



---

# **Technical Architecture for Implementing Services**

## **Concepts and Patterns**

Version 1.0 – 21 December 2006

For Comment

---

**National E-Health Transition Authority Ltd**

Level 25  
56 Pitt Street  
Sydney, NSW, 2000  
Australia.  
[www.nehta.gov.au](http://www.nehta.gov.au)

**Disclaimer**

NEHTA makes the information and other material ("Information") in this document available in good faith. 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 © 2006, 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

<b>1</b>	<b>Introduction .....</b>	<b>1</b>
1.1	Purpose .....	1
1.2	Scope.....	1
1.3	Definitions, acronyms, and abbreviations .....	2
1.4	Conventions used in this document .....	3
1.5	Overview .....	3
1.6	Feedback.....	3
<b>2</b>	<b>Concepts.....</b>	<b>4</b>
2.1	Technical service definitions .....	4
2.1.1	Technical service.....	4
2.1.2	Service interface.....	5
2.1.3	Service implementation .....	6
2.1.4	Service instance.....	6
2.2	Types of services .....	7
2.2.1	Infrastructure services.....	7
2.2.2	Enabling services .....	7
2.2.3	E-health services .....	8
2.2.4	Other services (out of scope).....	8
<b>3</b>	<b>Components.....</b>	<b>9</b>
3.1	Infrastructure services.....	9
3.1.1	Service instance directory .....	9
3.1.2	Certificate service .....	9
3.2	Enabling services .....	10
3.2.1	IHI services.....	10
3.2.2	HPI services .....	10
3.3	Example .....	10
<b>4</b>	<b>Technical patterns .....</b>	<b>12</b>
4.1	Behavioural technical patterns.....	12
4.1.1	Request/response .....	12
4.1.2	One-way messaging .....	13
4.2	Informational technical patterns .....	15
4.2.1	XML and XML Schema.....	15
4.3	Non-functional technical patterns.....	15
4.3.1	Signing and encryption .....	15
<b>5</b>	<b>Performance, scalability and availability .....</b>	<b>18</b>
5.1	Federation .....	18
5.2	Caching .....	18
5.3	Subscription.....	19
	<b>Appendix A: Architecture Principles.....</b>	<b>20</b>
A.1	Background.....	20
A.2	Principles .....	21
A.2.1	Semantically rich .....	21
A.2.2	Consistent.....	21
A.2.3	Connected.....	21
A.2.4	Adaptable .....	22
A.2.5	Scalable.....	22
A.2.6	Open and vendor neutral .....	22
A.2.7	Cost effective .....	22
	<b>Appendix B: References.....</b>	<b>23</b>

# Change history

Version	Date	Comments
1.0	21-12-2006	Release for comment

# 1 Introduction

## 1.1 Purpose

This document describes the technical concepts and patterns for service implementation within the national E-Health Infrastructure. These technical concepts and patterns form the basis for a Technical Architecture of the national E-Health Infrastructure.

The national E-Health Infrastructure is the infrastructure that is being developed by the National E-Health Transition Authority (NEHTA) to support electronic health in Australia.

An architecture is a plan that determines the overall organisation and structure of a system. This Technical Architecture describes the high level technical design for how electronic health interactions will be supported and conducted in the national E-Health Infrastructure. It does so by describing the concepts and patterns that make up the architecture.

The Technical Architecture is intended to provide guidance for both the technical development of the individual systems which make up the national E-Health Infrastructure as well as for the systems that utilise it. It helps analysts and architects of individual systems know how to interact with the national e-health community.

NEHTA will also publish a guidelines document that describes how to implement these technical patterns using Web services. This document provides the high level design, for which the guidelines document will describe the implementation details. These guidelines will be released in first quarter of 2007.

The idea of concepts and patterns come from the NEHTA Interoperability Framework [NIF2006], which also defines the three perspectives of e-health, being the organisational, informational, and technical perspectives. This document is primarily concerned with technical concepts and patterns. This document does not address organisational and informational perspectives, except when they have an impact on the technical aspects of the national E-Health Infrastructure.

## 1.2 Scope

The national E-Health Infrastructure provides the infrastructure to support clinical information management between organisations within the Australian healthcare sector.

This document only describes the technical aspects of the national E-Health Infrastructure. The following issues are out of scope for this document:

- Governance of the e-health infrastructure and services.
- Standards development and governance.
- Financial models for funding and charging.
- Information ownership.

The Architecture is based upon a set of principles, which are described in Appendix A. These should be used as principles when following or applying the Architecture.

This document is a technical framework for developing e-health solutions. It is useful to different people in different ways. For example:

- Business analysts need to identify business requirements and describe them in a manner suitable for the solution architects to use. The analyst needs to be aware of the environment that the solution will work in; this

document identifies some of the common infrastructure and enabling services. The environment described in this document is designed to fit into a Services-Oriented Architecture.

