Nationwide Health Information Network (NHIN)

Trial Implementations

Service Interface Specifications

Messaging Platform

V .1.9.1

01/30/2009

Contributors

Name	Organization	Area
Neel Phadke	CareSpark	Specification
Vinod Muralidhar		Specification
Taha Abdul-Basser	MedVirginia	Specification
Ashish Shah	Delaware	Specification
Craig Miller	NHIN-C	Specification
Erik Rolf	CareSpark	Specification
Thomas Silvious	New York	Specification
David L. Riley	Fed NHIE	Specification

Document Change History

Version	Date	Changed By	Items Changed Since Previous Version
0.1		Craig Miller	Initial Draft
1.0		M&C Subgroup	Updates
1.1		M&C Subgroup	Updates
1.2		M&C Subgroup	Updates
1.3		M&C Subgroup	Updates
1.4		M&C Subgroup	Sub group feedback incorporated
1.5		M&C Subgroup	Updates
1.6		Craig Miller	Updates
1.7		Neel Phadke	Formatting
1.8		Craig Miller	Updates based on T&S Workgroup feedback
1.8.1		Deborah Lafky	Formatting, preparation for HITSP review
1.9		Craig Miller	Provided explanatory text re SOAP versioning, included specification for SOAP messaging style and SOAP faults, included specification for WS-Base Notification
1.9.1	01/30/2009	David L. Riley	Re-formatted and edited to prepare for publication

Document Approval

Version	Date	Approved By	Role

Table of Contents

1 Preface

1.1 Introduction

The NHIN Trial Implementations Service Interface Specifications constitute the core services of an operational Nationwide Health Information Network. They are intended to provide a standard set of service interfaces that enable Nationwide Health Information Exchange (NHIE) to NHIE exchange of interoperable health information. These services provide such functional capabilities as patient look-up, document query and retrieve, notification of consumer preferences, and access to logs for determining who has accessed what records and for what purpose for use. These functional services rest on a foundational set of messaging and security services. The current set of defined core services includes the following:

1. NHIN Trial Implementations Message Platform Service Interface Specification,
2. NHIN Trial Implementations Authorization Framework Service Interface Specification,
3. NHIN Trial Implementations Subject Discovery Service Interface Specification,
4. NHIN Trial Implementations Query for Documents Service Interface Specification,
5. NHIN Trial Implementations Document Retrieve Service Interface Specification,
6. NHIN Trial Implementations Audit Log Query Service Interface Specification,
7. NHIN Trial Implementations Consumer Preferences Service Interface Specification
8. NHIN Trial Implementations Health Information Event Messaging Service Interface Specification
9. NHIN Trial Implementations NHIE Service Registry Interface Specification
10. NHIN Trial Implementations Authorized Case Follow-Up Service Interface Specification

It is expected that these core services will be implemented together as a suite since the functional level services are dependent on the foundational services. Specifications #1 through #7 were the focus of the August 2008 testing event and September AHIC demonstrations. Specifications #1 through #9 were included in the November testing and demonstrations during the December 2008 NHIN Trial Implementations Forum.

1.2 Intended Audience

The primary audience for the NHIN Trial Implementations Service Interface Specifications is the individuals responsible for implementing software solutions that realize these interfaces for a NHIE. After reading this specification, one should have an understanding of the context in which the service interface is meant to be used, the behavior of the interface, the Web Services Description Language (WSDLs) used to define the service, any Extensible Markup Language (XML) schemas used to define the content and what “compliance” means from an implementation testing perspective.

1.3 Focus of this Specification

This document presents the NHIN Trial Implementations Messaging Platform Service Interface Specification. In order for the NHIN to work, messages exchanged between participating NHIEs must adhere to a well known and agreed to set of standard specifications. This document identifies the base set of messaging standards that must be implemented by each NHIE in order to securely exchange interoperable health information on the NHIN.

1.3.1 Business Needs Supported by this Specification

This specification is designed to address an organization’s need to exchange interoperable health information with other organizations using health information in the provision of services where those organizations are in separate security domains.

