set from the **[local name]** property and **[namespace name]** property of the **notUnderstood** EII.

8.5.4.2 A value of the **NotUnderstood** type shall be generated from the **NotUnderstood** EII, with a **[normalized value]** property of the **qname** AII that is the concatenation of a prefix (P, say), a COLON (":"), and a local name, with:

a) a value of the **uri** component that is the **[namespace name]** property of the NII, among the members of the **[in-scope namespaces]** property of the **NotUnderstood** EII, with the **[prefix]** property P; and

b) a value of the **name** component that is the local name.

8.5.4.3 The value of the **NotUnderstood** type shall be encoded, using Basic Aligned PER, to an octet string that shall be the value of the **encoded-value.encoding** component.

9 Extended SOAP processing of embedded ASN.1 encoded values

9.1 General

9.1.1 The extended processing specified in the following subclauses extends the processing of W3C SOAP messages, specified in W3C SOAP Part 1, to allow additional transformations by a SOAP node of content EIIs that have been mapped from ASN.1 SOAP messages.

NOTE – The extended processing is required because a content EII will contain, as a sequence of child CIIs, an embedded ASN.1 encoded value, which is opaque to the SOAP node unless further processing is performed to generate an ASN.1 value from the sequence of child CIIs.

9.1.2 The content EII shall be a child EII of a **Body** EII, a **Header** EII (W3C SOAP header blocks), a **Detail** EII.

NOTE – The content EII is normally processed as follows:

a) an ultimate SOAP receiver processes the child of a Body EII or the child of a Detail EII and any targeted W3C SOAP header blocks;

b) a SOAP intermediary processes any targeted W3C SOAP header blocks;

c) a SOAP node processes targeted W3C SOAP header blocks as a consequence of a) or b); and

d) a SOAP node (such as an active intermediary) processes information items by additional processing not described by W3C SOAP header blocks.

9.1.3 The content (child) EII will have among the members of its **[attributes]** property an **encodingStyle** AII with a **[normalized value]** property of

"urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope:encoding-style:aper"

as specified in 7.5.3.1.

9.1.4 The application of the extended processing to a content EII shall be the result of the following conceptual steps:

a) the ASN.1 type of the embedded ASN.1 encoded value is identified as specified in 9.2;

b) the ASN.1 value is generated from the identified embedded ASN.1 encoded value, given the ASN.1 type as specified in 9.3;

c) the generated ASN.1 value is processed by the SOAP node, usually producing one or more resulting ASN.1 values with identifiers; and

d) the resulting ASN.1 values with identifiers are inserted into the new W3C SOAP message infoset (see 6.4 b) as embedded ASN.1 encoded values as specified in 9.4.

NOTE – These four steps are only conceptual. There is no requirement for an implementation to actually generate an ASN.1 value from an identified embedded ASN.1 value since there is no requirement for an implementation to generate a representation of a W3C SOAP message infoset.

9.2 Identifying the ASN.1 type of an embedded ASN.1 encoded value

9.2.1 For the purposes of identification, a value of the **Identifier** type shall be generated from the content EII in 9.2.2 and 9.2.3, and the value shall identify the ASN.1 type of an embedded ASN.1 encoded value.

9.2.1.1 The processing SOAP node shall fault, as specified in 9.5, if the ASN.1 type cannot be identified from the value of the **`Identifier`**.

9.2.1.2 The means by which the processing SOAP node obtains, and manages, the set of **`Identifier`** values and identified ASN.1 types is not specified in this Recommendation | International Standard.

NOTE – A SOAP node may obtain a (partial) set of identified ASN.1 types from a service description (see 13.8).

9.2.2 If the content EII has a **[local name]** property of "roid" and a **[namespace name]** property of "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope**", and the **roid** AII (see 7.5.3.4) is among the members of the **[attributes]** property, then:

a) a value of the **`Identifier`** type with the **`roid`** alternative present shall be generated; and

b) its value shall be set from the **[normalized value]** property of the **roid** AII that is encoded as a "XMLRelativeOIDValue" using only the "XMLNumberForm" (see ITU-T Rec. X.680 | ISO/IEC 8824-1, clause 32).

9.2.3 If the **roid** AII (see 7.5.3.4) is not among the members of the **[attributes]** property of the content EII, then:

a) a value of the **`Identifier`** type with the **`qName`** alternative present shall be generated; and

b) its value shall be set from the **[local name]** and **[namespace name]** properties of the content EII.

9.3 Generating an ASN.1 value from an identified embedded ASN.1 encoded value

An ASN.1 value shall be generated from the child CIIs of the content EII (the embedded ASN.1 encoded value). These child CIIs are the Base64 encoding of the octet string (as specified in IETF RFC 2045, 6.8) consisting of the Basic Aligned PER encoding of the ASN.1 value whose ASN.1 type has been identified, as specified in 9.2.

9.4 Insertion of an ASN.1 value (with an identifier) into a W3C SOAP message

9.4.1 General

9.4.1.1 An ASN.1 value with an identifier that is a value of the **`Identifier`** type, and possibly additional values, shall be inserted as an embedded ASN.1 encoded value of a generated content EII into a W3C SOAP message as specified in the following subclauses.

NOTE – Values of the **`Identifier`** type and additional values may be obtained from a service description or supplied by an application of a SOAP node.

9.4.1.2 If the ASN.1 value to be inserted as an embedded ASN.1 encoded value of a generated content EII is the child of the **Header** EII (a W3C SOAP header block), 9.4.2 shall apply.

9.4.1.3 If the ASN.1 value to be inserted as an embedded ASN.1 encoded value of a generated content EII is the child of the **Body** EII, 9.4.3 shall apply.

9.4.1.4 If the ASN.1 value to be inserted as an embedded ASN.1 encoded value of a generated content EII is the child of the **Detail** EII, 9.4.4 shall apply.

9.4.2 Insertion as a child EII of the Header EII

9.4.2.1 A content EII shall be generated from the ASN.1 value and **`Identifier`** value as specified in 9.4.5. The content EII shall be a W3C SOAP header block that is a child of the **Header** EII. Additional values (if present) shall result in the insertion of AIIs among the members of the **[attributes]** property of the content EII as specified in the three following subclauses.

NOTE – The order in which SOAP header block EIIs are inserted is dependent on the processing SOAP node.

9.4.2.2 An additional URI (if present) corresponding to the semantics of a **role** AII shall generate the **role** AII and its **[normalized value]** property shall be the character string value of the URI.

9.4.2.3 An additional boolean value (if present) corresponding to the semantics of the **mustUnderstand** AII shall generate the **mustUnderstand** AII if the boolean value is **`TRUE`**, and its **[normalized value]** property shall be "**1**".

9.4.2.4 An additional boolean value (if present) corresponding to the semantics of the **relay** AII shall generate the **relay** AII if the boolean value is **`TRUE`**, and its **[normalized value]** property shall be "**1**".

9.4.3 Insertion as child EII of the Body EII

A content EII shall be generated from the ASN.1 value and **`Identifier`** value as specified in 9.4.5. The content EII shall be the only child of the **Body** EII and shall replace any existing child EII of the **Body** EII.

9.4.4 Insertion as a child EII of the Detail EII

A content EII shall be generated from the ASN.1 value and **`Identifier`** value as specified in 9.4.5. The content EII shall be the only child of the **Detail** EII and shall replace any existing child EII of the **Detail** EII.

9.4.5 Generation of a content EII from an ASN.1 value and `Identifier` value

9.4.5.1 A content EII shall be generated from the ASN.1 value (to be inserted as an embedded ASN.1 encoded value) and **`Identifier`** value as specified in the four following subclauses.

9.4.5.2 If the value of the **`Identifier`** type has the **`qName`** alternative present, then the **[namespace name]** and **[local name]** properties of the content EII shall be set from the **`qName`**.

9.4.5.3 If the value of the **`Identifier`** type has the **`roid`** alternative present, then the **[local name]** property of the content EII shall be "roid" and the **[namespace name]** property shall be "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope**". An AII, among the members of the **[attributes]** property of the content EII, shall be generated with:

a) a **[local name]** property of "roid";

b) a **[namespace name]** property of "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope**";

c) a **[specified]** property of "true"; and

d) a **[normalized value]** property that shall be the value of the **`roid`** component encoded as an "XMLRelativeOIDValue" using only the "XMLNumberForm" (see ITU-T Rec. X.680 | ISO/IEC 8824-1, clause 32).

9.4.5.4 A sequence of child CIIs (the embedded ASN.1 encoded value) of the content EII shall be generated from the Base64 encoding of the octet string (as specified in IETF RFC 2045, 6.8) consisting of the Basic Aligned PER encoding of the ASN.1 value.

9.4.5.5 An **encodingStyle** AII (see W3C SOAP Part 1, 5.1.1), among the members of the **[attributes]** property of the EII, shall be generated with a **[normalized value]** property of

"**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope:encoding-style:aper**".

9.5 The "ASN.1 type not identifiable" fault

9.5.1 A SOAP node shall fault if the SOAP node cannot identify the ASN.1 type of the embedded ASN.1 encoded value from a value of the **`Identifier`** type generated in 9.2. The following subclauses specify the infoset for fault information of a W3C SOAP message infoset that is a fault.

9.5.2 A **Value** EII (child of a **Code** EII) shall be generated with a sequence of child CIIs that is a concatenation of the following character strings:

a) the **[prefix]** property as specified in 7.1.2; and

b) a COLON (":"); and

c) the string "Sender".

9.5.3 A unique **[prefix]** property of an NII with a **[namespace name]** property of "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope**" among the members of the **[in-scope namespaces]** property of the **Envelope** EII (or a child EII at any depth up to and including the **Value** EII generated in 9.5.4) shall be generated with its value chosen by the SOAP node.