- Enterprise Architects need to develop an enterprise architecture, policies and methodologies for their organisation. That enterprise architecture needs to take into account the organisation’s interactions with external parties. How those external interactions work is described in this document. The organisation’s policies also need to be shaped by those external interactions, especially when it comes to what the organisation makes available as externally available services. The organisation’s methodology will determine how services are structured. This document guides what is needed for structuring externally available services. Also, an organisation’s enterprise architecture needs to allow the internal systems to work with the world outside of the organisation. This document informs the Enterprise Architect how to interoperate with the outside world.
- Solution Architects develop technical designs and specifications for particular systems. They need to know what approaches they should use, the external influences or constraints, issues they need to be concerned about, and how they should structure their designs. This document provides information on each of these areas.

### 1.3 Definitions, acronyms, and abbreviations

Enabling services	Services that support the implementation of a national e-health infrastructure, and are visible to consumers and healthcare professionals.
E-health services	Services which are used directly by healthcare professionals or organisations in the provision of healthcare.
Infrastructure services	Services that support the implementation of a national e-health infrastructure, but are not visible to consumers and healthcare professionals.
Invoke	The action of a service requestor using a service which is being offered by a service provider.
Service implementation	A product (i.e. software) that conforms to a service interface.
Service instance	A specific deployment of a service implementation.
Service interface	The definition of the functionality of a service and its interface.
Service provider	The party operating a technical service.
Service requestor	The party using or consuming a technical service.
CA	Certificate Authority
HPI	Healthcare Provider Identifier
IHI	Individual Healthcare Identifier
NEHTA	National E-Health Transition Authority
PKI	Public Key Infrastructure
SOA	Service-Oriented Architecture
XML	Extensible Markup Language

## 1.4 Conventions used in this document

In this document, the national E-Health Infrastructure Technical Architecture will be referred to as “the Architecture”.

## 1.5 Overview

Chapter 2 “Concepts” identifies services as a central concept. It decomposes the specification of services into three attributes (behavioural, informational, and non-functional), which will be important for specifying services. It also classifies services into three categories: infrastructure, enabling and e-health services.

Chapter 3 “Components” describes the services which have been identified for the Technical Architecture.

Chapter 4 “Technical patterns” describe the common technical patterns.

Chapter 5 “Performance, scalability and availability” discusses performance, scalability and availability issues.

Appendix A lists the principles for the technical Architecture.

Appendix B lists the references used in this document.

## 1.6 Feedback

This document is currently a draft. It will be expanded and refined during the development of the national E-Health Infrastructure. In particular, it is expected that additional components and technical patterns will be added over time.

Feedback and contributions to this document is being sought, particularly from health and messaging software vendors, public and private health institutions and government health jurisdictions.

The deadline for feedback is 31 March 2007.

All comments should be directed to either:

Email: [securemessaging@nehta.gov.au](mailto:securemessaging@nehta.gov.au)

Post: Secure Messaging Initiative  
NEHTA  
Level 5  
300 Adelaide Street  
Brisbane, Qld 4000  
Australia.

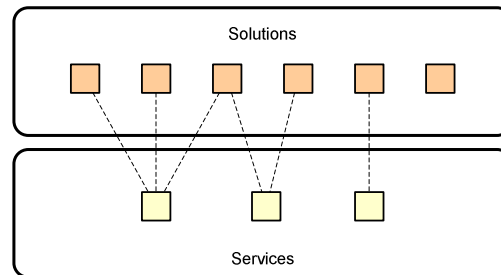
## 2 Concepts

This chapter describes the concepts associated with the Technical Architecture.

This chapter expands on the concepts defined in the NEHTA Interoperability Framework [NIF2006]. It defines a terminology for services and a classification of services.

### 2.1 Technical service definitions

The National E-Health Infrastructure follows a Service-Oriented Architecture (SOA) approach. The main impact of SOA is at the organisational level, where the requirements are analysed in terms of services to support the processes of providing healthcare. However, the decision to follow an SOA approach influences how solutions are implemented at the technical layer—resulting in solutions that are implemented using technical services, as illustrated in Figure 1.



**Figure 1: Solutions built using a set of services**

#### 2.1.1 Technical service

Services are well defined and self-contained units of functionality, which can be invoked by other software components.

The term “service” has many different definitions in both technical and non-technical contexts. Sometimes the term is used to refer to healthcare delivery services that a healthcare organisation provides to patients. However, this document is concerned only with technical services, as defined by the NEHTA Interoperability Framework.

Technical services are loosely coupled, so services can individually be easily substituted or upgraded. This is possible because a service is defined by its service interface, which is separate from how it is implemented—the interface can remain the same, while the implementation behind it can be changed. They can be developed and deployed by different parties, and they can be deployed in different locations.

Some technical services are standalone, while some services depend upon other services to perform their tasks. The implementation of a service can invoke other services to carry out its functionality, but it does not have to.

Technical services are not accessed directly by people, rather they are invoked (i.e. programmatically used) by a software component or another technical service. One possible method to allow users to use a technical service is via a “portal” or Web application. However, in that situation it is the portal that programmatically uses the technical service; the user only interacts with the user interface provided by the portal.