1.3.2 Key Attributes of the Specification

There are three fundamental assumptions that drive this specification:

1. The NHIN is a web-services based service oriented architecture and therefore every message between NHIEs on the NHIN is a SOAP message over HTTP.
2. PKI is used to establish a technical “trust fabric” for the NHIN

3. All inter-NHIE messages (gateway to gateway communication) in the NHIN are HL7 v 3.0 RIM based messages.

1.3.3 Scenarios for Use

This specification in combination with the NHIN Trial Implementations Authorization Framework Service Interface Specification forms the foundation of the NHIN. All other service specifications for inter-NHIE communication over the NHIN assume the details of these two specifications are implemented as a part of their technical pre-conditions.

1.4 Definitions

None

1.5 Related Documents

The following documents and standards were referenced during the development of this specification:

- WS-I Basic Profile 1.2
  - http://www.ws-i.org/Profiles/BasicProfile-1.2.html
- WS-I Security Profile 1.1
  - http://www.ws-i.org/Profiles/BasicSecurityProfile-1.1.html

1.6 Relationship to HITSP Constructs

A separate document has been developed as an Excel spreadsheet that maps the specifications within this document to their origins within the relevant HITSP and IHE technical specifications. This document is available on the NHIN portal for download under the name “Standards Mapping for Messaging Specifications.xls”.

1.7 Relationship to other NHIN Cooperative Specifications

The NHIN Trial Implementations Messaging Platform Service Interface Specification implemented with the NHIN Trial Implementations Authorization Framework Service Interface Specification are the foundation of the NHIN for any organization wanting connect to the NHIN must at a minimum implement these two specifications.

The following NHIN Trial Implementations Service Interface Specifications assume and are implemented on the foundation of the NHIN Trial Implementations Messaging Platform Service Interface Specification and the NHIN Trial Implementations Authorization Framework Service Interface Specification:

1. NHIN Trial Implementations Subject Discovery Service Interface Specification,
2. NHIN Trial Implementations Query for Documents Service Interface Specification,
3. NHIN Trial Implementations Document Retrieve Service Interface Specification,
4. NHIN Trial Implementations Audit Log Query Service Interface Specification,
5. NHIN Trial Implementations Consumer Preferences Service Interface Specification
6. NHIN Trial Implementations Health Information Event Messaging Service Interface Specification
7. NHIN Trial Implementations NHIE Service Registry Interface Specification
8. NHIN Trial Implementations Authorized Case Follow-Up Service Interface Specification

2 Interface Description

2.1 Requirements

The Platform is based on the following architectural principles:

1. There shall be a common transport layer for all messages. For the NHIN to be a truly scalable, secure and interoperable network, a common transport layer is essential. The Messaging Platform will be based on SOAP 1.1 messages over HTTP. All higher-level messaging elements and attachments must be bound to a SOAP message. This excludes the use of incompatible transport protocols such as HL7 MLLP for NHIE to NHIE messaging.
2. Messages between NHIEs must be secure. This will necessitate the use of encryption as part of the message transport layer.
3. The common message envelope must support assertions about security and trust between NHIEs.
4. The basis for authentication for NHIE participants shall be X.509 certificates. All NHIE certificates for NHIN Trial Implementations will be issued by a common trusted certificate authority yet to be determined. All NHIE to NHIE messages must be digitally signed for purposes of authentication and non-repudiation.
5. The NHIN shall be based on interoperability profiles that have been fully approved as an industry interoperability standard and are capable of being implemented by NHIN Trial Implementations using available SOA platforms.

2.2 Web Services

The proposed NHIN Messaging Platform is based on the use of web services, as articulated within interoperability profiles established by the Web Services Interoperability Organization (WS-I). Collectively, the WS-I Basic and Basic Security Profiles define a common platform for secure and reliable exchange of messages between NHIEs in a manner that is independent of specific operating system or programming language frameworks.

