



E-health web services profiles



This Australian Standard® was prepared by Committee IT-014, Health Informatics. It was approved on behalf of the Council of Standards Australia on 3 December 2013.
This Standard was published on 17 December 2013.

The following are represented on Committee IT-014:

- Aged Care Association Australia
- Allied Health Professions Australia
- Australasian College of Health Informatics
- Australian and New Zealand College of Anaesthetists
- Australian Association of Pathology Practices
- Australian College of Nursing
- Australian Commission on Safety and Quality in Healthcare
- Australian Healthcare and Hospital Association
- Australian Industry Group
- Australian Information Industry Association
- Australian Institute of Health and Welfare
- Australian Institute of Radiography
- Australian Medical Association
- Australian Private Hospitals Association
- Commonwealth Department of Health
- Consumers' Federation of Australia
- Consumers' Health Forum of Australia
- CSIRO e-Health Research Centre
- Department of Health (SA)
- Department of Health (Vic.)
- Department of Health (WA)
- Department of Human Services
- Edith Cowan University
- Engineers Australia
- GS1 Australia
- Health Informatics Society of Australia
- Health Information Management Association of Australia
- HL7 Australia
- Integrating the Healthcare Enterprise Australia
- Medical Software Industry Association
- National E-Health Transition Authority
- National ICT Australia
- NSW Ministry of Health
- Queensland Health
- Royal Australian College of Medical Administrators
- Royal College of Pathologists of Australasia
- Services for Australian Rural and Remote Allied Health
- The Pharmacy Guild of Australia
- The Royal Australian and New Zealand College of Radiologists
- The University of Sydney
- University of Western Sydney

Additional Interests:

- ACT Health
- Australian Healthcare Messaging Laboratory
- Buderim Gastroenterology Centre
- CAL2CAL Australia
- Casprel
- Deontik
- DH4
- Flinders University Northern Territory
- Global Health
- Global Informatic Health
- Health Communication Network
- Healthscope
- HL7 Systems and Services
- Kestral Computing
- Lantana Group
- Laughing Mind
- Llewelyn Grain Informatics
- MacIsaac Informatics
- Macquarie Health Corporation
- Medical Objects
- Medisecure
- Michael Legg and Associates
- Montage Systems
- Norman Disney and Young
- Ocean Informatics
- Oridashi
- Primary Healthcare
- Semantic Identity
- Smart Health Solutions
- Sullivan Nicolaides Pathology
- University of Wollongong
- Victoria Avenue Medical Centre

This Standard was issued in draft form for comment as DR AS 5550.

Standards Australia wishes to acknowledge the participation of the expert individuals that contributed to the development of this Standard through their representation on the Committee and through the public comment period.

Keeping Standards up-to-date

Australian Standards® are living documents that reflect progress in science, technology and systems. To maintain their currency, all Standards are periodically reviewed, and new editions are published. Between editions, amendments may be issued.

Standards may also be withdrawn. It is important that readers assure themselves they are using a current Standard, which should include any amendments that may have been published since the Standard was published.

Detailed information about Australian Standards, drafts, amendments and new projects can be found by visiting www.standards.org.au

Standards Australia welcomes suggestions for improvements, and encourages readers to notify us immediately of any apparent inaccuracies or ambiguities. Contact us via email at mail@standards.org.au, or write to Standards Australia, GPO Box 476, Sydney, NSW 2001.

Documents

Document Sharing Hub

Australian Standard[®]

E-health web services profiles

Originated as ATS 5820—2010.
Revised and redesignated as AS 5550—2013.

COPYRIGHT

© Standards Australia Limited

All rights are reserved. No part of this work may be reproduced or copied in any form or by any means, electronic or mechanical, including photocopying, without the written permission of the publisher, unless otherwise permitted under the Copyright Act 1968.

Published by SAI Global Limited under licence from Standards Australia Limited, GPO Box 476, Sydney, NSW 2001, Australia

ISBN 978 1 74342 639 5

PREFACE

This Standard was prepared by the Standards Australia Technical Committee IT-014, Health Informatics, to supersede ATS 5820—2010, *E-health web services profiles*.

This Standard is designed to be compatible with implementations of ATS 5820—2010. The XML namespace and XML Schema are unchanged and there have been minor corrections and clarifications to the conformance points. The changes are not expected to impact communications between implementations of ATS 5820—2010 and this Standard. WS-Security is an optional element in this Standard. WS-Security has been removed from AS 5552 and will be removed from this Standard at the next release. It is currently included for backward compatibility only.

The objective of this Standard is to define a common set of interoperable mechanisms using SOAP Web services for interconnecting computer systems in e-health applications.