9.5.4 A **Subcode** EII (child of a **Code** EII) shall be generated with a single child **Value** EII.

9.5.4.1 The single child **Value** EII shall have a sequence of child CIIs that is a concatenation of the following character strings:

a) the **[prefix]** property as specified in 9.5.3;

b) a COLON (":"); and

c) the string "NotIdentified".

NOTE – Such a not identifiable fault (a W3C SOAP message) serialized as an XML document is presented as follows:

```
<env:Envelope xmlns:env=" http://www.w3.org/2003/05/soap-envelope ">
    <env:Body>
        <env:Fault>
          <env:Code>
                <env:Value>env:Sender</env:Value>
```

```
                                        <env:Subcode>
                                                <env:Value
xmlns:fws="urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope">fws:NotIdentified</env:Value>
                                        </env:Subcode>
                                    </env:Code>
                                </env:Fault>
                            </env:Body>
                        </env:Envelope>
```

10 ASN.1 SOAP HTTP Binding

The ASN.1 SOAP HTTP Binding is a modification and extension of the SOAP HTTP Binding (see W3C SOAP Part 2, clause 7) and provides a binding of W3C SOAP to HTTP for the transmission of ASN.1 SOAP messages encoded using Basic Aligned PER. The ASN.1 SOAP HTTP Binding conforms to the SOAP protocol binding framework (see W3C SOAP Part 1, clause 4). Annex D provides tutorial material on the interoperation of Fast Web Services and XML Web services using features of the ASN.1 SOAP HTTP Binding.

10.1 HTTP media type

10.1.1 For the purposes of implementation of this Recommendation | International Standard, W3C SOAP Part 2, 7.1.4, shall be modified with the three following subclauses.

10.1.2 An implementation of an ASN.1 SOAP HTTP Binding shall be capable of sending and receiving ASN.1 SOAP messages using the "**application/fastsoap**" media type whose proper use and parameters are specified in B.1.

10.1.3 An implementation may also send requests and responses using other media types providing that such media types identify W3C SOAP messages.

NOTE – Such W3C SOAP messages may be, *inter alia*, XML SOAP messages, or fast infoset SOAP messages.

10.1.4 A binding may, when sending requests, provide an HTTP header field **Accept** (see IETF RFC 2616, 14.1). This header:

a) shall specify an ability to accept at least the "**application/fastsoap**" media type; and

b) may additionally indicate a willingness to accept other media types for the transfer of W3C SOAP messages.

NOTE – An implementation may send an XML SOAP message request with an HTTP header field **Accept** of "**Accept: application/fastsoap, application/soap+xml**" to specify the ability to accept ASN.1 SOAP messages in addition to indicate a willingness to accept XML SOAP messages (see the pessimistic strategy described in D.2).

10.2 Behavior of responding SOAP nodes

10.2.1 W3C SOAP Part 2, 7.5.2, shall be extended with the two following subclauses.

10.2.2 A responding SOAP node that receives an HTTP header field **Accept** with one or more additional media types of equal preference to the "**application/fastsoap**" media type shall interpret the "**application/fastsoap**" media type as having the highest preference (see IETF RFC 2616, 14.1), and shall respond with that media type.

10.2.3 A responding SOAP node shall add the HTTP header field **Fast-Enabled**, with an empty field value, to indicate Fast-enabled Web service support if the responding node cannot ascertain from the HTTP request that the sender can process content identified by the "**application/fastsoap**" media type.

11 Fast infoset SOAP messages and the SOAP HTTP Binding

This clause specifies a SOAP HTTP Binding (see W3C SOAP Part 2, clause 7) that supports the transmission of fast infoset SOAP messages over HTTP (called the fast infoset SOAP HTTP Binding).

11.1 An implementation of a fast infoset SOAP HTTP Binding shall be capable of sending and receiving fast infoset SOAP messages using the "**application/soap+fastinfoset**" media type whose proper use and parameters are specified in B.2.

NOTE – The SOAP HTTP Binding supports the sending of requests and responses using other media types if such media types specify the transfer of W3C SOAP message infosets.

11.2 A binding may, when sending requests, provide an HTTP header field **Accept** (see IETF RFC 2616, 14.1). This header:

a) shall specify an ability to accept at least the "**application/soap+fastinfoset**" media type.

b) may additionally indicate a willingness to accept other media types for the transfer of W3C SOAP messages.

NOTE – An implementation may send an XML SOAP message request with an HTTP header field **Accept** of "**Accept: application/soap+fastinfoset, application/soap+xml**" to specify the ability to accept fast infoset SOAP messages in addition to indicate a willingness to accept XML SOAP messages.

12 SOAP-oriented service descriptions supporting the ASN.1 SOAP interface binding

12.1 General

12.1.1 This clause specifies SOAP-oriented service descriptions that support the ASN.1 SOAP interface binding (see 12.4.7).

12.1.2 A SOAP-oriented service description is a set of documents specifying the interfaces and the semantics of a Web service that is to be provided through the exchange of SOAP messages.

12.1.3 A SOAP-oriented service description shall meet the requirements specified in this clause, but otherwise, there are no restrictions on the form of those documents or on the language (either natural or formal) in which they are written.

NOTE – Annex E describes the use of WSDL 1.1 [2] as a language for writing SOAP-oriented service descriptions.

12.1.4 A SOAP-oriented service description shall specify:

a) a set of schemas (see 12.2);

b) a set of abstract interfaces, each containing a set of abstract operations (see 12.3); and

c) a set of interface bindings (set of ASN.1 SOAP interface bindings), each containing a set of operation bindings (see 12.4);

12.2 Schemas

12.2.1 The SOAP-oriented service description for a given Web service may include the definition of one or more types of content data that are to be carried in SOAP messages in the provision of that Web service. This includes content data to be carried in the body of the message, in header blocks, and in faults.

12.2.2 The content data (if any) shall be defined by one or more XSD schemas. Each schema shall either be imported by providing its namespace URI, or embedded in the service description.

NOTE – The "XSD schema" here is not a "schema document", but an abstract schema (a set of schema components – see W3C XML schema) whose XML representation consists of one or more "xsd:schema" element information items. Usually, a schema is embedded in a service description by including its XML representation.

12.2.3 The set of all XSD schemas imported or embedded in a service description is called the original schema set.

12.2.4 Each type of content data in the original schema set may be specified either by a top-level **element declaration** schema component or a top-level **complex type definition** or **simple type definition** schema component.

12.3 Abstract interfaces and abstract operations

12.3.1 An abstract interface is specified by providing the information for a set of abstract operations and implicitly contains the following schema information (deriving from other information in the service description):

a) the RPC schema (see 12.3.2); and

b) the ASN.1 schema set (see 12.3.3).

12.3.2 The RPC schema is a specially-constructed XSD schema supporting RPC-style concrete interfaces, and shall be generated as specified in 12.5. The original schema set with the RPC schema added is called the complete schema set of the abstract interface.

12.3.3 The ASN.1 schema set is the ASN.1 mapping of the complete schema set of the abstract interface. Each XSD schema in the complete schema set shall be independently mapped to ASN.1 as specified in ITU-T Rec. X.694 | ISO/IEC 8825-5. The ASN.1 modules generated by the X.694 mapping form the ASN.1 schema set of the abstract interface.

12.3.4 An abstract operation is specified by providing the following information:

a) the name of the operation (a qualified name);

b) (optionally) an input message definition;

c) (optionally) an output message definition; and

d) a set of zero or more fault message definitions.

12.3.5 If both an input message and output message definition are present, the order in which they occur (for example, a request and response) is not specified by this Recommendation | International Standard but shall be specified by the service description.

12.3.6 The input message definition and the output message definition shall have one of the following forms:

a) zero or one top-level **element declaration** that belongs to one of the schemas in the original schema set (see 12.2.4); or

b) a list of zero or more distinct unqualified names, each associated with a top-level **complex type definition** or **simple type definition** that belongs to one of the schemas in the original schema set (see 12.2.4).

NOTE – In some service description languages (for example, WSDL 1.1 [2]), the form of the input or output message definition will be specified by the operation binding as constraints on information (not specified by this Recommendation | International Standard) that is provided by the abstract operation.

12.3.7 Each fault message definition shall specify a top-level **element declaration** that belongs to one of the schemas in the original schema set (see 12.2.4).

12.3.8 An interface is called document-based if for each operation the input message definition (if present) has form a) of 12.3.6 and the output message definition (if present) has form a).

12.3.9 An interface is called RPC-based if for each operation the input message definition is present and has form b) of 12.3.6 and the output message definition (if present) has form b).

12.3.10 Otherwise, the interface is neither document-based nor RPC-based and a service description containing such abstract interfaces is not a SOAP-oriented service description as specified by this Recommendation | International Standard.

12.4 Interface bindings and operation bindings

12.4.1 An interface binding associates an abstract interface with additional information, resulting in the complete specification of a concrete interface.

12.4.2 An interface binding is specified by providing the following information:

a) (optionally) an object identifier assigned to the concrete operation;

b) a set of operation bindings;

c) the URI of a transport;

d) the style of the concrete interface (either document-style or RPC-style); and

e) whether the concrete interface supports Fast Web Services.

12.4.3 The object identifier assigned to the concrete operation (if any) shall be allocated according to ITU-T Rec. X.660 | ISO/IEC 9834-1 and shall uniquely identify the concrete operation.

NOTE – Two concrete operations based on the same abstract operation will have different object identifiers.

12.4.4 An operation binding associates an abstract operation with additional information, resulting in the complete specification of a concrete operation that is to be performed through the exchange of W3C SOAP messages.

12.4.5 A transport is a protocol used to transmit a SOAP message from one SOAP node to another SOAP node, and shall be specified as a URI.