Briefly stated, the Messaging Platform describes the common web service protocols that should underlie every message transmitted between NHIEs. These specifications represent a common platform for all other NHIN Interface Specifications. The Messaging Platform specification utilizes a single version of all relevant standards to ensure interoperability and consistency, although this issue may be revisited if required after the 2008 Trial Implementations if necessary.

Based on the requirements expressed above, the following messaging platform has been selected for the 2008 NHIN Trial Implementations:

2.2.1 Basic Profile:

WS-I Basic Profile 1.2
http://www.ws-i.org/Profiles/BasicProfile-1.2.html

Specification	Version	Notes
Simple Object Access Protocol (SOAP)	1.1	SOAP 1.1 will be utilized for the 2008 Trial Implementations. It is anticipated that the NHIN will migrate to SOAP 1.2 for 2009 and beyond.
SOAP Message Encoding Style		Document Literal
SOAP Faults		No custom SOAP faults are required.
Hypertext Transfer Protocol (HTTP)	1.1	
WS-Addressing	1.0	
WS-BaseNotification	1.3	
Message Transmission Optimization Mechanism (MTOM) binding for SOAP 1.1	1.0	
Web Services Description Language (WSDL)	1.1	
XML Schema	1.0	
Universal Discovery and Description Interface (UDDI)	2.0.4	

2.2.2 Security Profile:

WS-I Security Profile 1.1
http://www.ws-i.org/Profiles/BasicSecurityProfile-1.1.html

Specification	Version	Notes
Transport Layer Security	1.0	Note that this is equivalent to Secure Sockets Layer 3.0
XML Signature	1.0	
Web Services Description Language (WSDL)	1.1	
Symmetric Encryption Algorithm and Key Length	AES 128-bit	AES = Advanced Encryption Standard
X.509 Token Profile	1.0	
SAML Token Profile	1.1	Note that SAML Token Profile 1.1 mandates the use of the SAML 2.0 Base Specification
Attachment Security	1.0	

2.3 Consistent Time

Consistent Time provides a means to ensure that the system clocks and time stamps of the many computers in a network are well synchronized. This profile specifies synchronization with a median error less than 1 second. This is sufficient for most purposes.

HITSP T16 specifies the use of IHE ITI-TF Revision 4.0 or later, Consistent Time (CT) Integration Profile. The Consistent Time Integration Profile defines mechanisms to synchronize the time base between multiple actors and computers. Various infrastructure, security, and acquisition profiles require use of a consistent time base on multiple computers. The Consistent Time profile requires the use of the Network Time Protocol (NTP) defined in RFC 1305. When the Time Server is grouped with a Time Client to obtain time from a higher tier Time Server, the Time Client shall utilize NTP. For some Time Clients that are not grouped with a Time Server, SNTP may be usable.

Base specifications are as follows:

- IETF NTP (Version 3) Specification, Implementation and Analysis RFC #1305, March, 1992
- IETF Simple Network Time Protocol (SNTP) Version 4, RFC #2030, October, 1996

2.4 Public Key Infrastructure

2.4.1 Certificate Authority for the NHIN Trial Implementations

For the September and December 2008 Trial Implementation demonstrations, a single certificate authority (CA) will be established for this purpose to issue X.509 certificates to all NHIEs. The fact that all certificates are signed by the common designated CA will serve as the basis of trust and authentication between NHIEs; all messages between NHIEs are both signed and encrypted using these certificates.

For the 2008 demonstrations, we will not implement more complex PKI functionality such as chaining certificate authorities or certificate revocation, for example. However, this functionality will clearly need to be addressed beyond the Trial Implementation period.

The Trial Implementations will not assume a centralized identity provider exists for identity proofing and credentialing.

2.4.2 Digitally Signing SAML Assertions

The following information is provided as guidance for digitally signing SAML Assertions

2.4.2.1 Policy Definition

Within the WSDL defining an available Web Service, the recipient of the HTTP request has opportunity to assert the various WS-SP policies that define the security expectations that control access to the requested service. It is within this policy that the use of a secure transport is described as well as the use of a selected algorithm suite for use in the digital signature and for encryption purposes. In addition a supporting token is defined that distinguishes the various types of SAML Token Authentication.