Interoperability is enhanced by refining the Web services specifications. The Web services specifications by themselves are not sufficient to ensure that different systems can be interconnected. The specifications cover a broad range of situations, so they contain optional and implementation defined features.

Interoperability is also enhanced by focusing on mechanisms that can be implemented using existing software toolkits. The toolkits may not implement the full specification, may sometimes implement optional or ambiguous features differently, and may contain bugs. The profiles in this Standard were developed to identify the features that can be implemented using existing toolkits, rather than what is permitted by the specifications but cannot be easily implemented.

This Standard defines three profiles: a base Web services profile, and profiles for securing Web services using either TLS or WS-Security. These profiles are designed to be referenced by specific service interface specifications to enable those services to inherit the interoperability properties built into the profiles. These profiles clarify the optional and implementation defined features from the Web services specifications.

The term ‘normative’ has been used in this Standard to define the application of the appendix to which it applies. A ‘normative’ appendix is an integral part of a Standard.

This publication has been developed with assistance from the Australian Government Department of Health. The Australian Government makes no representation or warranty that the information in this publication is correct and accurate.

Standards Australia wishes to thank the Department of Health for its continued financial support in helping to develop this Australian Standard.

CONTENTS

	<i>Page</i>
SECTION 1 SCOPE AND GENERAL	
1.1 SCOPE AND INTENDED AUDIENCE	4
1.2 OVERVIEW	4
1.3 NORMATIVE REFERENCES	5
1.4 DEFINITIONS.....	6
1.5 ACRONYMS AND ABBREVIATIONS	7
1.6 ARTEFACTS	7
SECTION 2 WEB SERVICES BASE PROFILE	
2.1 INTRODUCTION	9
2.2 CONFORMANCE	9
2.3 WEB SERVICES SPECIFICATION	9
2.4 WEB SERVICES POLICIES	12
2.5 HYPERTEXT TRANSPORT PROTOCOL	14
2.6 SOAP PROTOCOL	15
2.7 METADATA	21
SECTION 3 TLS PROFILE	
3.1 INTRODUCTION	28
3.2 CONFORMANCE	28
3.3 PKI FOR TLS	28
3.4 TRANSPORT LAYER SECURITY (TLS).....	29
SECTION 4 WS-SECURITY PROFILE	
4.1 INTRODUCTION	32
4.2 CONFORMANCE	32
4.3 PKI FOR WS-SECURITY	32
4.4 WS-SECURITY	35
APPENDIX A BASE SOAP FAULTS	56
BIBLIOGRAPHY	61

STANDARDS AUSTRALIA

Australian Standard

E-health web services profiles

SECTION 1 SCOPE AND GENERAL

1.1 SCOPE AND INTENDED AUDIENCE

1.1.1 Scope

This Standard defines profiles of the SOAP Web services specifications. It defines a base set of specifications for Web services, so that interoperable Web services can be defined for e-health. It also defines profiles that can be used to secure those Web services.

This Standard does not define any service interfaces. Instead, it specifies profiles that provide a set of mechanisms that service interface authors can use to meet their particular requirements. It is outside the scope of this document to know which requirements are needed by the service interface specifications, and to address them.

The profiles defined in this Standard have been designed with the goal of achieving interconnectivity between implementations using Web services toolkits. Compared to the Web services specifications and other Web services profiles, it provides details to improve interconnectivity.

1.1.2 Intended audience

This Standard is intended for—

- (a) authors of service interface specifications who incorporate the profiles into the service interface specifications they produce;
- (b) solution architects who incorporate the service interface specifications into the system specifications they design;
- (c) software developers who create software programs that conform to the system specifications and service interface specifications; and
- (d) conformance testers who test software to determine how it conforms to the service interface specifications, and the profiles that it uses.

The primary user of these profiles is the service interface specification author. Solution architects, software developers and conformance testers will use these profiles because they are referenced by the service interface specifications which they are using.

The reader is expected to have a detailed knowledge of SOAP Web services, WSDL, WS-Addressing, WS-Policy, TLS, WS-Security and PKI.

1.2 OVERVIEW

This Standard defines the following profiles:

- (a) Web Services Base Profile (Section 2).
- (b) TLS Profile (Section 3).
- (c) WS-Security Profile (Section 4).

1.3 NORMATIVE REFERENCES

The following are the normative documents referenced in this Standard.

NOTE: Documents for informative purposes are listed in the Bibliography.

