

nehta

Connectivity

Implementation Guide

Version 1.0 — 1 December 2008

Release

National E-Health Transition Authority Ltd

Level 25

56 Pitt Street

Sydney, NSW, 2000

Australia.

www.nehta.gov.au**Disclaimer**

NEHTA makes the information and other material (“Information”) in this document available in good faith but without any representation or warranty as to its accuracy or completeness. NEHTA cannot accept any responsibility for the consequences of any use of the Information. As the Information is of a general nature only, it is up to any person using or relying on the Information to ensure that it is accurate, complete and suitable for the circumstances of its use.

Document Control

This document is maintained in electronic form. The current revision of this document is located on the NEHTA Web site and is uncontrolled in printed form. It is the responsibility of the user to verify that this copy is of the latest revision.

Copyright © 2008, NEHTA.

This document contains information which is protected by copyright. All Rights Reserved. No part of this work may be reproduced or used in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems—without the permission of NEHTA. All copies of this document must include the copyright and other information contained on this page.

Table of contents

Table of contents	iii
Document information	iv
Change history	iv
1 Introduction	1
1.1 Document purpose	1
1.2 Intended audience	1
1.3 Document map	1
1.4 Definitions, acronyms, abbreviations	2
1.5 Normative references	2
1.6 Overview	2
2 Implementing Without the National Infrastructure	3
2.1 Unique Healthcare Identifiers	3
2.1.1 Alternatives to IHI	3
2.1.2 Alternatives to HPI	3
2.2 Authentication	4
2.2.1 Alternatives to NASH	4
2.3 Service instance Locator	4
2.3.1 Pre-SIL deployments	4
2.3.2 Alternatives to the Base-SIL	5
3 Performance and reliability	6
3.1 Performance	6
3.1.1 Caching	6
3.2 Reliability	6
3.2.1 Retries	6
3.2.2 Escalation	7
3.3 Auditing and tracking	7
3.3.1 Tracking	7
3.3.2 Auditing	7
4 Security	9
4.1 Security protocol	9
4.1.1 Transport Layer Security (TLS)	9
4.2 Service invokers	9
4.2.1 Firewalls	9
4.3 Service providers	9
4.3.1 Firewalls	9
4.3.2 DMZ	10
5 Deployment	11
5.1 Service Instance Locator (SIL)	11
5.1.1 Automatic SIL updating	11
5.2 Integration	11
5.2.1 Integration with proprietary systems	11
Appendix A: Informative references	13

Document information

Change history

Version	Date	Comments
1.0	2008-12-01	Release

1 Introduction

1.1 Document purpose

This document provides guidance on how to design and implement systems which follow the connectivity architecture and its related specifications.

This is a non-normative document. It clarifies certain aspects of the connectivity specification and offers suggested approaches for handling common issues. Implementations can, but do not have to, use this document.

1.2 Intended audience

This is a technical document.

This document is intended for:

- Enterprise architects who develop strategic plans for solutions that involve connectivity.
- Solution architects who develop solutions that implement or interact with the connectivity architecture.
- Software developers who implement the solutions designed by the solution architects.

The reader is expected to be familiar with the *Connectivity: Architecture* document [CA2008].