Mutual authentication, TLS, is supported through the definition of the sp:TransportBinding policy assertion. The following sp:TransportToken declares that the secure transport will be over Https via the sp:HttpsToken assertion, and it declares mutual authentication via the sp:RequireClientCertificate assertion.

```
<sp:TransportToken>
        <wsp:Policy>
                <sp:HttpsToken>
                        <wsp:Policy>
                                <sp:RequireClientCertificate/>
                        </wsp:Policy>
                </sp:HttpsToken>
        </wsp:Policy>
</sp:TransportToken>
```

The cipher suite recommendation from the OASIS committee for SAML2.0 over the TLS protocol was for forward looking applications to implement TLS_RSA_WITH_AES_128_CBC_SHA, which has both speed and security strength.[1] This translates to the Basic128 algorithm suite as defined in the WS-SecurityPolicy[2] and is declared within the sp:AlgorithmSuite assertion as shown in the following example.

```
<sp:AlgorithmSuite>
        <wsp:Policy>
                <sp:Basic128/>
        </wsp:Policy>
</sp:AlgorithmSuite>
```

The SAML2.0 Holder-of-Key Assertion is declared differently dependent upon the defined binding. In the event of Transport Binding which declares the TLS case, the SAML2.0 Holder-of-Key appears as an sp:EndorsingSupportingToken. Endorsing tokens can also be used to define additional message parts to sign and/or encrypt; however, these are not used at this time. When transport security is defined as in this case, the signature must also cover the message timestamp. The sp:SamlToken distinguishes that this is a SAML Token assertion that will always expect the token to be included on messages sent to the recipient. It is the WssSamlV20Token11 that actually declares this as Saml version 2.0. The following example provides an sp:EndorsingSupportingToken.

```
<sp:EndorsingSupportingTokens>
        <wsp:Policy>
                <sp:SamlToken sp:IncludeToken = " http://docs.oasis-open.org/ws-sx/ws-
securitypolicy/200702/IncludeToken/AlwaysToRecipient">
                        <wsp:Policy>
                                <sp:WssSamlV20Token11/>
                        </wsp:Policy>
                </sp:SamlToken>
```

[1] Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0

[2] WS-SecurityPolicy 1.3

```
        </wsp:Policy>
</sp:EndorsingSupportingTokens>
```

Putting this all together we have the policy requirements to support the declaration of a SAML2.0 assertion over TLS.[3]

```
<sp:TransportBinding>
    <wsp:Policy>
        <sp:TransportToken>
            <wsp:Policy>
                <sp:HttpsToken>
                    <wsp:Policy>
                        <sp:RequireClientCertificate/>
                    </wsp:Policy>
                </sp:HttpsToken>
            </wsp:Policy>
        </sp:TransportToken>
        <sp:Layout>
            <wsp:Policy>
                <sp:Strict/>
            </wsp:Policy>
        </sp:Layout>
        <sp:IncludeTimestamp/>
        <sp:AlgorithmSuite>
            <wsp:Policy>
                <sp:Basic128/>
            </wsp:Policy>
        </sp:AlgorithmSuite>
    </wsp:Policy>
</sp:TransportBinding>
<sp:EndorsingSupportingTokens>
    <wsp:Policy>
        <sp:SamlToken sp:IncludeToken = " http://docs.oasis-open.org/ws-sx/ws-
securitypolicy/200702/IncludeToken/AlwaysToRecipient">
            <wsp:Policy>
                <sp:WssSamlV20Token11/>
            </wsp:Policy>
        </sp:SamlToken>
    </wsp:Policy>
</sp:EndorsingSupportingTokens>
```

2.4.2.2 Digital Signature Specification

XML Digital Signatures are applied to data objects through an indirection or URI reference. The data object is digested and that digest value is placed within the signature's <ds:DigestValue> element which is cryptographically signed. The table below discusses the elements and attributes for the <ds:Signature> element.