- | | |
|-----------|--|
| RFC2246 | IETF, RFC 2246: The TLS Protocol Version 1.0, T. Dierks and C. Allen, January 1999.
< http://ietf.org/rfc/rfc2246.txt > |
| RFC2616 | IETF, RFC 2616: HyperText Transfer Protocol—HTTP/1.1, R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, June 1999.
< http://ietf.org/rfc/rfc2616.txt > |
| RFC4122 | IETF, RFC 4122: A Universally Unique Identifier (UUID) URN Namespace, P. Leach, M. Mealling, R. Salz, July 2005.
< http://ietf.org/rfc/rfc4122.txt > |
| SOAP2007b | W3C, SOAP Version 1.2, Part 1: Messaging Framework (Second Edition), W3C Recommendation, 27 April 2007.
< http://www.w3.org/TR/2007/REC-soap12-part1-20070427/ > |
| SOAP2007c | W3C, SOAP Version 1.2, Part 2: Adjuncts (Second Edition), W3C Recommendation, 27 April 2007.
< http://www.w3.org/TR/2007/REC-soap12-part2-20070427/ > |
| WSA2006 | W3C, Web Services Addressing 1.0—Core, W3C Recommendation, 9 May 2006.
< http://www.w3.org/TR/2006/REC-ws-addr-core-20060509 > |
| WSAM2007 | W3C, Web Services Addressing 1.0—Metadata, W3C Recommendation, 4 September 2007.
< http://www.w3.org/TR/2007/REC-ws-addr-metadata-20070904 > |
| WSDL2001 | W3C, Web Services Description Language (WSDL) 1.1, W3C Note, 15 March 2001.
< http://www.w3.org/TR/2001/NOTE-wsdl-20010315 > |
| WSPL2007 | W3C, Web Services Policy 1.5—Framework, W3C Recommendation, 4 September 2007.
< http://www.w3.org/TR/2007/REC-ws-policy-20070904 > |
| WSS2006 | OASIS, Web Services Security: SOAP Message Security 1.1, OASIS Standard, 1 February 2006.
< http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf > |
| WSSPL2007 | OASIS, WS-SecurityPolicy 1.2, OASIS Standard, 1 July 2007.
< http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/ws-securitypolicy-1.2-spec-os.pdf > |
| X509-2005 | ITU-T, Information Technology—Open Systems Interconnection—The Directory: Public-key and attribute certificate frameworks, ITU-T Recommendation X.509, August 2005.
See http://www.itu.int/rec/T-REC-X.509-201210-P/en |

1.4 DEFINITIONS

For the purpose of this Standard, the definitions below apply.

1.4.1 Conformance point

A provision or possible action against which the conformance of a system can be attested. A requirement, indicated by the use of the word 'shall', is mandatory for compliance with this Standard. A recommendation, indicated by the word 'should', provides best practice solutions that are not mandatory.

1.4.2 Profile

Identifies a set of specifications and how they are to be used.

1.4.3 Service invoker

A program that makes calls on the Web service offered by a service provider. It acts as a client. A service invoker functions as a SOAP message creator (because it creates request messages) and as a SOAP message consumer (because it consumes response and fault messages).

1.4.4 Service provider

A program that offers a Web service. It acts as a server. A service provider functions as a SOAP message consumer (because it consumes request messages) and as a SOAP message creator (because it creates SOAP response messages and SOAP fault messages).

1.4.5 Shall

Indicates that a statement is mandatory.

1.4.6 Should

Indicates a recommendation.

1.4.7 SOAP

A protocol for using XML-based messages to exchange structured data in a distributed computing environment.

1.4.8 SOAP message consumer

A program that receives and processes SOAP messages.

1.4.9 SOAP message creator

A program that creates and sends SOAP messages.

1.4.10 Toolkit

A set of tools that assists the implementation of Web services protocols. These tools usually include a programming library, code generation tools, and deployment and configuration programs. A toolkit can be provided with a platform or as a third-party component.

1.5 ACRONYMS AND ABBREVIATIONS

DES	Data Encryption Standard
DMZ	Demilitarized Zone
HTTP	HyperText Transfer Protocol
MEP	Message Exchange Pattern
MIME	Multipurpose Internet Mail Extension
NTP	Network Time Protocol
PKI	Public Key Infrastructure
RPC	Remote Procedure Call
SHA	Secure Hash Algorithm
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UUID	Universally Unique Identifier
WSDL	Web Services Definition Language
XML	Extensible Markup Language

1.6 ARTEFACTS

The conformance points in this document are categorized according to the artefacts to which they apply.

The artefacts are as follows:

- (a) *Service interface specifications*—The conformance points on service interface specifications apply to both service invokers and service providers because both programs are required to conform to the service interface specifications. They also indirectly apply to SOAP message creators and SOAP message consumers because service invokers and service providers are also SOAP message creators and SOAP message consumers. The conformance points on service interface specifications will apply to service invokers and service providers who implement those service interface specifications. The conformance points on service interface specifications apply to them indirectly because they follow the service interface specifications.
- (b) *Service invokers*—Service invokers use a service and create request messages. They also consume response messages and fault messages. A service invoker is also a SOAP message creator because it creates SOAP request messages. A service invoker is also a SOAP message consumer because it consumes SOAP response messages and SOAP fault messages.
- (c) *Service providers*—Service providers offer a service and consume request messages. They also create response messages and fault messages. A service provider is also a SOAP message creator because it creates SOAP response messages and SOAP fault messages. A service provider is also a SOAP message consumer because it consumes SOAP request messages.