NOTE – Within the context of SOAP, transports are called bindings. Common examples are the XML SOAP HTTP binding (see W3C SOAP Part 2, clause 7) and the ASN.1 SOAP HTTP binding (see clause 10).

12.4.6 If the abstract interface is document-based, then the style of the concrete interface shall be document-style. If the abstract interface is RPC-based, then the style of the concrete interface shall be RPC-style.

12.4.7 An interface binding that supports Fast Web Services is referred to as an ASN.1 SOAP interface binding, and concrete operations may be performed through the exchange of ASN.1 SOAP messages.

12.4.8 An operation binding is specified by providing the following information:

a) (optionally) a SOAP action URI;

b) zero or more SOAP header block definitions, each consisting of a top-level **element declaration**;

c) zero or more object identifiers assigned to top-level **element declaration**s; and

d) for each top-level **element declaration**, an indication of whether it is to be represented as a subtree or as an embedded ASN.1 value.

12.4.9 A SOAP action URI is a URI to be placed (if present) in the **action** parameter of the "**application/fastsoap**" MIME type (see B.1) for an HTTP request when the ASN.1 SOAP HTTP binding (see clause 10) is specified as the transport.

12.4.10 Each top-level **element declaration** in a header block definition shall belong to one of the schemas in the original schema set (see 12.2.4).

12.4.11 If an object identifier is assigned to the concrete operation, a unique object identifier may be assigned to one or more of the top-level **element declaration**s specified in an input message definition, output message definition, fault message definition, header block definition, or implicitly generated for an RPC-style concrete operation. If the concrete operation has no object identifier, the assignment of object identifiers to those **element declaration**s is not permitted.

12.4.12 Each object identifier assigned to an **element declaration** shall be allocated according to ITU-T Rec. X.660 | ISO/IEC 9834-1 and shall uniquely identify the **element declaration**. Each such object identifier shall be the same as the object identifier of the concrete operation with one or more additional object identifier components added to the right.

NOTE – This enables the use of relative object identifiers to identify the type of an embedded ASN.1 encoded value in SOAP messages (see 9.2.2). Each such relative object identifier will consist only of the additional object identifier components, while the earlier components of the object identifier will not be transmitted.

12.4.13 Any **element declaration** with an object identifier assigned to it shall be an additional indication to 12.4.8 d) that the **element declaration** is to be encoded as an embedded ASN.1 encoded value rather than as a subtree.

NOTE – The use of embedded ASN.1 encoded values does not require an object identifier. In the absence of an object identifier for a top-level **element declaration**, the qualified name of the **element declaration** is used to identify the type of an embedded ASN.1 encoded value in SOAP messages (see 9.2.3).

12.5 RPC schema

12.5.1 An RPC schema is a specially-constructed XSD schema supporting RPC-style concrete interfaces (see 12.4.6). An RPC schema is implicitly generated and not imported or embedded in a service description.

NOTE – An RPC schema is implicitly present in all SOAP-oriented service descriptions, but it is empty if there are no RPC-style interfaces in the service description.

12.5.2 The RPC schema for a given abstract interface (that is, bound to a RPC-style concrete interface) shall be constructed as follows.

12.5.3 For each abstract operation specified in the RPC-based abstract interface, an **element declaration** schema component with the following properties:

- **name**: the local name of the operation;
- **target namespace**: the namespace name of the operation;
- **type definition**: a **complex type definition** schema component as specified in 12.5.4;
- **scope**: **global**,

and with the remaining properties either **absent** or set to **false** or empty (as appropriate), shall be added to the RPC schema.

12.5.4 The **complex type definition** schema component in the **type definition** property shall have the following properties:

- **name**: none;
- **target namespace**: **absent**;
- **base type definition**: the **ur-type**;
- **derivation method**: **restriction**;
- **content type**: **element-only**, and a **particle** schema component as specified in 12.5.5,

with the remaining properties either **absent** or set to **false** or empty (as appropriate), and shall be added to the RPC schema.

12.5.5 The **particle** schema component in the **content type** property shall have the following properties:

- **min occurs**: 1;
- **max occurs**: 1;
- **term**: a **model group** schema component as specified in 12.5.6,

and shall be added to the RPC schema.

12.5.6 The **model group** schema component in the **term** property shall have the following properties:

- **compositor**: **sequence**;
- **particles**: a list of zero or more **particle** schema components as specified in 12.5.7 (see 12.3.6 b),

and shall be added to the RPC schema.

12.5.7 Each **particle** in the list of particles in the **particle**s property shall have the following properties:

- **min occurs**: 1;
- **max occurs**: 1;
- **term**: an **element declaration** schema component as specified in 12.5.8,

and shall be added to the RPC schema.

12.5.8 The **element declaration** schema component in the **term** property shall have the following properties:

- **name**: one of the unqualified names specified in the input message definition of the abstract operation;
- **target namespace**: absent;
- **type definition**: the **complex type definition** or **simple type definition** schema component associated with the unqualified name in the input message definition of the abstract operation (see 12.3.6 b);
- **scope**: the **complex type definition** schema component specified in 12.5.4,

and shall be added to the RPC schema.

12.5.9 For each abstract operation specified in the RPC-based abstract interface with an output message definition, an **element declaration** schema component with the following properties:

- **name**: the local name of the operation with the suffix "Response";
- **target namespace**: the namespace name of the operation;
- **type definition**: a **complex type definition** schema component as specified in 12.5.10;
- **scope**: **global**,

and with the remaining properties either **absent** or set to **false** or empty (as appropriate), shall be added to the RPC schema.

12.5.10 The **complex type definition** schema component in the **type definition** property shall have the following properties:

- **name**: none;
- **target namespace**: **absent**;
- **base type definition**: the **ur-type**;
- **derivation method**: **restriction**;
- **content type**: **element-only**, and a **particle** schema component as specified in 12.5.11,

with the remaining properties either **absent** or set to **false** or empty (as appropriate), and shall be added to the RPC schema.

12.5.11 The **particle** schema component in the **content type** property shall have the following properties:

- **min occurs**: 1;
- **max occurs**: 1;
- **term**: a **model group** schema component as specified in 12.5.15,

and shall be added to the RPC schema.

12.5.12 The **model group** schema component in the **term** property shall have the following properties:

- **compositor**: **sequence**;
- **particles**: a list of zero or more **particle** schema components as specified in 12.5.13 (see 12.3.6 b),

and shall be added to the RPC schema.

12.5.13 Each **particle** in the list of particles in the **particle**s property shall have the following properties:

- **min occurs**: 1;

- **max occurs**: 1;
- **term**: an **element declaration** schema component as specified in 12.5.14,

and shall be added to the RPC schema.

12.5.14 The **element declaration** schema component in the **term** property shall have the following properties:

- **name**: one of the unqualified names specified in the output message definition of the abstract operation;
- **target namespace**: **absent**;
- **type definition**: the **complex type definition** or **simple type definition** schema component associated with the unqualified name in the output message definition of the abstract operation (see 12.3.6 b);
- **scope**: the **complex type definition** schema component specified in 12.5.4,

and shall be added to the RPC schema.

12.5.15 The **complex type definition** or **simple type definition** schema component in the **type definition** property in 12.5.8 and 12.5.14 shall be a copy of a schema component in one of the XSD schemas in the original schema set. This schema component shall be added to the RPC schema (unless previously added), along with a copy of any schema component that occurs in one of its properties (unless previously added) or in a property within a property, to any depth.

13 Use of SOAP-oriented service descriptions with ASN.1 SOAP interface bindings

13.1 A SOAP-oriented service description containing ASN.1 SOAP interface bindings for a given Fast Web Service affects the form and content of all the ASN.1 SOAP messages mapped to and from W3C SOAP messages (described by the ASN.1 SOAP interface bindings) in the provision of that Fast Web Service.

13.2 Each W3C SOAP message shall be an input or an output message of a concrete operation of a concrete interface (see 12.4) specified in the service description. Those that are input messages shall go from a client SOAP node to a service SOAP node, and those that are output messages shall go in the opposite direction. W3C SOAP messages that are output messages of concrete operation of an RPC-style concrete interface are permitted only for concrete operations that have an input message definition.

13.3 Any header block (child of the **Header** EII) or fault (child of the **Detail** EII) in a W3C SOAP message, which is an input or output message of a given concrete operation, shall be an embedded element information item complying with one of the top-level **element declaration**s for header blocks and faults (respectively) of that operation (see 12.3.7 and 12.4.10 respectively).

13.4 The body (child of the **Body** EII) of a W3C SOAP message, which is an input or output message of a given concrete operation, shall be an embedded element information item complying with the following **element declaration**:

a) if the concrete operation is a member of an RPC-style concrete interface (see 12.4.6), the top-level **element declaration** implicitly generated (in the RPC schema, see 12.5) for the input or output message of that operation, respectively (see 12.5.3 and 12.5.9 respectively); or

b) if the concrete operation is a member of a document-style concrete interface (see 12.4.6), the top-level **element declaration** specified in the input or output message definition of that operation, respectively (see 12.3.6 a).

13.5 An embedded element information item that is described by an ASN.1 SOAP interface binding shall be represented in a W3C SOAP message (that is mapped to an ASN.1 SOAP message) as follows:

a) as an element information item that is a subtree; or

NOTE 1 – Such items will be mapped to components of an ASN.1 SOAP message that are embedded fast infoset documents.

b) as an embedded ASN.1 encoded value that is generated from the element information item.

NOTE 2 – The service description indicates whether the embedded element information item is represented as a subtree or as an embedded ASN.1 encoded value (see 12.4.8 d).

13.6 The generation of an embedded ASN.1 encoded value shall require the following information:

a) an ASN.1 type;

b) an identification of the ASN.1 type; and

c) an ASN.1 value of the identified ASN.1 type.