The technical services in the e-health environment need to have these properties:

- Have semantics which are defined through standards;
- Be connected via a common network; and
- Be defined using open standards which are available to all vendors and implementers.

In this document, the term “service provider” refers to the party operating the service. The term “service requestor” refers to the party using the service. The service requestor “invokes” the service to use it.

An important feature of good services is the documentation and agreement on the service interface. The “service interface” describes how the service is to be used and what it does.

Services must be well specified and documented, so that others can use them.

The terms “service interface,” “service implementation” and “service instance” will be defined by the NEHTA Interoperability Framework [NIF2006]. The definitions for these terms are repeated below.

### 2.1.2 Service interface

The “service interface” defines the external view of a technical service: how external parties interact with the service, and what the service contractually promises to provide. However, it does not specify how the service itself should be constructed. The service itself can be constructed using any computer platform or technology, as long as it provides an external interface that conforms to the service interface.

The service interface is a specification. It is a document or set of documents that describe the external view of a service. The documents can be written in natural language for people to use, or in a formal language for computer programs to use, or a combination of both.

This Architecture has chosen to divide a service interface into three separate sets of attributes:

- **Behavioural** attributes, which define the tasks or functions provided by the service;
- **Informational** attributes, which define the data that is used and produced by the service; and
- **Non-functional** attributes, which defines constraints that affect the qualities of the service, but not the functionality of the service. For example: security, reliability, and performance.

It is recommended that services be specified by separating their interfaces into behavioural, informational, and non-functional attributes.

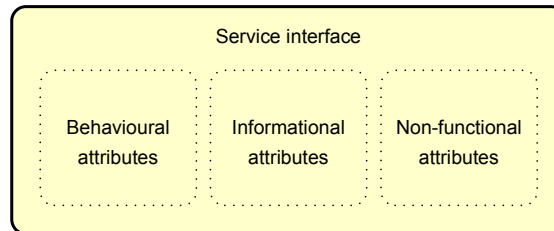
It is important that these different attributes are kept distinct as this makes the service interfaces more adaptable, and allows the e-health environment to be more consistent, scalable, and cost effective.

The separation of these three attributes allows the different parts of a service interface to be reused for other service interfaces. For example, “store and retrieve” type of behaviour could be used with x-ray images, to define a radiology service. A referral service could be defined that handles referral documents instead of x-ray images—they both share the same behavioural, but use different informational, attributes. Another example is when there are two referral services: one referral service could specify a high level of security and reliability as its non-functional attributes, and a different referral service could specify a lower level of security and reliability—they both share the

same behavioural and informational attributes, but have different non-functional attributes.

A technical service needs all three attributes to be specified as a part of its service interface. In the past, e-health standards have focussed more on the information (i.e. the content of the messages) than on how that information is supposed to be used. It is important to recognise that all three attributes are required for technical services to be interoperable.

It should be noted that this Architecture is at the technical level, so it only addresses the technical aspect of these attributes. For example, the behaviour is only about the operational service behaviour. It is not the complete behaviour of the service, which would need to include other behavioural aspects of the service (e.g. service policies and process behaviours).



**Figure 2** Attributes making up a service interface

### 2.1.3 Service implementation

The term "service implementation" refers to a software component that conforms to a service interface.

There can be zero or more service implementations for a particular service interface. That is, different products can be created to implement the same service interface.

In this document, the term "service implementation" refers to an artefact, such as a computer program. The term "implementation" is not used here to refer to the development, training, roll out or deployment process. However, the term "implement" is used in this document to refer to the development or creation of the service implementation artefact.

### 2.1.4 Service instance

The term "service instance" refers to a particular deployment of a service implementation.

These deployments can be on one computer, or on several different computers. There can be multiple service instances running at the same time. Since a service interface can have multiple service implementations, a single service interface can also have multiple service instances (and those service instances could consist of the same service implementation or different service implementations.)

To bring these three concepts together, consider an example of an automatic teller machine (ATM) used by banks. There is a specification that describes how an ATM communicates with a banking network that is used by many different banks—that is the service specification. It describes how other systems expect to communicate with an ATM, but not how it should be constructed. A particular manufacturer creates a model of an ATM—that is a service implementation. A bank buys several ATMs from that manufacturer and locates them at its branches—each one of those models is service instance. Other manufacturers can also build ATMs which conform to that specification. That bank can choose to buy ATMs from a number of different manufacturers if it wanted to. As long as the service specification is followed, those different service instances can work together.

## 2.2 Types of services

Three broad categories of technical services have been identified:

- Infrastructure services;
- Enabling services; and
- E-health services.

The purpose of this classification is to distinguish between the different roles the services play in the national E-Health Infrastructure. The classification also provides guidance as to which services are owned and managed by common infrastructure organisations versus those owned by specific solutions.

### 2.2.1 Infrastructure services

Infrastructure services are defined as those that support the implementation of a national E-Health Infrastructure, but are not visible to consumers and healthcare professionals. For example, a policy management service or activity monitoring service is an infrastructure service.