- (d) *SOAP message creators*—SOAP message creators generate SOAP messages and SOAP fault messages. SOAP message creators are service invokers generating request messages, and service providers generating response messages and fault messages.
- (e) *SOAP message consumers*—SOAP message consumers process SOAP messages and SOAP fault messages. SOAP message consumers are service providers processing request messages, and service invokers processing response messages and fault messages.

Many of the conformance points in this Standard apply to the service interface specifications rather than directly to software implementations. The software implementations are expected to conform to the service interface specifications (e.g. WSDL) they implement. Therefore, those implementations will indirectly satisfy this Standard if those service interface specifications conform to this Standard.

SECTION 2 WEB SERVICES BASE PROFILE

2.1 INTRODUCTION

The Web Services Base Profile consists of profiles of a set of core specifications for implementing SOAP Web services for e-health.

2.2 CONFORMANCE

2.2.1 General

The Web Services Profile does not include any security mechanisms, but can be used with a number of different security mechanisms. Communicating service provider and service consumer need to use the same security mechanism. Therefore, the service interface specification needs to define what that common security mechanism is.

2.2.2 Use with other profiles

2.2.2.1 *Conformance point on service interface specifications*

The following conformance point applies:

The service interface specifications shall define which security profile (if any) is used with the service.

2.2.2.2 *Informative note*

The Web Services Base profile can be used with the TLS Profile, the WS-Security Profile and with another security mechanism or without any security, depending on the needs of the service.

2.3 WEB SERVICES SPECIFICATION

2.3.1 WSDL 1.1

2.3.1.1 *Conformance point on service interface specifications*

The following conformance point applies:

Service interface specifications shall provide a description of the technical service using the Web Services Description Language (WSDL) 1.1. WSDL 1.1 is defined by WSDL2001.

2.3.1.2 *Informative notes*

WSDL is a machine-readable form for describing what a service does and how to invoke it. The description can be used as part of the formal documentation of the service, and can also be used as input for development tools and programs.

WSDL can also be used to indicate where a service instance is located, but this feature will not be used in this profile. This is because WSDL is being used to define the design-time features of a service interface. In this profile it is not being used to describe a run-time service instance.

A WSDL document is only one part of the service interface specification, but there are aspects of the service interface that the WSDL cannot represent. Those other aspects of the service interface will need to be represented in other machine-readable formats or using natural language.

Programmers should be aware that the WSDL is primarily being used as a specification document. There is no requirement to use the WSDL file to generate computer code, even though most Web services toolkits provide a mechanism to do this. Programmers have the option to implement the service interface using another method as long as the result conforms to what the WSDL specifies.

2.3.2 Separation of interface and bindings

2.3.2.1 Conformance point on service interface specifications

The following conformance point applies:

Service interface specifications should separate their WSDL descriptions into at least two files—one file containing the interface elements *types*, *messages*, *portType*, and a separate file containing the binding and service elements.

2.3.2.2 Informative notes

A single WSDL file is easier to show and describe, but a single WSDL file is difficult to reuse.

To assist with reuse, the concrete aspects can be separated from the abstract aspects. This allows the concrete aspects to be modified without needing to modify the abstract aspects.

The file containing the binding and service elements can use the WSDL import mechanism to refer to the interface WSDL.

2.3.2.3 Example 1: WSDL containing service interface information

```
<?xml version="1.0"?>

<definitions
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:tns="http://ns.example.com/example/..."
  targetNamespace="http://ns.example.com/example/..."
  name="...">

  <types>
    ...
  </types>

  <message name="...">
    ...
  </message>
  ...

  <portType name="...">
    ...
  </portType>

</definitions>
```

2.3.2.4 Example 2: WSDL containing service instance binding information

```
<?xml version="1.0"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/" ...>
  <import namespace="urn:..." location="...wsdl"/>
  <binding ...>
    ...
  </binding>
  <service name="...">
    ...
  </service>
</definitions>
```

2.3.3 Namespace values

2.3.3.1 Conformance points on service interface specifications

The following conformance points apply:

- (a) Service interface specifications shall specify WS-Addressing Action values as described in WSAM2007.
- (b) An absolute URI shall be specified as a 'service namespace' to identify the service. This service namespace shall be a value such that the Action values (defined in list items (A) to (D) below) are absolute URIs.
- (c) A separator character shall be specified. The separator character should be—
 - (i) a slash character (Unicode U+002F) if the service namespace is a URL; and
 - (ii) a colon character (Unicode U+003A) if the service namespace is a URN.