1.3 Document map

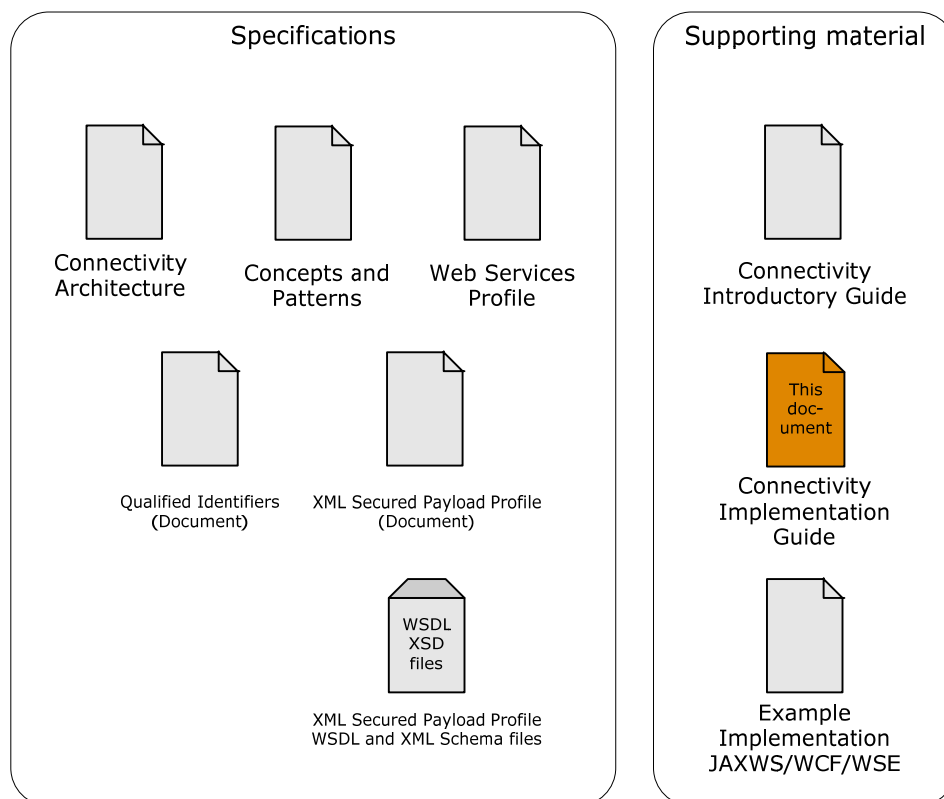


Figure 1: Connectivity document map

1.4 Definitions, acronyms, abbreviations

HPI	Healthcare Provider Identifier
IHI	Individual Healthcare Identifier
NASH	National Authentication Service for Health
NEHTA	National E-Health Transition Authority
SIL	Service Instance Locator
TLS	Transport Layer Security

1.5 Normative references

The following NEHTA specifications and other references contain provisions which, through reference in this text, constitute provisions of this Specification. At the time of publication, the editions indicated were valid. All Specification and other references are subject to revision: all users of this Specification are therefore encouraged to investigate the possibility of applying the most recent edition of the Specification and other references listed below.

[CA2008]	NEHTA, <i>Connectivity: Architecture v1.0</i> , 1 December 2008.
[NIF2007]	NEHTA, <i>Interoperability Framework v2.0</i> , 2008.
[QI2008]	NEHTA, <i>Qualified Identifiers v1.0</i> , 1 December 2008.
[XSP2008]	NEHTA, <i>XML Secured Payload v1.0</i> , 1 December 2008.

1.6 Overview

This document provides guidelines on a number of topics. Although they are all related to connectivity, these topics are not necessarily related to each other. For ease of presentation, they are grouped into four chapters:

- Chapter 2 on implementing without the national infrastructure services topics.
- Chapter 3 on performance and reliability topics.
- Chapter 4 on security topics.
- Chapter 5 on deployment topics.

For each topic, the issue, recommendation and tradeoffs are described.

2 Implementing without the National Infrastructure

The national infrastructure services are a set of common services whose implementation is being managed by NEHTA. These services are being deployed according to their own timelines, so there will be an interim period when they are not all available for e-health interactions.

This chapter describes alternatives for the national infrastructure services. These can be used as temporary substitutes until the national service becomes available.

2.1 Unique Healthcare Identifiers

2.1.1 Alternatives to IHI

2.1.1.1 Issue

Implementing connectivity before the NEHTA Individual Healthcare Identifier (IHI) numbers and services are available.

2.1.1.2 Recommendation

Use local individual identifiers which are agreed upon between the communicating parties.

For example, a pathology laboratory can use their internal patient identifiers as long as the receiving General Practice can match those identifiers to the correct patient in their system. These should be represented however within a globally unique qualifier that identifies the type of identifier. This scheme should conform with the concepts in [QI2008].

Use local services to lookup and retrieve these identifiers. An external mechanism must be defined to ensure that the communicating parties can identify the same individual using the local identifier, or can map local identifiers from one system to another. Privacy and security concerns should be given serious consideration if these schemes are to be used.

2.1.1.3 Tradeoffs

Local identifiers will not be meaningful to all providers in the nation. This may limit the scope of where such identifiers can be used.

2.1.2 Alternatives to HPI

2.1.2.1 Issue

Implementing connectivity before the NEHTA Healthcare Provider Identifier (HPI) numbers and services are available.

2.1.2.2 Recommendation

Use local healthcare provider identifiers which are agreed upon between the communicating parties.