Element/Attribute	Usage	Description
@Id	Optional	Identifies this signature for possible later reference
SignedInfo	Required	Contains the definition of the Canonicalization method, the Signature method, and the reference to

[3] WS-SecurityPolicy Examples

		the object being signed.
SignatureValue	Required	Contains the actual value of the digital signature
KeyInfo	Optional (Required for NHIN)	Contains the information such that the recipient can validate the signature. If this is omitted then the recipient is expected to be able to identify the key by some other means. For the current NHIN set-up this should be used to convey this information.
Object ID	Optional	Optional element that may contain other data such as MIME type, an ID, or Encoding attributes.

The <ds:SignedInfo> element is rather like a container for the following elements and attributes.

Element/Attribute	Usage	Description
@Id	Optional	Identifies this element for possible later reference
Canonicalization Method	Required	Must support the required canonicalization algorithms. It is recommended that Exclusive Canonicalization be used.[4] http://www.w3.org/2001/10/xml-exc-c14n#
SignatureMethod	Required	Identifies the cryptographic functions involved in the signature operation. It is recommended that SAML processors support the use of RSA signing and verification.[4] http://www.w3.org/2000/09/xmldsig#rsa-sha1
Reference	Required	This must contain the URI of that which is being signed. For example this would contain the Id of the Assertion element when signing that element, or the Id of the <wsu:Timestamp> when signing that element.

The <ds:Reference> element specifies the digest algorithm the digest method and what is being signed. For SAML2.0 there is this single contained <ds:Reference> element. It has the following elements and attributes.

Element/Attribute	Usage	Description
@Id	Optional	Identifies this element for possible later reference
@URI	Optional (Required for SAML2.0)	For SAML2.0 this must identify the object being signed using that elements Id.[4]
@Type	Optional	Identifies the type of object being signed.
Transforms	Optional	This is an ordered list of transformations that took place on the data object. This is only allowed to contain a subset of enveloped signature transform, exclusive canonicalization transform, and exclusive canonicalization with comments.[4]

[4] Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0

		http://www.w3.org/2000/09/xmldsig#enveloped-signature) http://www.w3.org/2001/10/xml-exc-c14n# http://www.w3.org/2001/10/xml-exc-c14n#WithComments).
DigestMethod	Required	Defines the digest algorithm that is applied. For the Basic128 Algorithm Suite this is SHA1. http://www.w3.org/2000/09/xmldsig#sha1
DigestValue	Required	This is the encoded value of the digest.

The <ds:KeyInfo> element provides the means by which the signature is validated. There are many possible children such as PGPData, SPKIData, X509Data, etc. This document will present one possible simple implementation for validating the signature of the Assertion Token by defining only the <ds:KeyValue> element. The KeyValue element contains a single public key that can be used to validate the signature. These are structured formats for defining the DSA (Digital Signature Algorithm) with a recommendation to use the RSA. The <ds:RSAKeyValue> has the following elements:

Element/Attribute	Usage	Description
Modulus	Required	This is a prime modulus used in the DSA.
Exponent	Required	This is the exponent term.

These arbitrary-length integers are represented as octet strings as defined by the ds:CryptoBinary type which is a base64Binary.

The following is an example of the digital signature given the above requirements.

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:SignedInfo>
                <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
                <ds:Reference URI="#51cb7689-0957-46a2-938e-1add75577ab7">
                        <ds:Transforms>
                                <ds:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                                <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                        </ds:Transforms>
                        <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                        <ds:DigestValue>a3XVN23H2N/ga+08AGqGHD1euKc=</ds:DigestValue>
                </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>L8Liyz+6pLwNP9YBfIRbrDVUJtM2YcLuN3+HPjspQEHmZ2uTXWYuy7XTM
9dqmN93w0ypVM7egjRe=</ds:SignatureValue>
        <ds:KeyInfo>
                <ds:KeyValue>
                        <ds:RSAKeyValue>