In the following conformance points ((A) to (E)), {serviceNS} is the service namespace, {sep} the separator character, {portType} the port type name, {operation} the operation name, and {faultName} the name of the fault. The remaining text appears literally as a part of the value.

The WS-Addressing Action values should be as follows:

- (A) The target namespace for the WSDL description should be—
{serviceNS}
- (B) The WS-Addressing Action value for input messages should be—
{serviceNS}{sep}{portType}{sep}{operation}Request
- (C) The WS-Addressing Action value for output messages should be—
{serviceNS}{sep}{portType}{sep}{operation}Response
- (D) The WS-Addressing Action value for fault messages should be—
{serviceNS}{sep}{portType}{sep}{operation}{sep}Fault{sep}{faultName}
and
- (E) The XML namespace for the wrapper element should be—
{serviceNS}

2.3.3.2 *Informative notes*

Conformance points 2.3.3.1 (A) to (E) are based on the default convention described in the WS-Addressing bindings for WSDL described in WSAM2007.

See Clause 2.6.4 for the definition of a wrapper element.

The service namespace needs to be globally unique. This is usually achieved by following a particular organization's policies for assigning namespaces. Such policies ensure that the namespaces used by the organization can be uniquely associated with the organization and are uniquely allocated within the organization.

WS-Addressing 1.0 specifies that the Action has to be an absolute Internationalized Resource Identifier (IRI). To satisfy this requirement, it is suggested that the service namespace be an absolute URL. A URN could be used, but can be more difficult to assign because of the registration process for new types of URNs.

2.3.3.3 *Example*

The following example uses the following URL as the service namespace:

`http://ns.example.com/example/Svc/ExampleService/1.0`

This service namespace is treated as an opaque value. Although it appears to have components separated by the slash character, they are not significant to the naming scheme described in this Clause, Clause 2.3.3.

Consider an operation called `bar` with a `portType` of `foo`. This operation can return a `noID` fault.

The target namespace for the WSDL is—

`http://ns.example.com/example/Svc/ExampleService/1.0`

The WS-Addressing Action for the input message is—

`http://ns.example.com/example/Svc/ExampleService/1.0/foo/barRequest`

The WS-Addressing Action for the output message is—

`http://ns.example.com/example/Svc/ExampleService/1.0/foo/barResponse`

The WS-Addressing Action for the fault message is—

`http://ns.example.com/example/Svc/ExampleService/1.0/foo/bar/Fault/noID`

The XML namespace for the wrapper element is—

`http://ns.example.com/example/Svc/ExampleService/1.0`

2.4 WEB SERVICES POLICIES

2.4.1 WS-Policy 1.5—Framework

2.4.1.1 *Conformance points on service interface specifications*

The following conformance points apply:

- (a) Service interface specifications should describe the technical policies of the Web services in the WSDL using WS-Policy 1.5. WS-Policy 1.5 is defined by WSPL2007.
- (b) Service interface specifications should apply the policy statements within the binding element of the WSDL.

2.4.1.2 *Informative note*

WS-Policy 1.5 is a W3C recommendation that defines a framework for specifying the policies of Web services. The framework acts as a container for policy statements, but it does not define any particular policy statements to use. Other specifications define those policy statements.

WS-Policy provides a way to describe some behaviours of a Web service in a WSDL specification. This improves the role of the WSDL as the technical specification of a Web service.

Policies represent the concrete aspects of a Web service. In keeping with the conformance point in Clause 2.3.2, it should be applied to the WSDL binding rather than the WSDL port type.

Currently, only these two policy languages have been identified:

- (a) WS-Security Policy 1.2 [WSSPL2007] (see conformance points in Clause 2.4.2).
- (b) WS-Addressing 1.0 (Metadata) [WSAM2007] (see conformance point in Clause 2.4.3).

2.4.1.3 *Example: WSDL containing service instance information*

The following example shows a policy applied at the root level of the WSDL binding element:

```
<?xml version="1.0"?>

<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:wsp="http://www.w3.org/ns/ws-policy"
  xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wsssecurity-utility-1.0.xsd"
  ...>
  ...
  <binding ...>
    <wsp:PolicyReference URI="#APolicy"/>
    ...
  </binding>
  ...
  <wsp:Policy wsu:Id="APolicy">
    ...
  </wsp:Policy>
</definitions>
```

2.4.2 WS-SecurityPolicy 1.2

2.4.2.1 *Conformance point on service interface specifications*

The following conformance point applies:

Service interface specifications should use WS-SecurityPolicy 1.2 to describe the security constraints and requirements of Web services. WS-SecurityPolicy 1.2 is defined by WSSPL2007.