13.7 The ASN.1 type shall be a member of the ASN.1 schema set of the abstract interface (see 12.3.3), mapped from the top-level **element declaration**, and the embedded element information item compiles with the top-level **element declaration**.

13.8 The identification of the ASN.1 type shall be a value of the **`Identifier`** type.

13.8.1 If an object identifier is assigned to the top-level **element declaration** (see 12.4.12), then the **`roid`** alternative of the **`Identifier`** value shall apply, and the **`roid`** value shall be set from the relative object identifier that is the additional object identifier component to the assigned object identifier (see 12.4.12).

13.8.2 Otherwise the **`qName`** alternative of the **`Identifier`** value shall apply, and the **`qName`** value shall be set from the qualified name of the top-level **element declaration**.

13.9 Given the ASN.1 type, the **`Identifier`** value, and the ASN.1 value, an embedded ASN.1 encoded value shall be generated and inserted into the W3C SOAP message as specified in 9.4.

NOTE – Subclause 9.4 specifies the insertion of an ASN.1 value (with an identifier) into a W3C SOAP message. The ASN.1 value will be represented as a sequence of CIIs that is the Base64 encoding of the Basic Aligned PER encoding of the ASN.1 value. Such a representation will be mapped to a component of an ASN.1 SOAP message that is an octet string, whose value is the Basic Aligned PER encoding of corresponding ASN.1 value.

Annex A

ASN.1 module for ASN.1 SOAP

(This annex forms an integral part of this Recommendation | International Standard)

The ASN.1 module for ASN.1 SOAP is given below. The schema reuses some types defined in the XSD module as specified in ITU-T Rec. X.694 | ISO/IEC 8825-5 and the **Document** type defined in the **FastInfoset** module as specified in ITU-T Rec. X.891 | ISO/IEC 24824-1.

```
ASN1SOAP {joint-iso-itu-t(2) asn1(1) generic-applications(10) fast-web-services(1)
        modules(0) asn1soap(0)}
DEFINITIONS AUTOMATIC TAGS ::=
BEGIN
IMPORTS
      AnyURI, Int, Language, QName
         FROM XSD {joint-iso-itu-t asn1(1) specification(0) modules(0)
                   xsd-module(2)}
      Document, finf-doc-no-decl
         FROM FastInfoset {joint-iso-itu-t(2) asn1(1) generic-applications(10)
                           fast-infoset(0) modules(0) fast-infoset(0)};

Envelope ::= SEQUENCE {
      header        Header,
      body-or-fault CHOICE {
          body  Body,
          fault Fault}}

Header ::= SEQUENCE OF HeaderBlock

HeaderBlock ::= SEQUENCE {
      mustUnderstand BOOLEAN OPTIONAL,
      relay          BOOLEAN OPTIONAL,
      role           XSD.AnyURI DEFAULT ultimateReceiver,
      content        Content}

ultimateReceiver XSD.AnyURI ::=
      "http://www.w3.org/2003/05/soap-envelope/role/UltimateReceiver"

Body ::= SEQUENCE {
      content Content OPTIONAL}

Fault ::= SEQUENCE {
      code   Code,
      reason SEQUENCE SIZE(1..MAX) OF Text,
      node   XSD.AnyURI OPTIONAL,
      role   XSD.AnyURI OPTIONAL,
      detail Content OPTIONAL}

Code ::= SEQUENCE {
      value    Value,
      subcodes SEQUENCE OF XSD.QName}

Value ::= ENUMERATED { versionMismatch, mustUnderstand, dataEncodingUnknown,
                       sender, receiver}

Text ::= SEQUENCE {
      lang XSD.Language,
      text UTF8String}

Content ::= CHOICE {
      encoded-value SEQUENCE {
          schema-identifier OCTET STRING (SIZE (16)) OPTIONAL,
          id         Identifier,
          encoding   OCTET STRING },
      fast-infoset-document OCTET STRING
              (CONTAINING Document ENCODED BY finf-doc-no-decl)}

Identifier ::= CHOICE {
      roid   RELATIVE-OID,
      qName  XSD.QName}

NotUnderstood ::= XSD.QName
```

```
notUnderstoodIdentifier Identifier ::= qName : {
      uri   "http://www/w3/org/2003/05/soap-envelope",
      name  "NotUnderstood"}

END
```

Annex B

MIME media types for Fast Web Services

(This annex forms an integral part of this Recommendation | International Standard)

This annex defines two MIME media types for use with Fast Web Services:

a) the "**application/fastsoap**" media type describes ASN.1 SOAP messages, specifically values of the ASN.1 **Envelope** type, encoded using the Basic Aligned PER encoding (see B.1);

b) the "**application/soap+fastinfoset**" media type describes fast infoset W3C SOAP messages serialized as fast infoset documents (see B.2).

The MIME media types are specified below using the IETF MIME registration template, and have been registered in accordance with IETF procedures.

B.1 The "application/fastsoap" media type

```
MIME media type name:
       application

MIME subtype name:
       fastsoap

Required parameters:
       None.

Optional parameters:
       "action": This parameter shall be used to identify the intent of the ASN.1 SOAP
       message as specified for the "action" parameter of the W3C SOAP 1.2
       "application/soap+xml" MIME media type (see W3C SOAP 1.2 Part 2, Appendix A). The
       value of the "action" parameter shall be an absolute URI-reference as specified in
       IETF RFC 2396. No restriction shall be placed on the specificity of the URI or
       that it is resolvable.


Encoding considerations:
       This media type is used to identify content that is a value of the ASN.1 Envelope
       type specified in the ASN1SOAP module of ITU-T Rec. X.892 | ISO/IEC 24824-2,
       encoded using Basic Aligned Packed Encoding Rule specified in ITU-T Rec. X.691 |
       ISO/IEC 8825-2.

       Use of this MIME media type will require additional specification if used on
       transports that do not provide 8-bit binary transparency.  (For the purposes of
       Fast Web Services, ITU-T Rec. X.892 | ISO/IEC 24824-2, this media type is always
       used with the Basic Aligned Packed Encoding Rule and with HTTP as the transport
       mechanism, and no further specification is needed.)

Security considerations:
       Because ASN.1 SOAP messages can carry application-defined data whose semantics is
       independent from that of any MIME wrapper (or context within which the MIME
       wrapper is used), one should not expect to be able to understand the semantics of
       the ASN.1 SOAP message based on the semantics of the MIME wrapper alone.
       Therefore, whenever using the "application/fastsoap" media type, it is strongly
       recommended that the security implications of the context within which the ASN.1
       SOAP message is used is fully understood. The security implications are likely to
       involve the specific ASN.1 SOAP binding to an underlying protocol as well as the
       application-defined semantics of the data carried in the ASN.1 SOAP message.

Interoperability considerations:
       There are no known interoperability issues.

Published specification:
        ITU-T Rec. X.892 | ISO/IEC 24824-2

Applications which use this media type:
       No known applications that use this media type.

Additional information:
       File extension(s):
              ASN.1 SOAP messages are not required or expected to be stored as files.

Person & e-mail address to contact for further information:
       ITU-T ASN.1 Rapporteur (contact via tsbmail@itu.int)
       ISO/IEC JTC1/SC6 ASN.1 Rapporteur (contact via ittf@iso.org)
```

Intended usage:
COMMON

Author/Change controller:
Joint ITU-T | ISO/IEC balloting procedures in accordance with ITU-T Rec. A.23 *Collaboration with the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) on information technology*, Annex A and ISO/IEC JTC1 Directives, Annex K.

B.2 The "application/soap+fastinfoset" media type

MIME media type name:
application

MIME subtype name:
soap+fastinfoset

Required parameters:
None.

Optional parameters:
"action": This parameter shall be used to identify the intent of the W3C SOAP message infoset as specified for the "action" parameter of the W3C SOAP 1.2 "application/soap+xml" MIME media type (see W3C SOAP 1.2 Part 2, Appendix A). The value of the "action" parameter shall be an absolute URI-reference as specified in IETF RFC 2396. No restriction shall be placed on the specificity of the URI or that it is resolvable.

Encoding considerations:
This media type is used to identify W3C SOAP message infosets serialized as fast infoset documents as specified in ITU-T Rec. X.892 | ISO/IEC 24824-2.

Use of this MIME media type will require additional specification if used on transports that do not provide 8-bit binary transparency. (For the purposes of Fast Web Services, ITU-T Rec. X.892 | ISO/IEC 24824-2, this media type is always used with HTTP as the transport mechanism, and no further specification is needed.)

Security considerations:
Because W3C SOAP message infosets can carry application defined data whose semantics is independent from that of any MIME wrapper (or context within which the MIME wrapper is used), one should not expect to be able to understand the semantics of the W3C SOAP message infoset based on the semantics of the MIME wrapper alone. Therefore, whenever using the "application/soap+fastinfoset" media type, it is strongly recommended that the security implications of the context within which the W3C SOAP message infoset is used is fully understood. The security implications are likely to involve the specific SOAP binding to an underlying protocol as well as the application-defined semantics of the data carried in the W3C SOAP message infoset.

Interoperability considerations:
There are no known interoperability issues.

Published specification:
ITU-T Rec. X.892 | ISO/IEC 24824-2

Applications which use this media type:
No known applications that use this media type.

Additional information:
Magic number(s):
For details on the identification of a fast infoset document, refer to the magic number section of the "application/fastinfoset" media type.

The identification of a W3C SOAP message infoset serialized as a fast infoset document requires that the fast infoset document be parsed and that the properties of the element information item, corresponding to the root of the element tree, conform to the properties of the SOAP Envelope element information item specified in W3C SOAP 1.2, 5.1.

File extension(s):
W3C SOAP message infosets serialized as fast infoset documents are not required or expected to be stored as files.