For example, a pathology laboratory can use their Australian Business Number (ABN) as long as the receiving General Practice can match those numbers to the laboratory.

Define a globally unique qualifier for that type of identifier, so that it can be uniquely represented as a qualified identifier [QI2008].

Use local services to lookup and retrieve these identifiers. An external mechanism must be defined to ensure that the communicating parties can identify the same individual using the local identifier, or can map local identifiers from one system to another.

2.1.2.3 Tradeoffs

Local identifiers will not be meaningful to all providers in the nation. This may limit the scope of where such identifiers can be used.

2.2 Authentication

2.2.1 Alternatives to NASH

2.2.1.1 Issues

Implementing connectivity before the NEHTA National Authentication for Health (NASH) PKI Authentication for Health (NASH) PKI and services are available.

2.2.1.2 Recommendation

Use certificates from an alternative source with agreement between the communicating parties.

For example, one possible source of certificates are Medicare Australia's HeSA PKI certificates.

2.2.1.3 Tradeoffs

Every entity that uses the certificates must be able to support the types of certificates being used. This involves supporting it technically (e.g. knowing how to check them for revocation) as well as supporting it in policy (e.g. trusting them).

Medicare Australia issues dual keys—one key pair for digital signing and a different key pair for encryption. This is different from the NEHTA Web Services Profile and the NEHTA NASH, which is designed around a single key environment—where one key pair can be used for both digital signing and encryption.

Implementations that use HeSA certificates will have to behave differently to support dual keys. They will have to be modified when the NASH becomes available.

Dual keys are not supported by Microsoft Windows Communication Foundation (WCF). Developers cannot implement an interim solution using Microsoft WCF and HeSA certificates. Also an interim solution that uses dual keys cannot communicate with programs implemented using Microsoft WCF.

2.3 Service instance Locator

2.3.1 Pre-SIL deployments

2.3.1.1 Issues

Implementing the core connectivity interaction without using a Service Instance Locator (SIL).

2.3.1.2 Recommendation

This situation should not occur, because a deployment of a NEHTA service should also include a deployment of the SIL.

In the absence of a SIL, the values of the SIL must be manually obtained and configured into the service invoker. For example, they could be interchanged as data files which are imported into the service invoker.

2.3.1.3 Tradeoffs

The SIL has been the problems of locating and using service instances in a distributed and dynamic environment. If the SIL is not used, these problems will have to be addressed manually or using proprietary directories.

Although manual methods can work in a small environment, it does not scale to a large environment. It might work during the early stages of deployment where there are a small number of service instances, but it will become impractical as more and more service instances are deployed.

2.3.2 Alternatives to the Base-SIL

2.3.2.1 Issues

The Base-SIL is a SIL that contains entries for the national infrastructure services.

Service invokers use the Base-SIL to obtain the address and identify the certificates for services such as the IHI service, HPI-I service, HPI-O service and NASH service.

2.3.2.2 Recommendation

In the absence of a Base-SIL, applications should be locally configured with the information they require. Instead of looking up the Base-SIL, they would obtain the information from a local configuration file or local database.

2.3.2.3 Tradeoffs

The local configuration file or local database will have to be manually updated when the information changes.

3 Performance and reliability

3.1 Performance

3.1.1 Caching

3.1.1.1 Issue

Invoking multiple services takes time and places a load on those services.

3.1.1.2 Recommendation

Cache data whenever it is possible.

Caching data means to save data that is obtained from a service for future use. If it is required again, the saved data is used instead of obtaining it from the service.

Service interface specifications should indicate what data can be cached and provide mechanisms to support caching whenever possible.

3.1.1.3 Tradeoffs

Caching consumes local storage space. Appropriate policies must be defined to remove data from the cache.

Only data that does not change frequently can be successfully cached. It is expected that some data from the IHI, HPI, NASH and SIL can be cached.

When data does change, mechanisms must be in place to ensure that the cached data is not used or does not cause problems if it is used.

3.2 Reliability

3.2.1 Retries

3.2.1.1 Issue

Sometimes a service instance is temporally unavailable.

For example, the service is undergoing maintenance or there is temporary network failure.

3.2.1.2 Recommendation

Service invokers can retry invoking the service instance at a later time. It can repeat this process if the retry also fails.

3.2.1.3 Tradeoffs

This is only suitable for batched or asynchronous processes, where an immediate result is not needed.

The service instance may be permanently unavailable, in which case retries will never succeed. Mechanisms must be in place to handle this situation.