2.4.2.2 *Informative notes*

WS-SecurityPolicy is an OASIS specification for specifying policy assertions for Web services security specifications such as WS-Security or TLS. It is intended to be used within the WS-Policy framework.

Web services toolkits provide varying levels of support for WS-SecurityPolicy. Some toolkits might use the WS-SecurityPolicy statements to configure their implementations.

Other toolkits might ignore or prohibit WS-SecurityPolicy statements, in which case they will serve only as documentation that is not useful to the toolkit.

Developers will need to check how their toolkit handles WS-SecurityPolicy statements. They should not assume that security is turned on and configured correctly simply because the WSDL from which a Web service was derived contains WS-SecurityPolicy statements. Additionally, some toolkits support a different version or a subset of WS-SecurityPolicy and can reject the policies in the WSDL. As the purpose of the normative WSDL is documentation, developers are permitted to modify their local WSDL as required to adapt it to their toolkit, provided the resulting SOAP messages and interfaces conform to the service interface specifications.

2.4.3 WS-Addressing 1.0—Metadata

2.4.3.1 Conformance point on service interface specifications

The following conformance point applies:

Service interface specifications shall use mechanisms defined in WS-Addressing 1.0 (Metadata) to describe addressing requirements in Web services. WS-Addressing 1.0 (Metadata) is defined by WSAM2007.

2.4.3.2 Informative notes

WS-Addressing 1.0 (Metadata) is a W3C Recommendation that specifies how to define WS-Addressing properties, such as Action, in a WSDL and how to specify the use of addressing in a WSDL using the WS-Policy framework.

Like WS-SecurityPolicy, toolkit support for the mechanisms in WS-Addressing 1.0 (Metadata) can vary. Developers cannot assume that addressing is automatically included by their toolkit just because the addressing policy statements appear in a WSDL.

Conformance points for WS-Addressing are documented in Clause 2.7. Those conformance points translate into statements from WS-Addressing 1.0 (Metadata) which are then used in WSDL descriptions.

NOTE: For a discussion about why WS-Addressing is used by the profile, see Clause 2.7.1.3.

2.5 HYPERTEXT TRANSPORT PROTOCOL

2.5.1 HTTP 1.1

2.5.1.1 Conformance point on service invokers

The following conformance point applies:

Service invokers shall use the HyperText Transport Protocol (HTTP) version 1.1 as the transport mechanism for Web services.

2.5.1.2 Conformance point on service providers

The following conformance point applies:

Service providers shall use the HyperText Transport Protocol (HTTP) version 1.1 as the transport mechanism for Web services. HTTP 1.1 is defined by RFC2616.

2.5.1.3 Informative note

HTTP 1.1 is the most widely supported transport mechanism for Web services. Although other transports have been proposed for SOAP Web services, they are not as widely supported or as mature as HTTP.

2.5.2 HTTP persistent connections

2.5.2.1 Conformance point on service invokers

The following conformance point applies:

Service invokers should avoid closing the TCP/IP connection if not doing so can improve performance.

2.5.2.2 Conformance point on service providers

The following conformance point applies:

Service providers should avoid closing the TCP/IP connection if not doing so can improve performance.

2.5.2.3 Informative notes

Under HTTP 1.1, the underlying TCP/IP connection is kept alive by default. If the client or server wishes to close the connection, they can send a 'Connection: close' HTTP header. See Sections 8.1 and 14.10 of RFC2616 for details. Establishing a new TCP/IP connection for every operation would be inefficient if there are multiple operations being invoked consecutively.

Both the conformance point in Clause 2.5.2.2 and this Clause, 2.5.2.3, suggest that service invokers and service providers should not send 'Connection: close' on every request or response if multiple operations are being consecutively invoked.

2.6 SOAP PROTOCOL

2.6.1 SOAP 1.2

2.6.1.1 Conformance point on service invokers

The following conformance point applies:

Service invokers shall use SOAP 1.2 as the Web services protocol.

2.6.1.2 Conformance point on service providers

The following conformance point applies:

Service providers shall use SOAP 1.2 as the Web services protocol. SOAP 1.2 is defined by SOAP2007b and SOAP2007c.

2.6.1.3 Informative notes

SOAP is the common protocol for Web services. The primer for the SOAP 1.2 specification is SOAP2007a.

SOAP defines a protocol for exchanging messages in a decentralized and distributed environment. It defines how messages are represented using the XML Infoset [INFO2004], which defines a set of XML data concepts. The Infoset is serialized using the XML 1.0 syntax [XML2006]. It also defines a framework for message processing. SOAP is an extensible protocol which can support different types of payloads, behaviours, and interaction patterns.