Person & e-mail address to contact for further information:
ITU-T ASN.1 Rapporteur (contact via tsbmail@itu.int)
ISO/IEC JTC1/SC6 ASN.1 Rapporteur (contact via ittf@iso.org)

Intended usage:
COMMON

Author/Change controller:
Joint ITU-T | ISO/IEC balloting procedures in accordance with ITU-T Rec. A.23 *Collaboration with the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) on information technology*, Annex A and ISO/IEC JTC1 Directives, Annex K.

Annex C

Tutorial on Fast Web Services

(This annex does not form an integral part of this Recommendation | International Standard)

This annex provides tutorial material on Fast Web Services. Some of the advantages of using Fast Web Services are described. The differences between the conceptual and optimized processing of SOAP messages are highlighted, followed by an example. The example is based on a simple exchange in which a client sends a request message and receives a response message. The use of service descriptions is discussed, followed by an example service description (in WSDL 1.1 – see [2]) that describes the service provided by the messaging example.

C.1 Advantages of Fast Web Services

The Fast Web Services specification is based on the use of an ASN.1 definition of SOAP messages and their contents, and on the use of binary encodings of those messages. This provides the main advantage (fast computer processing and low message bandwidth) of Fast Web Service, but a number of further optimizations of XML SOAP are provided that are discussed below.

C.1.1 ASN.1 tools

ASN.1 tools can be used in the development of ASN.1 SOAP processors, whereas XML SOAP processors are, for the most part, written by hand, with the W3C XML Schema for SOAP used only as a guide, since XML binding tools are unlikely to aid in the development of optimal XML SOAP processors. The ASN.1 approach allows for a choice of either tools or handcrafting to develop SOAP processors, without any serious performance penalties, and with potential gains in time-to-market.

C.1.2 Optimized features

ASN.1 SOAP provides a number of optimization features (beyond compaction and efficient processing offered by the use of ASN.1 and PER – see ITU-T Rec. X.691 | ISO/IEC 8825-2) for SOAP nodes:

a) The body of an ASN.1 SOAP message is explicitly separated from the encoding of the fault of an ASN.1 SOAP message. This makes faults easier to identify and manage.

b) Recursive fault subcodes (see W3C SOAP Part 1, 5.4.6) for W3C SOAP messages are flattened to a sequence of fault subcodes for ASN.1 SOAP messages. This enables a decoder to know how many fault subcodes there are before decoding.

c) ASN.1 relative object identifiers can be used instead of qualified names. Messages for service descriptions can be annotated with relative object identifiers and such identifiers, when encoded, are generally much more compact than qualified names, resulting in smaller message sizes.

d) Default values for all attribute-related ASN.1 SOAP header block components are specified.

e) Enumerated values are used for W3C SOAP-specified fault codes instead of qualified names.

C.1.3 Compact messages and efficient processing

ASN.1 SOAP messages encoded using the ASN.1 Packed Encoding Rules generally provides Web services that require less processing power (and hence provide a higher transaction processing rate) and that require less network bandwidth than use of the character encoding of XML data. This can be advantageous in a number of domains:

a) Constrained devices, such as mobile phones, smart cards or even Radio-Frequency Identification (RFID) devices, which have limited processing power, memory and battery life.

NOTE 1 – There is no equivalent Moore's law for battery technology (battery life is not doubling every 18 months).

b) Bandwidth-restricted systems, such as wireless networks.

NOTE 2 – Radio frequencies for wireless networks, such as the mobile phone GSM (Global System for Mobile Communications) network, can be fixed for 10 years. There is no equivalent Moore's law for radio frequencies (bandwidth is not doubling every 18 months).

c) High throughput transaction systems, such as systems required to process a required number of SOAP messages per second from many clients.

C.1.4 Efficient processing for SOAP intermediaries

SOAP intermediaries have the potential to process many more SOAP messages than initial SOAP senders and ultimate SOAP receivers. SOAP intermediaries processing ASN.1 SOAP messages may easily identify ASN.1 SOAP header blocks for processing (including decoding) while skipping (and copying) other SOAP header blocks (destined for other SOAP intermediaries or the SOAP ultimate receiver) and the SOAP body. (This is because the SOAP header blocks and the SOAP body are encoded as a length-prefixed sequence of octets.)

NOTE – ASN.1 SOAP intermediaries can also manage **fault**s efficiently, since a **fault** will always occur at the end of a message (after the SOAP header blocks) and will be guaranteed to start at a byte boundary if header blocks are present. Thus it is not necessary for an intermediary to decode the **fault** unless the intermediary performs processes not specified by the SOAP processing model.

C.2 Conceptual and optimized processing of ASN.1 SOAP messages

C.2.1 General

C.2.1.1 The conceptual mapping from ASN.1 SOAP messages to W3C SOAP messages and vice versa ensures that the W3C SOAP processing model can be applied to ASN.1 SOAP messages. The six following subclauses highlight the conceptual steps required by an initial SOAP sender, a SOAP intermediary and an ultimate SOAP receiver to process messages, and the optimized steps required for a SOAP intermediary.

C.2.1.2 An initial SOAP sender (see W3C SOAP Part 1, 1.5.3) implementing the ASN.1 SOAP HTTP Binding generates ASN.1 SOAP messages in the following steps:

a) create a new W3C SOAP message and insert new embedded ASN.1 abstract values into the W3C SOAP message; and

b) map the W3C SOAP message to an ASN.1 SOAP message; and

c) encode the ASN.1 SOAP message, using Basic Aligned PER, to a sequence of octets that is the content of an HTTP request.

C.2.1.3 If the initial SOAP sender uses the SOAP Request-Response Message Exchange Pattern (see W3C SOAP Part 2, 6.2), then the SOAP sender (see W3C SOAP Part 1, 1.5.3) will wait for a response and change roles to become an ultimate SOAP receiver (see W3C SOAP Part 1, 1.5.3).

C.2.1.4 A SOAP intermediary (see W3C SOAP Part 1, 1.5.3) implementing the ASN.1 SOAP HTTP Binding processes ASN.1 SOAP messages in the following steps:

a) decode the sequence of octets, obtained from the content of an HTTP request or response, using Basic Aligned PER, to obtain an inbound ASN.1 SOAP message; and

b) map the inbound ASN.1 SOAP message to obtain an inbound W3C SOAP message; and

c) identify and process embedded ASN.1 abstract values in the inbound W3C SOAP message; and

d) modify the inbound W3C SOAP message to become an outbound W3C SOAP message and insert new embedded ASN.1 abstract values into the outbound W3C SOAP message; and

e) map the outbound W3C SOAP message to an outbound ASN.1 SOAP message; and

f) encode the outbound ASN.1 SOAP message, using Basic Aligned PER, to a sequence of octets that is the content of an HTTP response or request.

C.2.1.5 An ultimate SOAP receiver implementing the ASN.1 SOAP HTTP Binding processes ASN.1 SOAP messages in the following steps:

a) decode the sequence of octets, obtained from the content of an HTTP request, using Basic Aligned PER, to obtain an ASN.1 SOAP message; and

b) map the ASN.1 SOAP message to obtain a W3C SOAP message; and

c) identify and process embedded ASN.1 abstract values in the W3C SOAP message.

C.2.1.6 If the ultimate SOAP receiver uses the SOAP Request-Response Message Exchange Pattern, then the SOAP node will change roles to become an initial SOAP sender and will send an ASN.1 SOAP message in reply.

C.2.1.7 The conceptual steps to map to and from W3C SOAP messages and process embedded ASN.1 values (identify and process in a W3C SOAP message and insert into a W3C SOAP message) are specified in clauses 6 to 9. A SOAP node may, however, choose to optimize the process by skipping the conceptual steps as long as the results are the same as if the conceptual steps were performed (see 6.4). For example steps b to e in C.2.1.4 are conceptual steps and the SOAP intermediary may choose to optimize by processing ASN.1 SOAP messages in the following steps:

a) decode the sequence of octets, obtained from content of the HTTP request, using Basic Aligned PER, to obtain an inbound ASN.1 SOAP message; and

b) identify and process embedded ASN.1 abstract values in the inbound ASN.1 SOAP message; and

c) modify the inbound ASN.1 SOAP message to become an outbound ASN.1 SOAP message (or create a new outbound ASN.1 SOAP message) and insert new embedded ASN.1 abstract values into the outbound ASN.1 SOAP message; and

d) encode the outbound ASN.1 SOAP message, using Basic Aligned PER, to a sequence of octets that is the content of the HTTP response.

C.2.2 Example

An example is given in the following subclauses from the perspective of an application sending an ASN.1 SOAP message request and receiving a response. The Fast Web Service is specified in C.3.2 using WSDL 1.1, and is based on the sample W3C SOAP message in W3C SOAP Part 1, 1.4. The service is essentially one in which an application may request the latest alert concerning some information that is important to the application (or user of the application). The requesting application will send an empty ASN.1 SOAP message (with no application-defined content) and receive, in response, an ASN.1 SOAP message with two pieces application-defined content (specified in C.3.2 using WSDL 1.1) for the alert that corresponds to:

a) a SOAP header block for properties of the alert, namely the priority of the alert and the time it expires; and

b) a SOAP body content for the alert itself, which is a textual description of the alert.

C.2.2.1 W3C SOAP message request

The application requests the latest alert by executing (using some appropriate programming language, such as Java) a method call with no input parameters that will return the alert. The initial SOAP sender will create a W3C SOAP message, with no content, represented in XML as:

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
        <env:Body>
        </env:Body>
</env:Envelope>
```

C.2.2.2 ASN.1 SOAP message request

This W3C SOAP message is mapped to an ASN.1 SOAP message request consisting of:

```
envelope Envelope ::= {
      header {}
      body-or-fault : body {} }
```

where the **`Envelope`** type is defined in Annex A (see also 6.1).

C.2.2.3 HTTP request

The ASN.1 SOAP message is then encoded, using Basic Aligned PER, to a sequence of octets that is the content of an HTTP request. The HTTP header field **Content-Type** is "**application/fastsoap**" and the **action** parameter is set to "**urn:alert**". The initial SOAP node declares, using the HTTP header field **Accept,** that both ASN.1 SOAP messages and XML SOAP messages (in this case SOAP 1.1 messages [1]) are supported.