If cached data is being used, mechanisms must be defined to handle the situation where stale cached data is being used.

3.2.2 Escalation

3.2.2.1 Issue

Sometimes the system cannot automatically recover from an error.

3.2.2.2 Recommendation

Systems should be designed to escalate the problem for handling by another process if the system cannot resolve the problem. The escalation process would eventually result in a person handling the problem if it cannot be resolved by the system.

3.2.2.3 Tradeoffs

Manual processes are costly to perform. It is preferable to automatically handle the error, but there will always be situations where a manual process cannot be avoided.

3.3 Auditing and tracking

3.3.1 Tracking

3.3.1.1 Issue

The operation of a system may need to be diagnosed when a problem occurs.

3.3.1.2 Recommendation

Messages that are sent or received should be tracked. Minimally, this will be a log entry that records a timestamp for the message, the message ID, and the source or destination. Additional information, such as the message itself, can also be logged.

It is the responsibility of each system to maintain its own tracking logs.

3.3.1.3 Tradeoffs

Logs incur an overhead to create and to store. These overheads increase as more information is stored in a log. Logs may be stored offline or purged to save space.

3.3.2 Auditing

3.3.2.1 Issue

Historical information may need to be kept for auditing purposes.

In this section, a distinction is made between tracking and auditing. Tracking is an optional feature used for technical diagnosis. Auditing is a mandatory feature used to support a business requirement (usually a legal requirement). It is possible to permit an operation to occur if its tracking requirements could not be met, but not if its auditing requirements could not be met.

3.3.2.2 Recommendation

Auditing requirements should be defined for a service. It can either be defined by the service interface specification, by an implementation profile of that specification, or by a mutual service level agreement.

The auditing requirements should always be based upon a real business requirement for it.

Currently, it is the responsibility of the individual systems to maintain their own audit logs. If there is sufficient requirements and demand, services to support auditing may be defined in the future.

3.3.2.3 Tradeoffs

Audit logs incur an overhead to create and to store. These overheads increase as more information is stored in an audit log. Audit logs may be store offline, but cannot be purged unless permitted by the business rules.

4 Security

4.1 Security protocol

4.1.1 Transport Layer Security (TLS)

4.1.1.1 Issue

WS-Security cannot be used in some situations. For example, when the target implementation toolkit and deployment environment do not adequately support WS-Security or there needs to be legacy support for TLS.

4.1.1.2 Recommendation

Service specifications should use Transport Layer Security (TLS) as an alternative mechanism for securing Web services [RFC2246].

TLS must be used with mutual authentication.

These TLS based services must be advertised in the SIL as a different interaction from those using WS-Security.

4.1.1.3 Tradeoffs

These services cannot be integrated with services that are secured with WS-Security.

4.2 Service invokers

4.2.1 Firewalls

4.2.1.1 Issue

Programs acting as Web services invokers should be operating from behind local firewalls. These firewalls may prevent outgoing connections, which will prevent connectivity from happening.

4.2.1.2 Recommendation

The firewall needs to be configured to allow the Web services software to initiate connections to the Internet.

4.3 Service providers

4.3.1 Firewalls

4.3.1.1 Issue

Programs acting as Web services providers should be operating from behind local firewalls. These firewalls may prevent incoming connections, which will prevent connectivity from happening.

4.3.1.2 Recommendation

The firewall needs to be configured to allow the Web services software to receive connections from the Internet.

4.3.2 DMZ

4.3.2.1 Issue

Many organisations deploy servers behind a Demilitarized Zone (DMZ). Confidential data may be compromised if it is left exposed in the DMZ.

4.3.2.2 Recommendation

Confidential data should not be stored in the clear in the DMZ. This also applies to decryption keys which can be used to decrypt confidential data.

A DMZ is created by having two firewalls. The outer firewall is placed between the Internet and the DMZ. The inner firewall is placed between the DMZ and the internal network. If computers in the DMZ are compromised the data in the DMZ could be at risk, but data in the internal network is still protected.

It is recommended that the decryption keys for confidential data be stored inside the inner firewall, and not in the DMZ. The decryption of confidential data will take place behind the inner firewall, and not in the DMZ. Therefore, no confidential data is ever stored in the clear in the DMZ.