        <ds:Modulus>vYxVZKIzVdGMSBkW4bYnV80MV/RgQKV1bf/DoMTX8laMO45P6=</ds:Modulus>
                                <ds:Exponent>AQAB</ds:Exponent>
                        </ds:RSAKeyValue>
                </ds:KeyValue>
        </ds:KeyInfo>
</ds:Signature>
```

When validating the signing of the timestamp, the <ds:KeyInfo> element will contain the <wsse:SecurityTokenReference>[5] as shown in this following example:

```
<ds:KeyInfo>
        <wsse:SecurityTokenReference wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV2.0">
                <wsse:KeyIdentifier ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLID">51cb7689-0957-46a2-938e-1add75577ab7</wsse:KeyIdentifier>
        </wsse:SecurityTokenReference>
</ds:KeyInfo>
```

The following reference provides more information on the syntax and processing of digital signatures: W3C XML Signature Syntax and Processing

2.4.2.3 Subject Confirmation

As part of the validation and processing of the assertion, the receiver must establish the relationship between the subject and claims of the SAML statements and the entity providing the evidence to satisfy the confirmation method defined for the statement. Statements attested for by the holder-of-key method must be associated with one or more holder-of-key SubjectConfirmation elements. The SubjectConfirmation elements must include a <ds:KeyInfo> element that identifies a public key that can be used to confirm the identity of the subject.[6] The SubjectConfirmation element has the following elements and attributes.

Element/Attribute	Usage	Description
@Method	Required	A URI reference that identifies a protocol or mechanism to be used to confirm the subject. For Holder-of-Key this is: urn:oasis:names:tc:SAML:2.0:cm:holder-of-key
BaseId NameId EncryptedId	Optional	Identifies the entity expected to satisfy the enclosing subject confirmation requirements.
SubjectConfirmationData	Optional	However, as this contains the KeyInfo element it is required for Holder-of-Key.

The SubjectConfirmationData element specifies additional data that allows the subject to be confirmed. This can include a variety of optional attributes: NotBefore, NotOnOrAfter, Recipient, InResponseTo, and Address. However, it is the <ds:KeyInfo> element that identifies the cryptographic key that is used to authenticate the attesting entity.[7]

The following is an example of the subject confirmation element given the above requirements.

```
<saml2:Subject>
        <saml2:NameID
    Format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName">
                CN=Alex G. Bell,O=1.22.333.4444,UID=abell
        </saml2:NameID>
        <saml2:SubjectConfirmation
          Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
```

[5] WS-SecurityPolicy – Appendix C

[6] Web Services Security: SAML Token Profile 1.1

[7] Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0

```
                <saml2:SubjectConfirmationData>
                        <ds:KeyInfo>
                                <ds:KeyValue>
                                        <ds:RSAKeyValue>