```
POST /AlertPort HTTP/1.1
Content-Type: application/fastsoap; action="urn:alert"
Accepts: application/fastsoap, application/text+xml
Content-Length: ....

... sequence of octets …
```

C.2.2.4 HTTP response

The initial SOAP sender will then change roles and become an ultimate SOAP receiver and waits until it receives a response to the request. The HTTP header field **Content-Type** on the response is "**application/fastsoap**".

```
HTTP/1.1 200 OK
Content-Type: application/fastsoap
Content-Length: ....

... sequence of octets …
```

C.2.2.5 ASN.1 SOAP message response

The ASN.1 SOAP message is decoded using Basic Aligned PER, to produce the ASN.1 value:

```
envelope Envelope ::= {
  header { {
    role "http://example.org/alertrole",
    content : encoded-value {
      id : qName {
        uri "http://example.org/alertcontrol",
        name "alertcontrol"},
      encoding  {......}}}},
  body-or-fault : body {
    content : encoded-value {
      id : qnName {
        uri "http://example.org/alert",
        name "alert"},
      encoding {......}}}}
```

C.2.2.6 W3C SOAP message response

C.2.2.6.1 The ASN.1 SOAP message is mapped to a W3C SOAP message. The W3C SOAP message contains an **alertcontrol** W3C SOAP header block and an **alert** element (information item) as the child of a **Body** EII:

```
<env:Envelope
        xmlns:env="http://www.w3.org/2003/05/soap-envelope">
    <env:Header>
        <n:altertcontrol
                xmlns:n="http://example.org/alertcontrol"
                env:role="http://example.org/alertrole"
env:encodingStyle="urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope:encoding-style:aper">
        … Base64 content …
        </n:altertcontrol>
    </env:Header>
    <env:Body>
        <m:alert
                xmlns:m=" http://example.org/alert"
env:encodingStyle="urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:soap-envelope:encoding-style:aper">
        … Base64 content …
        </m:alert>
    </env:Body>
</env:Envelope>
```

C.2.2.6.2 The embedded ASN.1 abstract value for the **alertcontrol** W3C SOAP header block is identified and processed since the requesting SOAP node operates in the "**http://example.org/alertrole**" role. The `Identifier` value for the **alertcontrol** W3C SOAP header block and the embedded ASN.1 value, decoded using Basic Aligned PER from the Base64 content using the ASN.1 type, `AlertControl`, associated with the `Identifier`, are as follows:

```
alertControlIdentifier Identifier ::= qName : {
          uri "http://example.org/alertcontrol",
          name "altertcontrol" }

alertcontrol Alertcontrol ::= {
          role "http://example.org/alertrole",
          priority 1,
          expires "2001-06-22T14:00:00-05:00" }
```

C.2.2.6.3 The embedded ASN.1 abstract value for the **alert** element (information item) is identified and processed as the SOAP node is an ultimate SOAP receiver. The `Identifier` value for the **alert** and the embedded ASN.1 value, decoded using Basic Aligned PER from the Base64 content using the ASN.1 type `Alert` associated with the `Identifier`, are as follows:

```
alertIdentifier Identifier ::= qName : {
          uri "http://example.org/alert",
          name "alert" }

alert Alert ::= {
          msg "Pick up Mary at school at 2pm" }
```

C.3 Service descriptions

C.3.1 General

C.3.1.1 Service descriptions expressed in WSDL 1.1 [2] can be used, without modification, for the description of ASN.1 SOAP endpoints. This increases the scope and usage of Fast Web Services, since the impact on the Web services developers is minimized.

C.3.1.2 A WSDL 1.1 binding interface (see Annex E) for SOAP 1.1 [1] can be reused for an ASN.1 SOAP binding interface provided the WSDL document is a SOAP-oriented service description (see clause 12 and Annex E) and WSDL 1.1 binding conforms to the clarifications and amendments specified by the WS-I Basic Profile 1.0 [3] (see Annex E).

C.3.2 Example

C.3.2.1 The service description (expressed in WSDL 1.1) shown in C.3.3 specifies an ASN.1 SOAP interface binding for the example in C.2.2.

C.3.2.2 The WSDL document has the two **xsd:schema** definitions contained in the **wsdl:type**s (specifying the child content of a **Body** EII, and W3C SOAP header block for the only response). The equivalent ASN.1 schema is obtained by applying ITU-T Rec. X.694 | ISO/IEC 8825-5 to the two schemas (see E.2 and 12.2).

C.3.2.3 The ASN.1 SOAP HTTP Binding will be used because the value of the **transport** attribute on the **soapbind:binding** element (the ASN.1 SOAP interface binding) is equal to "**http://schemas.xmlsoap.org/soap/http/**" (see E.4.2 and 12.4.2).

C.3.2.4 Fast Web Services support is explicitly specified for the ASN.1 SOAP binding interface by use of the ASN.1 SOAP interface binding annotation (a **fast-service:binding** element) in the **wsdl:binding** element and after the **soapbind:binding** element (see E.4.5 and 12.4.2 e).

C.3.2.5 The style of the ASN.1 SOAP binding interface is the document-style (see E.4.3 and 12.4.2 d), since the interface binding conforms to document-literal binding as specified by the WS-I Basic Profile 1.0.

C.3.2.6 The input message definition is empty (no top-level **element declaration**, since the **soapbind:body** in the **wsdl:input** in the **AlertOperation** operation binding references, implicitly, no **wsdl:parts** (see E.4.9.1 and 12.3.6 a)). However, a SOAP action URI exists, since the **AlertOperation** operation binding has a **soapAction** attribute (see E.4.10 and 12.4.9). The URI "**urn:alert**" will be placed in the **action** parameter of the "**application/fastsoap**" MIME type (see B.1) for the HTTP header field **Content-Type** of the HTTP request (that contains the empty ASN.1 SOAP message) .

C.3.2.7 The output message definition has one top-level **element declaration**, **alert:alert** (since the **soapbind:body** in the **wsdl:output** in the **AlertOperation** operation binding references, implicitly, one **wsdl:part** (see E.4.9.1 and 12.3.6 a)).

C.3.2.8 A SOAP header block definition (the **alertcontrol** W3C SOAP header block) is specified for output of the **AlertOperation** operation binding with a top-level **element declaration alertcontrol:alertcontrol** (see E.4.11 and 12.4.8 c)).

C.3.3 Service description expressed in WSDL 1.1

```
<definitions name="Alert"
     xmlns="http://schemas.xmlsoap.org/wsdl/"
     xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/ "
     xmlns:soapbind="http://schemas.xmlsoap.org/wsdl/soap/"
     xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns:fast-service="urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:description"
     xmlns:tns="http://example.org/alert/service"
     targetNamespace="http://example.org/alert/service"
     xmlns:alert="http://example.org/alert"
     xmlns:alertcontrol="http://example.org/alertcontrol">

   <types>
     <schema
          targetNamespace="http://example.org/alertcontrol"
          xmlns="http://www.w3.org/2001/XMLSchema"
          xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
          elementFormDefault="qualified">
       <import namespace=" http://schemas.xmlsoap.org/wsdl/soap/"/>
       <element name="alertcontrol">
          <complexType>
            <sequence>
              <element name="priority" type="xsd:integer"/>
              <element name="expires" type="xsd:dateTime"/>
            </sequence>
            <xsd:attribute ref="soap:role"/>
```

```
            </complexType>
          </element>
        </schema>
        <schema
            targetNamespace="http://example.org/alert"
            xmlns="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified">
          <element name="alert">
            <complexType>
              <sequence>
                <element name="msg" type="xsd:string"/>
              </sequence>
            </complexType>
          </element>
        </schema>
      </types>

      <message name="AlertRequest">
      </message>

      <message name="AlertResponse">
        <part name="header" element="alertcontrol:alertcontrol"/>
        <part name="body" element="alert:alert"/>
      </message>

      <portType name="AlertPortType">
        <operation name="AlertOperation">
          <input message="tns:AlertRequest"/>
          <output message="tns:AlertResponse"/>
        </operation>
      </portType>

      <binding name="AlertBinding" type="tns:AlertPortType">
        <soapbind:binding
          transport="http://schemas.xmlsoap.org/soap/http"
          style="document"/>
        <fast-service:binding/>

        <operation name="AlertOperation" soapAction="urn:alert">
          <input message="tns:AlertRequest">
            <soapbind:body use="literal"/>
          </input>
          <output message="tns:AlertResponse">
            <soapbind:body use="literal" parts="body"/>
            <soapbind:header
                use="literal"
                message="tns:AlertResponse"
                part="header"/>
          </output>
        </operation>
      </binding>