A service in this category would normally pass all of the following tests:

- Is the service hidden from the view of consumers and healthcare professionals?
- Does the service provide or manage information used by multiple healthcare or enabling services?

This Architecture identifies a set of infrastructure services for the e-health environment. NEHTA shall define and standardise this initial set of infrastructure services.

As the e-health environment evolves to support more sophisticated solutions, additional infrastructure services could be required. NEHTA, with input from other organisations, would define and standardise these additional infrastructure services.

### 2.2.2 Enabling services

Enabling services are defined as those that support the implementation of a national e-health infrastructure and are visible to consumers and healthcare professionals. They are typically visible in the processes that relate to the operation of the infrastructure.

Enabling services are different from e-health services because the benefit to healthcare is a secondary rather than direct benefit.

A service in this category would normally pass all of the following tests:

- Is the service visible to healthcare professionals and consumers?
- Does the service provide or manage information used to enable multiple e-health services?

This Architecture identifies a set of enabling services for the e-health environment. NEHTA shall define and standardise this initial set of enabling services.

As the e-health environment evolves to support more sophisticated solutions, additional enabling services could be required. NEHTA, with input from other organisations, would define and standardise these additional enabling services.

### 2.2.3 E-health services

E-health services are defined as services which are used directly by healthcare professionals or organisations in the provision of healthcare.

A service in this category would normally pass all of the following tests:

- Is the service directly visible to the healthcare professional or consumer in healthcare scenarios?
- Does the service provide or manage information used directly for the care of consumers?

This Architecture does not define any e-health services. The e-health services should be developed by healthcare experts for their field of expertise.

### 2.2.4 Other services (out of scope)

There are other types of services, but these will be considered to be outside the scope of the national E-Health Infrastructure. These services will not be considered as part of the Architecture.

Examples of services that are not in the Architecture are:

- Proprietary or vendor specific services;
- Services which are only used internally within organisations and jurisdictions;
- Services which are applicable to only to a small subset of the e-health environment (e.g. specific to a particular jurisdiction); or
- Services which are not related to clinical healthcare (e.g. payroll).

These types of services should be influenced by the Architecture. However, it is outside the scope of this document to describe how these other services are to operate.

# 3 Components

This chapter describes the components of the Architecture.

The scope of national E-Health Infrastructure does not include the computer network that connects the service requestors to the service providers. The national E-Health Infrastructure will make use of existing computer networks that will be provided by external parties.

These services identify separate logical functions. They do not specify how the services may be implemented, physically deployed, or who is responsible for them. It is possible for more than one service to be implemented by one party. However, logically they are separate services because they provide different functions.

It is important to remember that the national E-Health Infrastructure is concerned with inter-organisation communications. Therefore, the components described here are used across organisations. They are not designed for internal intra-organisation use.

## 3.1 Infrastructure services

The following infrastructure services have been identified by NEHTA:

- Service instance directory
- Certificate service

### 3.1.1 Service instance directory

A “service instance directory” is used to determine which service instance to use, where that service is located, and how to interact with it.

The service instance directory is an important part of the infrastructure. In a national environment, service requestors might not know which service instance is required beforehand (e.g. it needs to contact a provider that it has not dealt with before). Also, because the service instances and where they are located might change over time.

The service instance directory allows lookup of services based on:

- Type of service (e.g. pathology report receiving service); and
- Provider being contacted.

### 3.1.2 Certificate service

The certificate service supports the use of Public Key Infrastructure (PKI) as a security mechanism.

PKI is the first of several possible security mechanisms that the Architecture will support. In the future, additional security related services could be introduced.

The certificate service supports the use of X.509 certificates for encryption and digital signatures. It will provide the following operations:

- Retrieve the public certificate for a particular provider (based on their HPI number);
- Check if a certificate is valid (i.e. it has not been revoked).

This service could be provided by a “Certificate Authority”, an organisation that issues certificates. However, it can also be provided by other parties.

## 3.2 Enabling services

### 3.2.1 IHI services

IHI services will support the use of IHIs in the National E-Health Infrastructure. These IHIs will be used to identify people who are consumers of e-health services.

There are a number of services being developed to support IHIs. Those that will be used in general healthcare communications are:

- Identification and matching service, to relate an individual to their IHI number.
- Directory service, to obtain published data associated with a particular IHI number.

The IHI service is being developed by NEHTA.

### 3.2.2 HPI services

The HPI service will support the use of HPIs in the national E-Health Infrastructure. These HPIs will be used to identify people and organisations who are providing e-health services.