The Web Services Base profile is based on SOAP 1.2 which is a W3C Recommendation. SOAP 1.1 is a W3C Note, not a W3C Recommendation [SOAP2000].

2.6.2 Document literal encoding

2.6.2.1 Conformance point on service interface specifications

The following conformance point applies:

Service interface specifications shall use the 'document/literal' WSDL style and usage.

2.6.2.2 Informative notes

The SOAP binding for WSDL 1.1 permits different encoding styles [WSDL2001]. Any given Web services operation has to adopt one particular style to use. Some styles are underspecified and are not as well supported by toolkit implementations. Use of those styles can cause integration problems between different toolkits.

In the ‘document/literal’ style, the structure of the SOAP body and the types used are defined by an XML Schema in the ‘Types’ section in the WSDL.

The literal style is used because XML Schema is a mechanism that allows a more expressive description of data than the SOAP encoding mechanism. This style also allows XML Schemas that are developed elsewhere to be used in the messages. The literal style is compliant with the widely implemented WS-I Basic Profile 1.1 [BP2006].

Modern toolkits currently provide better support for the ‘document/literal’ style than other possible styles. Some older toolkits only support RPC style operations. However, those older toolkits lack features needed to satisfy the other conformance points in this profile.

The document style is recommended over the RPC style because the RPC approach is tightly coupled and low-level, which makes it less useful for cross-organization operations in the e-health environment.

A coarse-grain approach should be used with Web services that build upon the document exchange metaphor. The aim is to send fewer messages, and avoid ‘chatty’ interfaces which frequently send messages.

2.6.2.3 Example: WSDL definition

The following example shows the use of the ‘document/literal’ style in a WSDL description of the service interface.

```
<wsdl:definitions
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
  ...
  <wsdl:binding ...>
    <wsdl:operation name="...">
      <soap:operation style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
      <wsdl:fault name="...">
        <soap:fault name="..." use="literal"/>
      </wsdl:fault>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>
```

2.6.3 SOAP Action

2.6.3.1 Conformance point on service interface specifications

The following conformance point applies:

Service interface specifications shall not define SOAP Action values for any operation in the service.

2.6.3.2 *Conformance point on SOAP message consumers*

The following conformance point applies:

SOAP message consumers shall not use any SOAP Action value found in a SOAP message.

2.6.3.3 *Informative notes*

The function of the SOAP Action has been replaced by the WS-Addressing Action header. The WS-Addressing Action is the preferred mechanism for indicating the Web services operation being invoked. Unlike the WS-Addressing Action, the SOAP Action is placed in HTTP headers rather than in the SOAP header. This means that the SOAP Action value cannot be signed using WS-Security (if it is being used), and therefore is open to man-in-the-middle attacks. The conformance points in Clauses 2.6.3.1 and 2.6.3.2 are in line with the WS-I Basic Security Profile [BSP2007], which recommends omitting the SOAP Action value from the WSDL.

Older toolkits could provide support for SOAP Action, but this feature should not be used. If a SOAP Action value is not available, some toolkits will send an empty string as the value of the SOAP Action. SOAP message consumers need to process these messages, but ignore the SOAP Action values in them to be compatible with those toolkits.

2.6.4 **Wrapped convention**

2.6.4.1 *Conformance point on service interface specifications*

The following conformance point applies:

Service interface specifications shall follow the wrapped convention for structuring the body of the SOAP message.

2.6.4.2 *Informative notes*

One of the key benefits of the wrapped convention is that it allows the document/literal style of Web services to be used while retaining the ease of use associated with the RPC/literal style implementations.

The wrapped convention is widely implemented by current toolkits but there is no normative definition. Its use helps improve compatibility with those toolkits.

The body of SOAP messages can contain arbitrary XML, but some toolkits expect the body to follow certain conventions. For example, some toolkits expect the SOAP body to contain only one XML element, and that element name should uniquely identify the operation. If these conventions are not followed, the toolkits cannot route or process the messages.

The wrapped convention is a constraint on how the SOAP body is structured [BUTE2005]. In the wrapped convention, the WSDL definitions have the following characteristics:

- (a) Input and output messages have only one part.
- (b) Each part references an XML Schema element (i.e. not an XML Schema type). This element is called a wrapper element.
- (c) The wrapper element's type is an XML Schema complex type.
- (d) The wrapper element's type has no attributes.
- (e) The local name of the wrapper element in the input message matches the message's operation name.
- (f) The local name of the wrapper element in the output message matches the message's operation name with 'Response' appended to it.

The wrapper element is the root element within the SOAP body. See Clause 2.3.3 for the conventions on the value of the namespace for the wrapper element.