      <service name="AlertService">
        <port name="AlertPort" binding="tns:AlertBinding">
          <soapbind:address location="http://example.org/AlertPort"/>
        </port>
      </service>
    </definitions>
```

Annex D

Common provision of services using Fast Web Services and XML Web services

(This annex does not form an integral part of this Recommendation | International Standard)

This annex describes strategies that may be applied by Fast-enabled Web service clients in order to interoperate with SOAP nodes that are not known to be Fast-enabled. The strategies use the ASN.1 HTTP binding features specified in clause 10.

The result of an applied strategy is a success if a Fast-enabled Web service client identifies a SOAP node as being Fast-enabled and interoperation by the exchange of ASN.1 SOAP messages occurs; otherwise the result is a failure and interoperation by the exchange of XML SOAP messages occurs.

NOTE 1 – This annex assumes that a service description is either not used or does not contain information describing Fast Web Service capability (or such information, if present, is ignored), as specified in clause 12.

Three strategies are described, one optimistic (see D.1) and two pessimistic (see D.2).

NOTE 2 – The importance and usefulness of some of these strategies are affected by whether the majority of Web services use a single request/response for a connection (or not), and on whether caching of information about a particular server is performed.

NOTE 3 – After a client has ascertained if a SOAP node supports Fast Web Service capabilities, then it may no longer be necessary to apply requesting hints (see D.2.1) or responding capability (see D.2.2). However, caching of both client and SOAP node capabilities should be used with care as either implementation could change. The capabilities of a SOAP node can only be guaranteed by the service description or by what has been ascertained over the lifetime of an HTTP connection. HTTP/1.1 has the capability to "keep-alive" connections such that multiple request/response pairs may be sent over the same connection.

D.1 Optimistic strategy

D.1.1 When using this strategy, a Fast-enabled Web service client optimistically assumes that the relevant SOAP nodes are Fast-enabled and capable of processing ASN.1 SOAP message requests and replying with ASN.1 SOAP message responses.

D.1.2 The reception of an ASN.1 SOAP message by a SOAP node may result in two possible outcomes:

a) the SOAP node responds with an HTTP Client Error status code of the 400-series class (see RFC 2616, 10.4). The Fast-enabled Web service client should expect a "415 Unsupported Media Type" HTTP status code, but is required to handle other status codes, namely "400 Bad Request"; or

NOTE 1 – "415 Unsupported Media Type" will occur because the SOAP node does not support the HTTP media type for ASN.1 SOAP messages and hence is not Fast-enabled.

NOTE 2 – HTTP provides an open-ended mechanism for supporting status codes defined by HTTP extensions. A conforming HTTP application must treat any unrecognized 4xx status code as being equivalent to the "400 Bad Request" status code.

b) the SOAP node responds with an ASN.1 SOAP message response.

D.1.3 If the case described in D.1.2 a occurs, then the optimistic strategy has failed and the Fast-enabled Web service client has to re-send a semantically equivalent XML SOAP message to interoperate, or else proceed to the pessimistic strategy described in D.2.

D.1.4 If the case described in D.1.2 b occurs, then the optimistic strategy has succeeded on the first request.

D.2 Pessimistic strategy

When using this strategy, a Fast-enabled Web service client pessimistically assumes that the relevant SOAP nodes may not be Fast-enabled and capable of processing ASN.1 SOAP message requests and replying with ASN.1 SOAP message responses. Two forms of pessimistic strategy are described in D.2.1 and D.2.2.

D.2.1 Pessimistic strategy with requesting hints

D.2.1.1 The Fast-enabled Web service client sends an XML SOAP message with requesting hints corresponding to an HTTP header field **Accept**, as specified in 10.1.4 of the ASN.1 SOAP HTTP Binding. The header field **Accept** will contain HTTP media types for the ASN.1 SOAP message, "**application/fastsoap**", and an XML SOAP message.

NOTE – This strategy utilizes server-driven content-negotiation (see IETF RFC 2616, 12.1), which is an HTTP/1.1 feature. The ASN.1 SOAP HTTP Binding supports HTTP/1.1 and HTTP/1.0. W3C SOAP Part 2, 7.1.2, recommends that implementations use HTTP/1.1.

D.2.1.2 The reception of an XML SOAP message by a SOAP node may result in two possible outcomes:

a) the SOAP node responds with an XML SOAP message; or

b) the SOAP node responds with an ASN.1 SOAP message.

D.2.1.3 If the case described in D.2.1.2 a occurs, then the pessimistic strategy has failed since the SOAP node is not Fast-enabled.

NOTE – Subclause 10.2.2 (ASN.1 SOAP HTTP Binding) guarantees that a SOAP node responds with an ASN.1 SOAP message if capable. Therefore, if failure occurs, it can be guaranteed that the SOAP node does not support Fast Web Services.

D.2.1.4 If the case described in D.2.1.2 b occurs, then the pessimistic strategy has succeeded on the first response.

D.2.2 Pessimistic strategy with responding Fast-enabled capability

D.2.2.1 The Fast-enabled Web service client sends an XML SOAP message with no requesting hints in the HTTP request.

D.2.2.2 The reception of an XML SOAP message by a SOAP node may result in two possible outcomes:

a) the SOAP node responds with an XML SOAP message with no indication of capability; or

b) the SOAP node responds with an XML SOAP message with capability defined as specified in 10.2.3.

NOTE – Fast-enabled capability is stated by an HTTP header field **Fast-Enabled**.

D.2.2.3 If the case described in D.2.2.2 a occurs, then the strategy has failed since the SOAP node is not Fast-enabled. The Fast-enabled Web service client has to re-send a semantically equivalent XML SOAP message to interoperate.

NOTE – Subclause 10.2.3 (ASN.1 SOAP HTTP Binding) guarantees that a SOAP node responds with an HTTP header field **Fast-Enabled** if the node is Fast-enabled. Therefore, if failure occurs, it can be guaranteed that the SOAP node does not support Fast Web Services.

D.2.2.4 If the case described in D.2.2.2 b occurs and the Fast-enabled Web service client can process the HTTP header field **Fast-Enabled**, then the strategy will succeed on the second request.

Annex E

SOAP-oriented service description in WSDL 1.1

(This annex does not form an integral part of this Recommendation | International Standard)

This annex describes the use of WSDL 1.1 [2] in conjunction with the WS-I Basic Profile 1.0 [3] as a language for writing SOAP-oriented service descriptions.

NOTE 1 – The WSDL 1.1 and WS-I Basic Profile 1.0 terminology are reused where appropriate; thus, W3C XML Information Set terms are not used when referring to XML elements and attributes specified by WSDL 1.1 or the WS-I Basic Profile 1.0.

This non-normative annex uses the following namespace prefixes for namespaces:

soapbind	"**http://schemas.xmlsoap.org/wsdl/soap/**"
wsdl	"**http://schemas.xmlsoap.org/wsdl/**"
xsd	"**http://www.w3.org/2001/XMLSchema**"

NOTE 2 – The choice of prefix is not semantically significant.

E.1 SOAP-oriented service descriptions expressed in WSDL 1.1

E.1.1 WSDL 1.1 documents conforming to the profile specified by the WS-I Basic Profile 1.0 meet the requirements of SOAP-oriented service descriptions specified in clause 12 and the use of such SOAP-oriented service descriptions specified in clause 13.

NOTE – The WS-I Basic Profile 1.0 clarifies and amends WSDL 1.1 to promote interoperability.

E.1.2 An interface binding (see E.4) for description of concrete operations that are to be performed through the exchange of SOAP 1.1 [1] messages is interpreted, without modification, as an ASN.1 SOAP interface binding for description of concrete operations that are to be performed through the exchange of ASN.1 SOAP messages mapped from W3C SOAP messages (see E.4.6).

NOTE – This ensures that existing WSDL 1.1 documents can describe Fast Web Services without modification. A Fast Web Service and an XML Web service may be supported using the same binding, and network location (specified by a URI) for receiving input messages and sending output messages that are SOAP 1.1 messages and ASN.1 SOAP messages (mapped from W3C SOAP messages).

E.2 Schema

The original schema set (see 12.2) is the set of XSD schemas declared, using **xsd:schema** elements, in the **wsdl:types** element.

E.3 Abstract interface and abstract operations

E.3.1 An abstract interface (see 12.3.1) is a **wsdl:portType** element in the **wsdl:definition** element. The set of abstract interfaces is the set of all **wsdl:portType** elements in the **wsdl:definition** element.

E.3.2 An abstract operation (see 12.3.4) of the abstract interface is a **wsdl:operation** element in the abstract interface. The set of abstract operations is the set of all **wsdl:operation** elements in the abstract interface.

E.3.3 The name of the operation (see 12.3.4 a) is the value of the **name** attribute on the abstract operation.

E.3.4 An input message definition (see 12.3.4 b) is a **wsdl:input** element in the abstract operation. The **wsdl:input** element is always present for document-based operations (see 12.3.8) and RPC-based operations (see 12.3.9).

NOTE – The WS-I Basic Profile 1.0 constrains the presence and order of input and output message definitions such that one-way operations (an input message definition is present and an output message definition is absent) and request-response operations (an input message definition is present and specified first and an output message definition is present and specified second) are only supported. (The solicit-response operation and notification operation are not supported by the profile).

E.3.5 An output message definition (see 12.3.4 c) is a **wsdl:output** element (if present) in the abstract operation.

E.3.6 A fault message definition (see 12.3.4 d) is a **wsdl:fault** element in the abstract operation. The set of fault message definitions is the set of all **wsdl:fault** elements in the abstract operation.

E.3.6.1 The top-level **element declaration** of the fault message definition (see 12.3.7) is the global **element declaration** that is the value of the **element** attribute on the only **wsdl:part** element in the **wsdl:message** element, which is referenced by the fault message definition (by the **message** attribute).

NOTE – WSDL 1.1, 3.6, specifies the following constraints for fault message definitions: the **wsdl:message** element contains only one **wsdl:part** element; and the **wsdl:part** element references a global **element declaration** (using the **element** attribute).

E.3.7 The form of the input message definition or output message definition (see 12.3.6) is specified by the binding of the abstract operation (see E.4.8).

NOTE 1 – The WS-I Basic Profile 1.0 (see 5.3 and 5.3.1) subsets the set of **wsdl:part** elements of the **wsdl:message** element referenced by input message definition or output message definition of an abstract operation such that the form corresponds to only that specified in 12.3.6 a or b.