There are a number of services being developed to support HPIs. Those that will be used in general healthcare communications are:

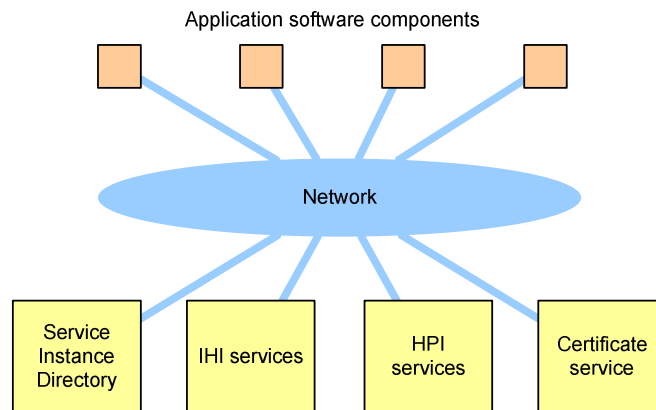
- Identification and matching service, to relate a provider to their HPI number.
- Directory service, to obtain published data associated with a particular HPI number.

The HPI service is being developed by NEHTA.

## 3.3 Example

In general, communications between software components will make use of information from the infrastructure services and enabling services.

This section describes an example which shows how these different services could be used.



**Figure 3 Infrastructure services**

Consider a situation where a General Practitioner wishes to send a referral to a specialist. The General Practitioner will be using a practice management software program (the “GP program”). The specialist uses a program (the “specialist’s program”) which implements the referral receiving service interface.

The following steps occur:

1. The GP program uses the IHI service to look up the IHI number for the patient. For example, this lookup might use the patient's name and date of birth to find the IHI number.
2. The GP program uses the HPI service to look up the HPI number for the specialist. For example, this lookup might use the name of the specialist.
3. The GP program uses the Service Instance Directory to find the service instance to send the referral to. This is a query asking for the "referral receiving service" for the specialist's HPI number. The result will be a service endpoint: an address of the service instance that the GP program must use to send a referral to that particular specialist.
4. The GP program uses the certificate service to obtain the public PKI certificate for the specialist. This is a query based on the specialist's HPI number.
5. The GP program creates a referral message that contains the patient's IHI number, digitally signs it with the GP's private PKI key, and encrypts it using information in the public PKI certificate of the specialist.
6. The GP program contacts the "referral receiving service" for the specialist, and sends it the message.
7. The specialist's program uses the certificate service to check the status of the GP's public PKI certificate.
8. The specialist's program decrypts the message using the specialist's private PKI key.
9. The specialist's program authenticates the sender by checking its digital signature. In this step, it also verifies the integrity of the message.

This process can vary, depending on how much information can be obtained from other sources (e.g. the patient has supplied their IHI number, or it has been cached from a previous lookup).

With appropriate caching, the above steps could simplify down to:

1. The GP program creates a referral message that contains the patient's IHI number, digitally signs it with the GP's private PKI key, and encrypts it using information in the specialist's public PKI certificate.
2. The GP program contacts the "referral receiving service" for the specialist, and sends it the message.
3. The specialist's program decrypts the message using the specialist's private PKI key.
4. The specialist's program authenticates the sender by checking its digital signature. In this step, it also verifies the integrity of the message.

# 4 Technical patterns

The concept of a “technical pattern” was defined in the NEHTA Interoperability Framework [NIF2006]. These technical patterns “capture some commonly occurring, existing or emerging, structures, approaches and technical characteristics.”

This section describes the technical patterns which have been identified for the Architecture. These technical patterns have been grouped according to whether they deal with behavioural, informational, or non-functional attributes of a service interface.

## 4.1 Behavioural technical patterns

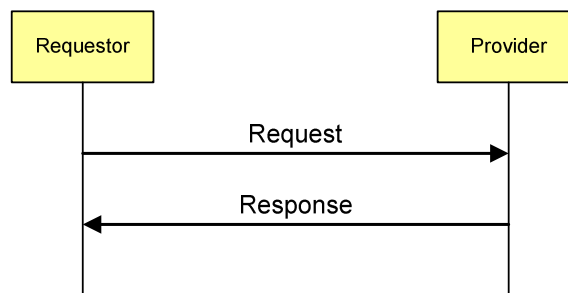
### 4.1.1 Request/response

The request/response technical pattern is an interaction where there is a single request and a single response that depends on the request.

There are two parties involved: the requestor and the provider. The requestor sends the request to the provider, and the provider then sends a response to the requestor.

In terms of SOA, the provider is offering a service to the requestor. The requestor is the service requestor, and the provider is the service provider.

The request/response technical pattern is useful for operations that need to return a result to the requestor. For example, querying a service for some information.



**Figure 4 Request/response technical pattern**

#### 4.1.1.1 Coupled interaction

In this technical pattern, the response is directly coupled to the request. When the request arrives at the provider, the response is immediately generated and sent back to the requestor.

The request/response technical pattern should not be used if there is no response or if the response is generated some time after the request was received. There are two possible options for handling decoupled responses:

- Define a separate request/response interaction to deliver the result. That is, a request/response pair is used to send the request, and another request/response pair is used to send the response.
- Use the one-way messaging technical pattern (described below). Sending a one-way message for the request, and another one-way message for the response.