        <ds:Modulus>vYxVZKIzVdGMSBkW4bYnV80MV/RgQKV1bf/D

X8laMO45P6rlEarxQiOYrgzuYp+snzz2XM0S6o3JGQtXQ=
                                                        <ds:Modulus>
                                                <ds:Exponent>AQAB</ds:Exponent>
                                        </ds:RSAKeyValue>
                                </ds:KeyValue>
                        </ds:KeyInfo>
                </saml2:SubjectConfirmationData>
        </saml2:SubjectConfirmation>
</saml2:Subject>
```

3 Interface Definition

3.1 Message Syntax

All messages on the NHIN between NHIEs are SOAP messages and therefore implement the SOAP syntax.

3.2 Namespaces

The messages defined in the NHIN Trial Implementations Service Interface Specifications utilize elements defined in several different places, as described in the following table. Readers and implementers are urged to pay careful attention to the XML namespaces used in the various NHIN Trial Implementations Service Interface Specifications.

Namespace prefix used in this document	Full namespace identifier	Description
wsnt	http://docs.oasis-open.org/wsn/b-2	WS-Base Notification standard
wsa	http://www.w3.org/2005/08/addressing	WS-Addressing standard
nhin	http://www.hhs.gov/healthit/nhin	A NHIN defined namespace
xacml	urn:oasis:names:tc:xacml:2.0:policy:schema:os	XACML standard
rim	urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0	ebXML Registry Information Model
ihe	urn:ihe:iti:xds-b:2007	IHE XDS.b schema
rs	urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0	Registry

4 Error Handling

No custom SOAP faults are required by this specification. Appendix A contains a sample SOAP Fault Message.

Each of the NHIN Trial Implementations Service Interface Specifications contains a section on error handling for describing the constraints and extensions on the base specifications they use for faults. The reader is directed to the particular specification for those details.

5 Auditing

When an NHIE should create an “Export” audit event or an “Import” audit event when sending or receiving messages to another NHIE is detailed in the various service interface specifications and the reader is directed to the particular specification for those details.

6 Potential Future Considerations

6.1 Other Messaging Platform Standards

The Security and Technical Working Group intends to adopt web service reliable messaging standards in a future revision of this document, pending an evaluation of the maturity of current implementations, harmonization of the current competing standards for reliable messaging, and a clear understanding of the specific requirements for reliability as expressed within AHIC use cases.

7 Appendix A - Sample SOAP Messages

7.1 SOAP Request

```
- <soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:wsse="http://docs.oasis-
    open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
    xmlns:wsu="http://docs.oasis-open.org//wss/2004/01/oasis-200401-wsswssecurity-
    utility-1.0.xsd" xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
- <soapenv:Header>
  - <!--
   MessageID, To and Action are required elements
    -->
  - <!--
   Unique identifier of this message
    -->
  <wsa:MessageID>urn:uuid:0fbfdced-6c01-4d09-a110-2201afedaa02</wsa:MessageID>
  - <!--
   URI of the service
    -->
  <wsa:To
    soapenv:mustUnderstand="1">http://stelsewhere.com/XdsService/IHEXDSRepository.svc
    </wsa:To>
  - <!--
   URI indicates specific action that is requested to be performed by the service
    -->
  - <!--
   Same as To URI in HTTP requests
    -->
  - <!--
   In a non-HTTP request, To and Action may be different, with Action pointing to the
    WSDL PortType
    -->
  <wsa:Action
    soapenv:mustUnderstand="1">urn:ihe:iti:2007:RetrieveDocumentSet</wsa:Action>
  - <!--
   URI of the requester
    -->
  <wsa:From>http://http://generalhospital.org/nhiegateway/getDocs</wsa:From>
  - <!--
   URI to which the response should be sent
    -->
  - <!--
   If ReplyTo is not the same as From, treat as an Asynchronous req/response
    -->
  - <!--
   If ReplyTo is the same as From, or is Anonymous, or omitted treat as a Synchronous
    request
    -->
- <wsa:ReplyTo>
  <wsa:Address>http://generalhospital.org/nhiegateway/getDocs</wsa:Address>
    </wsa:ReplyTo>
- <wsse:Security soapenv:mustUnderstand="true">
- <wsse:UsernameToken>
  - <!--
   For Audit logging purposes, a unique Username from the Requesting Organization /
    NHIE is also sent
```

```
    -->
  <wsse:Username>John Doe MD, General Hospital</wsse:Username>
    </wsse:UsernameToken>
  <wsse:BinarySecurityToken wsu:Id="X509Token" ValueType="wsse:X509v3"
     EncodingType="wsse:Base64Binary">MIIOlskew78Hjkds...</wsse:BinarySecurityToken>
   - <!--
    Timestamp of the Security element
    -->
   - <!--
    Often digitally signed to prevent replay attacks
    -->
- <wsu:Timestamp wsu:Id="MesgTimeStamp">
  <wsu:Created>2008-03-14T15:42:00Z</wsu:Created>
  <wsu:Expires>2008-03-14T16:00:00Z</wsu:Expires>
    </wsu:Timestamp>