This level of protection can be achieved using one of the following techniques:

- If the service uses an XML secured payload that is encrypted, the payload can be stored in the DMZ. Computers from the inner network can initiate a connection to the DMZ to retrieve the secured payload from the DMZ.
- The WS-Security secured SOAP message can be stored in the DMZ. Computers from the inner network can initiate a connection to the DMZ to retrieve it and provide any response data. To support request-response SOAP message exchange patterns, the retrieval of SOAP messages must occur very frequently.

5 Deployment

5.1 Service Instance Locator (SIL)

5.1.1 Automatic SIL updating

5.1.1.1 Issue

The Service Instance Locator must contain correct entries for deployed service instances.

It is necessary to ensure that the SIL is updated whenever service instances are deployed, modified, or decommissioned.

5.1.1.2 Recommendation

Service implementations should be designed to automatically update the entry in the SIL.

The SIL defines a standard service interface for maintaining SIL entries. A service instance can automatically invoke that interface whenever it is deployed. It can also check if its SIL entry needs updating when the service instance is started.

5.1.1.3 Tradeoffs

The service instance might require additional configurations to perform automatic updating of SIL entries.

5.2 Integration

5.2.1 Integration with proprietary systems

5.2.1.1 Issue

Not all systems will be compliant with the standards. For example, there are existing systems that do not support Web services. Over time, these may be modified to support the standards, but in some situations this may not be practical.

5.2.1.2 Recommendation

Use a gateway that translates between the standard interface and proprietary (non-compliant) system. The gateway connects to external organisations in the national e-health environment using the standard protocols. The gateway connects to the non-compliant system using some proprietary protocol or mechanism.

Using this approach, a non-compliant system can communicate with a compliant system. Two non-compliant systems can communicate with each other through their own gateways. This approach is illustrated in Figure 2.

The fact that a system is using a gateway should be invisible to an external organisation that communicates with it.

The gateway needs to perform the necessary transformations and mapping between the two protocols.

The gateway may be implemented by the owner of the non-complaint system, its vendor, or by a third party.

One possible design for a gateway component is the design described in the *Connectivity Coordinator: High Level Software Design* [CCHLSD2008].

5.2.1.3 Tradeoffs

The costs of implementing a gateway compared to modifying the non-compliant system needs to be considered.

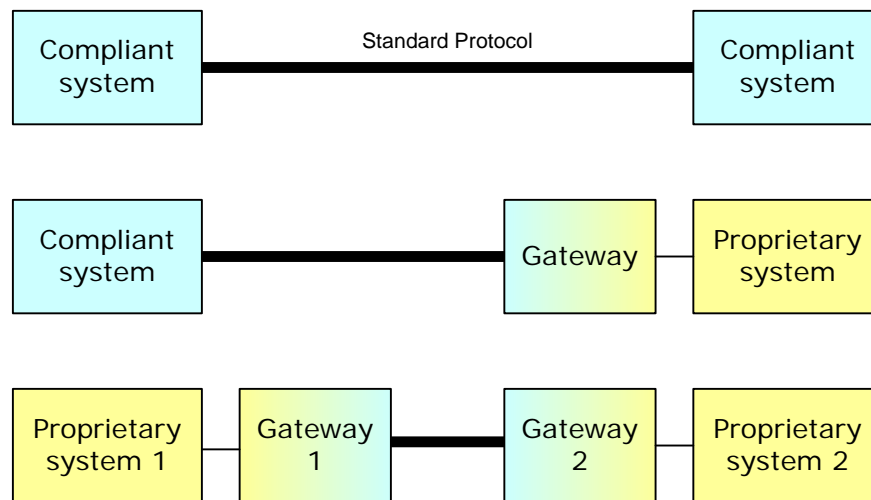


Figure 2: Integration with proprietary systems

Appendix A: Informative references

- [CCHLSD2008] NEHTA, *Connectivity Coordinator: High Level Software Design*, 1 December 2008.
- [EIIWSJAXWS] NEHTA, *Example Implementation of Interoperable Web Services: JAX-WS*, 1 December 2008.
- [EIIWSWCF] NEHTA, *Example Implementation of Interoperable Web Services: WCF*, 1 December 2008.
- [EIIWSWSE] NEHTA, *Example Implementation of Interoperable Web Services: WSE*, 1 December 2008.
- [RFC2246] IETF, *RFC 2246: The TLS Protocol Version 1.0*, T. Dierks and C. Allen, January 1999.
<http://ietf.org/rfc/rfc2246>