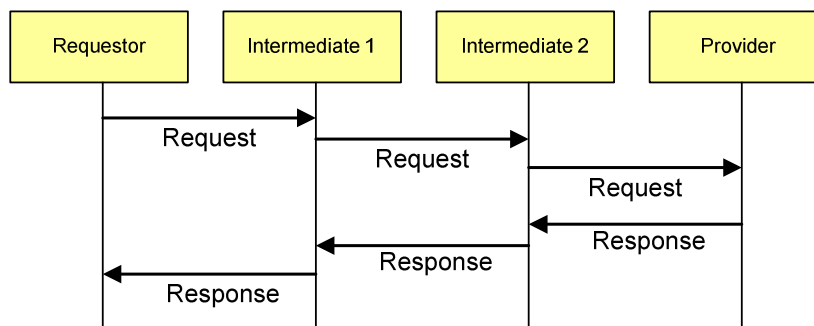
#### 4.1.1.2 Chained services

The request/response technical pattern can be used when there are multiple intermediate services between the requestor and the provider. The

intermediates forward the request on towards the provider, which generates the response, and then the response is forwarded back towards the requestor. The intermediates can provide additional value, by processing the request or response, before forwarding them on. This is illustrated in Figure 5.

The service interface offered by the intermediates is exactly the same as that offered by the provider. There are no new protocols involved. However, the implementation of the intermediate services is different, because they need to forward the requests and responses instead of just handling them.

It is expected that the major use of chained services would occur within organisations. Intermediate services would process requests before they leave the organisation, or process responses that arrive at the organisation. Therefore the exact operation of the intermediates (such as how the routing is performed) is outside the scope of the Architecture.



**Figure 5 Chained request/response technical pattern**

#### 4.1.2 One-way messaging

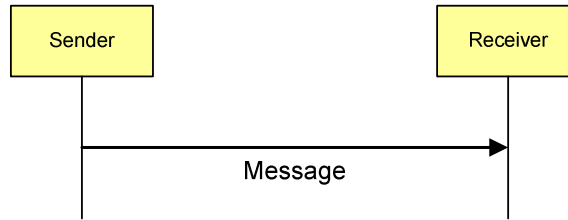
The one-way messaging technical pattern is an interaction where there is a single message sent from one point to another.

There are two parties involved: the sender and the receiver. The sender initially has the message to send, and the receiver ends up having the message.

In terms of SOA, the receiver is offering a service that is used to receive the message. The sender is the service requestor, and the receiver is the service provider.

The one-way messaging technical pattern is useful for message oriented processes. The processes of healthcare have commonly been message oriented, where documents are sent between providers (e.g. referrals, prescriptions, discharge summaries, and reports). Therefore, this technical pattern has been included to support these types of operations. However, as the e-health environment becomes more connected and the demand for interactive functions increase, the request/response technical pattern could be preferred over one-way messaging.

There is no explicit acknowledgement of the message from the receiver. However, this can be added through higher order processes. For example, an acknowledgement message could be sent in a separate interaction. If the acknowledgement message has not been received after a certain time, the original message could be resent or an alternative action can be taken.



**Figure 6 One-way messaging technical pattern**

4.1.2.1 Decoupled interactions

The one-way messaging technical pattern can be used for decoupled interactions, where the request and response are treated as two separate one-way messages.

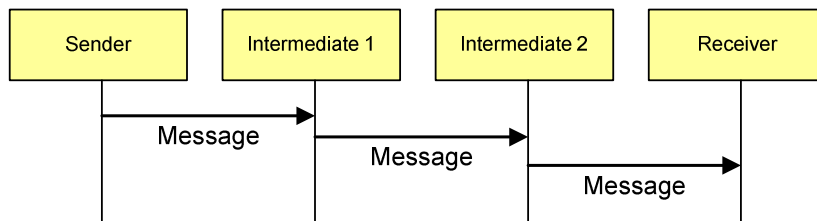
This technical pattern can also be used for notifications. However, the request/response technical pattern should also be considered as a possible option for delivering notifications. The technical pattern that is used will depend on the particular solution’s requirements.

4.1.2.2 Chained services

The one-way messaging technical pattern can be used when there are multiple immediate services between the sender and receiver. The intermediates forward the message to the receiver. This is illustrated in Figure 7.

The service interface provided by the intermediates is exactly the same as that offered by the receiver. There are no new protocols involved. However, the implementation of the intermediate services is different, because they need to forward the messages instead of just receiving them.

It is expected that the major use of chained services would occur within organisations. Intermediate services would process the messages before they leave the organisation, or process messages that arrive at the organisation. Therefore the exact operation of the intermediates (such as how the routing is performed) is outside the scope of the Architecture.



**Figure 7 Chained one-way messaging technical pattern**

4.1.2.3 Store and forward

A particular type of intermediate service is one that is used to store and forward the message.

This type of service will be called a “proxy receiver.” The sender sends the message to the proxy receiver, instead of directly to the receiver. When the receiver becomes available, it notifies the proxy that it is able to receive messages, and the proxy then sends the messages that it has to the receiver.