   - <!--
    Digital Signature used to sign parts of this document
    -->
- <ds:Signature>
- <ds:SignedInfo>
  <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
  <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
- <ds:Reference URI="#MesgTimeStamp">
- <ds:Transforms>
  <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
    </ds:Transforms>
  <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
  <ds:DigestValue>LyLsF094hPi4wPU...</ds:DigestValue>
    </ds:Reference>
- <ds:Reference URI="#nhinMesg">
- <ds:Transforms>
  <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
    </ds:Transforms>
  <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
  <ds:DigestValue>LyLsF094hPi4wPU...</ds:DigestValue>
    </ds:Reference>
    </ds:SignedInfo>
  <ds:SignatureValue>Hp1ZkmFZ/2kQLXDJbchm5gK...</ds:SignatureValue>
- <ds:KeyInfo>
- <wsse:SecurityTokenReference>
  <wsse:Reference URI="#X509Token" />
    </wsse:SecurityTokenReference>
    </ds:KeyInfo>
    </ds:Signature>
    </wsse:Security>
    </soapenv:Header>
- <soapenv:Body>
- <reqMesg wsu:Id="nhinMesg">

    </reqMesg>
    </soapenv:Body>
    </soapenv:Envelope>
```

7.2 SOAP Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://www.w3.org/2005/08/addressing">
- <soapenv:Header>
  - <!--
   MessageID, To and Action are required elements
    -->
  - <!--
   Unique identifier of this message
    -->
  <wsa:MessageID>urn:uuid:f0dedced-6c01-4d09-a110-2201afedf0f0</wsa:MessageID>
  - <!--
   Requester URI
    -->
  <wsa:To
    soapenv:mustUnderstand="1">http://http://generalhospital.org/nhiegateway</wsa:To>
  - <!--
   Same as To URI in HTTP requests
    -->
  - <!--
   In a non-HTTP request, To and Action may be different, with Action pointing to the
    WSDL PortType
    -->
  <wsa:Action
    soapenv:mustUnderstand="1">urn:ihe:iti:2007:RetrieveDocumentSetResponse</wsa:Actio
    n>
  - <!--
   Unique identifier of the original message to which this is a response
    -->
  <wsa:RelatesTo>urn:uuid:0fbfdced-6c01-4d09-a110-2201afedaa02</wsa:RelatesTo>
    </soapenv:Header>
- <soapenv:Body>
  <respMesg>RESPONSE Document ...</respMesg>
    </soapenv:Body>
    </soapenv:Envelope>
```

7.3 SOAP Fault

```
- <soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope/"
    xmlns:wsa="http://www.w3.org/2005/08/addressing">
- <soapenv:Header>
  - <!--
   MessageID, To and Action are required elements
    -->
  - <!--
   Unique identifier of this message
    -->
  <wsa:MessageID>urn:uuid:f0dedced-6c01-4d09-a110-2201afedf0f0</wsa:MessageID>
  - <!--
   Requester URI
    -->
  <wsa:To
    soapenv:mustUnderstand="1">http://http://generalhospital.org/nhiegateway</wsa:To>
  - <!--
   Same as To URI in HTTP requests
    -->
  - <!--
```

```
    In a non-HTTP request, To and Action may be different, with Action pointing to the
     WSDL PortType
     -->
   <wsa:Action
      soapenv:mustUnderstand="1">urn:ihe:iti:2007:RetrieveDocumentFault</wsa:Action>
   - <!--
    Unique identifier of the original message to which this is a Fault response
     -->
   <wsa:RelatesTo>urn:uuid:0fbfdced-6c01-4d09-a110-2201afedaa02</wsa:RelatesTo>
     </soapenv:Header>
- <soapenv:Body>
- <soapenv:Fault>
- <soapenv:Code>
   <soapenv:Value>env:Client</soapenv:Value>
     </soapenv:Code>
- <soapenv:Reason>
   <soapenv:Text xml:lang="en">There was an error in the incoming SOAP
      request</soapenv:Text>
     </soapenv:Reason>
     </soapenv:Fault>
     </soapenv:Body>
     </soapenv:Envelope>
```