NOTE 2 – The WS-I Basic Profile 1.0 constrains all the abstract operations of the abstract interface to only abstract operations specified as document-based (see 12.3.8) or RPC-based (see 12.3.9).

E.4 Interface bindings and operation bindings

E.4.1 An interface binding (see 12.4) is a **wsdl:binding** element (in the **wsdl:definition** element) that contains a **soap:binding** element. The set of interface bindings is the set of all **wsdl:binding** elements in the **wsdl:definition** element.

E.4.2 A URI of the transport (see 12.4.2 c) of a concrete interface is the value of the **transport** attribute on the **soap:binding** element. As specified by the WS-I Basic Profile 1.0, 5.6.2 (requirement R2702), only the HTTP transport is supported and the value of the **transport** attribute is "http://schemas.xmlsoap.org/soap/http". This value specifies the use of the ASN.1 SOAP HTTP Binding (see clause 10) for an ASN.1 SOAP Binding interface (see E.4.6).

E.4.3 The style of the concrete interface is document-style (see 12.4.2 d) if the interface binding conforms to a document-literal binding as specified by WS-I Basic Profile 1.0, 5.3 and 5.3.1.

NOTE – The WS-I Basic Profile 1.0 specifies a document-literal binding to be an interface binding with operation bindings that are all document-literal operations.

E.4.4 The style of the concrete interface is RPC-style (see 12.4.2 d) if the interface binding conforms to an rpc-literal binding as specified by WS-I Basic Profile 1.0, 5.3 and 5.3.1.

NOTE – The WS-I Basic Profile 1.0 specifies an rpc-literal binding to be an interface binding with operation bindings that are all rpc-literal operations.

E.4.5 The interface binding optionally specifies that the concrete interface supports Fast Web Services (see 12.4.2 e) by means of an extension to WSDL 1.1 that is referred to as an ASN.1 SOAP interface binding annotation. The annotation is an EII, occurring as a child of the **wsdl:binding** element and after the **soapbind:binding** element, with:

a) a **[local name]** property of "binding"; and

b) a **[namespace name]** property of "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:description**".

E.4.6 By default all concrete interfaces support Fast Web Services and are ASN.1 SOAP interface bindings (see 12.4.7).

E.4.7 The interface binding optionally specifies an object identifier assigned to all concrete operations (see 12.4.2 a and 12.4.3) by means of an extension to WSDL 1.1 that is referred to as an interface binding object identifier annotation. The annotation is an AII among the members of the **[attributes]** property of the ASN.1 SOAP interface binding annotation (see E.4.5), with:

a) a **[local name]** property of "object-identifier";

b) a **[namespace name]** property of "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:description**"; and

c) a **[normalized value]** property that is the object identifier encoded as an "XMLOIDValue" using only the "XMLNumberForm" (see ITU-T Rec. X.680 | ISO/IEC 8824-1, clause 32).

E.4.8 An operation binding (see 12.4.8) is a **wsdl:operation** element in the interface binding. The set of operation bindings is the set of all **wsdl:operation** elements in the interface binding.

E.4.9 The form of operation binding as specified by WS-I Basic Profile 1.0, 5.3 and 5.3.1, specifies the form of the corresponding message definitions of the abstract operation (see 12.3.6).

E.4.9.1 If the operation binding conforms to a document-literal operation as specified by WS-I Basic Profile 1.0, 5.3 and 5.3.1, then, the input message definition and the output message definition are of the form as specified in 12.3.6 a.

E.4.9.2 For the form specified in 12.3.6 a, the top-level **element declaration** is the global **element declaration** that is the value of the **element** attribute on the **wsdl:part** element referenced explicitly or implicitly by the **soap:body** element. No top-level **element declaration** occurs if no **wsdl:part** element is referenced by **soap:body** element.

NOTE – Requirements R2201, R2210, R2202, R2204, R2208 of the WS-I Basic Profile 1.0 specify that there is zero or one **wsdl:part** element present, and the **element** attribute is present (the **type** attribute is absent) on the element (if present).

E.4.9.3 If the operation binding conforms to an rpc-literal operation as specified by WS-I Basic Profile 1.0, 5.3 and 5.3.1, then, the input message definition and the output message definition are of the form as specified in 12.3.6 b.

E.4.9.4 For the form specified in 12.3.6 b, an unqualified name is the value of the **name** attribute on a referenced **wsdl:part** element and the associated top-level **complex type definition** or **simple type definition** is the value of the **type** attribute on the same **wsdl:part** element. The list of zero or more unqualified names corresponds to those obtained (in the same order) from the list **wsdl:part** elements that are referenced explicitly or implicitly (in an order specified by WS-I Basic Profile 1.0, 5.4.1) by the **soap:body** element.

NOTE – Requirements R2202, R2203, R2207, R2208 of the WS-I Basic Profile 1.0 specify that there is a list of zero or more **wsdl:part** elements present, and the **type** attribute is present (the **element** attribute is absent) on an element. Requirement R2301 specifies the order of the **wsdl:part** elements in the list of elements referenced (explicitly or implicitly) by the **soap:body** element.

E.4.10 A SOAP action URI (see 12.4.8 a) of a concrete operation is the value of the **soapAction** attribute on the **soap:operation** element (if present) in the operation binding.

E.4.11 A SOAP header block definition (see 12.4.8 b and 12.4.11) is the following:

a) a **soap:header** element in either the **wsdl:input** element or in the **wsdl:output** element in the operation binding; and

b) a **soap:headerfault** element in the **soap:header** element.

E.4.11.1 A top-level **element declaration** of the SOAP header block definition is the global **element declaration** that is the value of the **element** attribute on the **wsdl:part** element in the **wsdl:message** element, both elements of which are referenced by the SOAP header block definition (by the **part** and **message** attributes respectively).

NOTE – The WS-I Basic Profile 1.0 specifies for a SOAP header block definition that the **element** attribute on the **wsdl:part** element is present (and the **type** attribute is absent).

E.4.12 An object identifier may be assigned, as an annotation that is an extension to WSDL 1.1, to a top-level **element declaration** (see 12.4.8 c) for the following definitions:

a) an input message definition or output message definition, as an **element declaration** object identifier annotation (see E.4.13), that is an attribute on the input message definition or output message definition respectively;

b) a fault message definition, as an **element declaration** object identifier annotation (see E.4.13), that is an attribute on the fault message definition; and

c) a SOAP header block definition, as an **element declaration** object identifier annotation (see E.4.13), that is among the members of the **[attributes]** property of the SOAP header block annotation (see E.4.14).

E.4.13 An **element declaration** object identifier annotation (an extension to WSDL 1.1) is an AII with:

a) a **[local name]** property of "object-identifier"; and

b) a **[namespace name]** property of "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:description**";

c) a **[normalized value]** property that is the object identifier encoded as an "XMLOIDValue" using only the "XMLNumberForm" (see ITU-T Rec. X.680 | ISO/IEC 8824-1, clause 32).

E.4.14 A SOAP header block annotation (an extension to WSDL 1.1) corresponds (if present) to a SOAP header block definition that is present in the interface binding. The annotation is an EII that is an element in the input message definition or output message definition with:

a) a **[local name]** property of "header";

b) a **[namespace name]** property of "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:description**";

c) an AII among the members of the **[attributes]** property with:

- a **[local name]** property of "message"; and
- a **[normalized value]** property that is equal to the value of the **message** attribute on the corresponding SOAP header block definition;

d) an AII among the members of the **[attributes]** property with:

- a **[local name]** property of "part"; and
- a **[normalized value]** property that is equal to the value of the **part** attribute on the corresponding SOAP header block definition.

E.4.15 By default, top-level **element declaration**s are represented as embedded ASN.1 values (see 12.4.8 d).

E.4.16 A top-level **element declaration** may be represented as a subtree (see 12.4.8 d), by including an annotation that is an extension to WSDL 1.1, for the following definitions:

a) an input message definition or output message definition, as an **element declaration** subtree annotation (see E.4.17), that is an attribute on the input message definition or output message definition respectively;

b) a fault message definition, as an **element declaration** subtree annotation (see E.4.17), that is an attribute on the fault message definition; and

c) a SOAP header block definition, as an **element declaration** subtree annotation (see E.4.17), that is among the members of the **[attributes]** property of the SOAP header block annotation (see E.4.14).

E.4.17 An **element declaration** subtree annotation (an extension to WSDL 1.1) is an AII with:

a) a **[local name]** property of "subtree"; and

b) a **[namespace name]** property of "**urn:ohn:joint-iso-itu-t:asn1:generic-applications:fast-web-services:description**";

c) a **[normalized value]** property of "1" or "true".

E.4.17.1 A **[normalized value]** property of the element subtree annotation that is anything other than "1" or "true" (for example, "0" or "false") is equivalent to omitting the annotation.

E.4.18 A top-level **element declaration** annotated with an **element declaration** subtree annotation and an **element declaration** object identifier annotation is equivalent to a top-level **element declaration** annotated only with an **element declaration** subtree annotation.

Annex F

Assignment of object identifier values

(This annex does not form an integral part of this Recommendation | International Standard)

The following object identifier and object descriptor are assigned in this Recommendation | International Standard:

```
{joint-iso-itu-t(2) asn1(1) generic-applications(10) fast-web-services(1) modules(0)
                  asn1soap(0)}
```

```
"ASN.1 SOAP Module"
```

BIBLIOGRAPHY

[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, *W3C Note, 8 May 2000*. (See http://www.w3.org/TR/2000/NOTE-SOAP-20000508.)

[2] W3C Note, *Web Services Description Language (WSDL) 1.1*, Erik Christensen, Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, *W3C Note, 15 March 2001*. (See http://www.w3.org/TR/2001/NOTE-wsdl-20010315.)

[3] WS-I, *WS-I Basic Profile Version 1.0*, *Final Material, 16 April 2004*. (See http://www.ws-i.org/Profiles/BasicProfile-1.0-2004-04-16.html.)

ICS 35.100.60

Price based on 41 pages