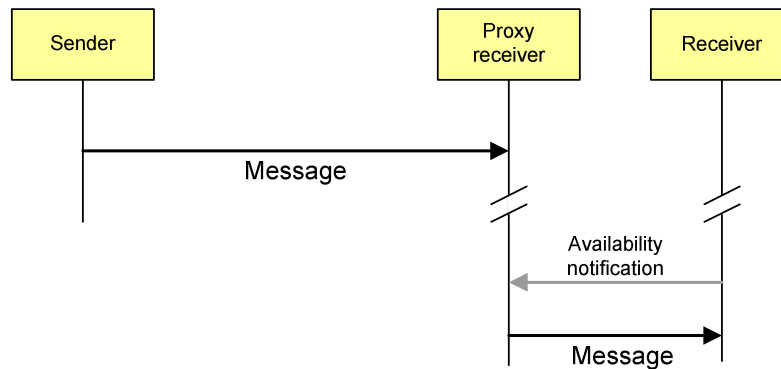
This allows the one-way messaging technical pattern to be used for disconnected receivers. These receivers are unavailable when the sender sends the message. For example, they might be intermittently connected the network (e.g. dialup access) or do not have a fixed location where the sender can always find them.

Store and forwarding can only be used if the message does not have to immediately arrive at the receiver. The sender has no guarantee as to when

the message will reach the receiver, because it is not known when the receiver will become available.

The same receiver service interface is implemented on the proxy. This allows the operation of the sender to be the same, irrespective of whether it is communicating with the receiver or a proxy. It also allows the receiver to operate the same, irrespective of whether the sender or the proxy is communicating with it. Thus, the proxy can be introduced or removed with minimal impact on the services involved.

Store and forwarding solves the problem of a not always connected environment. Although always-on broadband connections are becoming more widely available, there are still many users on dialup access. And there will always be a need to support rural and mobile providers. It also allows organisations to delegate the responsibility of receiving their messages to a third party.



**Figure 8** Proxy messaging technical pattern

## 4.2 Informational technical patterns

### 4.2.1 XML and XML Schema

Information can be modelled and represented in many different ways.

This technical pattern recommends that information can be represented using the XML infoset and encoded using the syntax of XML. It also recommends the use of XML Schema to describe those XML documents.

This technical pattern ensures that the data works natively with Web services. Other forms of data can be used with Web services. However, they would have to be represented as binary data which is encoded inside an XML wrapper.

## 4.3 Non-functional technical patterns

### 4.3.1 Signing and encryption

The signing and encryption technical pattern defines how digital signatures and encryption can be used to provide security over messages. They are used to provide both authentication (verifying the identity of the sender and receiver) and confidentiality (allowing only the intended recipient to view the message).

This pattern assumes the use of asymmetric cryptographic algorithms, such as those supported by Public Key Infrastructure (PKI).

This pattern is described in three parts:

- The first part defines a common set of roles involved in the sending and receiving of messages.
- The second part describes how digital signing should be used.
- The third part describes how encryption should be used.

#### 4.3.1.1 Messaging roles

This technical pattern identifies four roles in communications:

- Origin, the party responsible for creating the content of the communications.
- Sender, the party that performs the sending of the communications.
- Receiver, the party that accepts the communications.
- Destination, the party that needs to act on the communications.

For example, consider the situation of a discharge summary that is sent from a hospital to a GP. The origin is the discharge nurse who creates the discharge summary. The sender is the hospital, or more precisely the hospital's discharge system. The receiver is the GP clinic's practice management system. The destination is the General Practitioner.

Sometimes the origin and sender are the same entity. Sometimes the receiver and the destination are the same entity. However, in both cases the parties still need to be identified.

#### 4.3.1.2 Digital signing

The possible options for the origin and sender are shown in the table below. Entries that say "signs" mean that the entity provides a digital signature on the message. Entries that say "identified" means the identity of the entity is indicated in the message, but there is no technical means of verifying that assertion in the message.

Origin	Sender	Description
Identified	Identified	Not recommended, because there is no mechanism for the receiver or destination to verify the source of the message.
Identified	Signs	Recommended. This is expected to be the most common case, where the sender system is responsible for authenticating their local users and only needs to identify them in the message. The sender is responsible for the message.
Signs	Identified	This is the situation when the provider uses their own personal credentials to sign the message, and the organisation/system they use to send the message is simply identified.
Signs	Signs	This is when both the origin and sender signs the message.

#### 4.3.1.3 Encryption

The possible options for the receiver and destination are shown in the table below. Entries that say "encrypted" means that the message is encrypted so they can decrypt it—and in the process authenticating them to view the

message. Entries that say “identified” means the identity of the entity is indicated in the message, but there is no technical means in the message of enforcing that they are the only ones allowed to view the message.