In items (a) and (b) above, a ‘part’ refers to a WSDL part for the SOAP body. WSDL parts are also used to define SOAP headers, but the wrapped convention does not place any constraints on how the SOAP headers are used.

Some toolkits rely on the XML element in the SOAP body to identify the service operation and routing of messages. There are other mechanisms that can (and should) be used for this purpose (i.e. the WS-Addressing Action covered in Clause 2.7.2). However, some earlier toolkits rely on conventions instead of those other mechanisms. Therefore, using the wrapped convention will enable those toolkits to work, without preventing the newer toolkits from using the other mechanisms.

The wrapped convention ensures that the body element contains, at most, one child element, even though SOAP allows the possibility of multiple child elements. Knowing there is at most one child element can simplify how the body is processed. Having multiple elements in the body is not compliant with the WS-I Basic Profile 1.1 [BP2006].

Use of the wrapped convention can also simplify programming with toolkits [MANE2005]. The wrapped convention is akin to defining the RPC style using the ‘document/literal’ style. Some toolkits recognize this similarity and generate RPC-style method signatures in stub classes, which is easier for developers to program with.

2.6.4.3 Example: WSDL definition

The following example shows the use of a wrapper element in a WSDL document. The SOAP body refers to a complex type with a name that is the same as the operation’s name.

WSDL definition

```
<wsdl:definitions
  targetNamespace=
    "http://ns.example.com/example/Svc/ProfileExample/1.0"
  xmlns:tns=
    "http://ns.example.com/example/Svc/ProfileExample/1.0"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <xsd:schema ...>
      <xsd:element name="sendDS">
        <xsd:complexType ... />
      </xsd:element>
      <xsd:element name="sendDSResponse">
        <xsd:complexType ... />
      </xsd:element>
    </xsd:schema>
  </wsdl:types>

  <wsdl:message name="sendDSInputMessage">
    <wsdl:part name="parameters" element="tns:sendDS"/>
  </wsdl:message>
  <wsdl:message name="sendDSOutputMessage">
    <wsdl:part name="parameters" element="tns:sendDSResponse"/>
  </wsdl:message>

  <wsdl:portType name="...">
    <wsdl:operation name="sendDS">
      <wsdl:input message="tns:sendDSInputMessage"/>
      <wsdl:output message="tns:sendDSOutputMessage"/>
    </wsdl:operation>
  </wsdl:portType>
  ...
</wsdl:definitions>
```

2.6.5 Request-Response message exchange pattern (MEP)

2.6.5.1 Conformance point on service interface specifications

The following conformance point applies:

Service interface specifications shall use the SOAP Request-Response Message Exchange Pattern.

2.6.5.2 Informative notes

SOAP Message Exchange Patterns are introduced in Section 3.2 of SOAP2007b and the Request-Response SOAP Message Exchange Patterns is defined in Section 6.2 of SOAP2007c.

The SOAP Request-Response Message Exchange Pattern requires that an input and an output message type be provided for the operation type in the WSDL definition. Even if the request has no parameters, it needs to send a SOAP request message with a body containing just the wrapper element (see Clause 2.6.4) that contains an empty content model. Service providers need to generate a SOAP response for every request they receive.

The SOAP Request-Response Message Exchange Pattern is generally well supported by Web service toolkits. The other patterns (i.e. one-way, solicit response and notification) are generally less mature and increase the risk of interoperability problems. For example, some toolkits use a different MIME type for the HTTP response with the one-way message exchange pattern, which can cause clients to reject the response or not process it properly.

2.6.5.3 Example: WSDL definition

The following example shows the use of the SOAP Request-Response MEP for every operation. Every operation has both an input and an output message.

```
<?xml version="1.0"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap12/"
  xmlns:tns=
    "http://ns.example.com/example/Svc/ProfileExample/1.0"
  targetNamespace=
    "http://ns.example.com/example/Svc/ProfileExample/1.0"
  name="DischargeSummaryReceiver">
  ...

  <!-- tns:SendDischargeSummaryOutMsg is a dummy message -->

  <portType name="DischargeSummaryReceiver">
    <operation name="ping">
      <input message="tns:pingInMsg" .../>
      <output message="tns:pingOutMsg" .../>
    </operation>

    <operation name="sendDischargeSummary">
      <input message="tns:sendDischargeSummaryInMsg" .../>
      <output message="tns:sendDischargeSummaryOutMsg" .../>
      <fault name="..." message="tns:... " .../>
    </operation>

    <operation name="checkStatus">
      <input message="tns:checkStatusInMsg" .../>
      <output message="tns:checkStatusOutMsg" .../>
      <fault name="..." message="tns:... " .../>
    </operation>
    ...
  </portType>
  ...
</definitions>
```