Receiver	Destination	Description
Identified	Identified	Not recommended, because there is no mechanism to enforce confidentiality.
Identified	Encrypted	This is used when the message needs to be kept confidential from the receiver. For example, a report that is intended for a specific GP; even though it needs to be received by the GP’s clinic, it ensures that no one else in the clinic can view the message.
Encrypted	Identified	Recommended. This is expected to be the most common case, where the receiver system is responsible for directing the message to the destination and for enforcing appropriate confidentiality once the message has entered the receiver. It allows the receiver (usually an organisation) to do something different with the message if it is required (e.g. the destination person no longer works at the organisation).
Encrypted	Encrypted	Not recommended, because it provides little value over the encrypted receiver and identified destination approach.

# 5 Performance, scalability and availability

Performance, scalability and availability are often raised as concerns when shared services are used. These concerns can be addressed by choosing strategies for how the services are implemented and deployed—addressing these concerns do not change the concepts or patterns that are defined in this document. However, to mitigate these concerns, this chapter will describe some of the strategies that could be used to deal with performance, scalability and availability.

Consideration must be given to ensure that the performance and scalability of services will meet the demands of a national e-health environment.

The services need to handle the volume of operations that the national e-health environment will generate. This volume is expected to increase over time, as the deployment and use of e-health solutions increase.

Three strategies are proposed to ensure that the infrastructure has the capacity to meet current and future needs:

- Federation
- Caching
- Subscription

Which approach is used depends on the needs of the individual service. This list is not comprehensive, and other approaches may also be used.

## 5.1 Federation

Federation is using multiple systems together in such a way that they appear to operate as a single system.

Through the use of federation, the loads on a single service can be distributed to multiple systems. The performance of each individual system increases because the load on it is reduced. The capacity of the entire system increases with the adding of individual systems. The availability of the entire system is improved because individual systems can fail and their functions can still be supplied by the remaining systems.

## 5.2 Caching

Caching relies on locally saved copies of information to be used, instead of requesting the information every time it is required. This can reduce the load on the information source, as well as reducing the time required to obtain the information.

The main risk is that the cached information is out of date. Thus, whenever caching is used, the following must be well defined:

- The duration that information can be cached for;
- The failure situations if out of date information is used;
- How the cached information can be invalidated by the information source; and
- How the user can bypass the cache.

## 5.3 Subscription

Subscriptions are when a software component registers its interest in certain information with a service. When that information becomes available, the component is notified of it. Only components which have registered interest in the information are notified.

# Appendix A: Architecture Principles

## A.1 Background

The important drivers for this Architecture are the requirements to provide:

- Access to information where and when it is required. This is to enable more informed clinical care and to reduce unnecessary duplication.
- The appropriate exchange of information across the different parts of the health system—especially between different organisations and health departments. This level of interoperability is to enable collaboration and sharing of data between providers. The information exchanged must be able to be correctly understood by the parties involved in the exchange. This is enabled by having structured data and vocabularies, which enable systems to make more use of automated processing and processes.
- Components that can be composed and reused. This is to reduce inefficiencies and inconsistencies, as well as enhancing the ability to respond to changing needs in healthcare.

The Architecture addresses issues of interoperability between organisations and between health departments. It focuses on long-term, nationwide interoperability—instead of directly addressing immediate local integration issues.

However, the Architecture should also be considered when developing local systems (systems which are used internally within organisations or for niche solutions). While individual organisations will remain free to use different approaches to developing local systems, they must be able to provide external interfaces that properly interface to the national infrastructure. The approaches used in this Architecture can be applied at the local level, and there are obvious benefits in aligning the internal approaches used within organisations to those used externally. This alignment is encouraged.

Equally, the Architecture does not aim to address every aspect of what is a very large and diverse healthcare industry, but rather it defines the “e-health environment” as being those systems:

- Used for information management,
- In the context of clinical care,
- Between organisations and jurisdictions.

*Information management* means it is focused on data that can be interpreted and processed by a computer. This usually means dealing with structured data, as opposed to unstructured data. For example, handling voice telephone calls would be out of scope, since the conversation is not intended to be interpreted by a computer (assuming no voice recognition system is being used).

*Context of clinical care* means clinical data or data that is related to processes for providing clinical care.

*Between organisations and jurisdictions* indicates that it is concerned with the external communications between organisations and jurisdictions.

This technical Architecture is concerned with information technology. However, it must be a part of a larger strategy for achieving clinical and operational healthcare benefits. That larger strategy would need to involve policy and workplace issues, which are outside the scope of this technical Architecture.

This Architecture has been developed with the following set of principles. These principles should be considered when developing new components for the Architecture, or when revising the Architecture.

## **A.2 Principles**

### **A.2.1 Semantically rich**

The Architecture must support a semantically rich e-health environment. The term “semantically rich” is used here to mean that the information that is communicated around the e-health environment can be interpreted and processed.

Semantic richness is implicitly a part of any task in the current local and non-electronic healthcare environment. However, these semantics must be made explicit so that they are available for processing by computers. Semantic richness is vital for interoperability in a nationwide e-health environment.

The information must be properly understood for the recipient to ensure that it can be checked, processed, and stored correctly. In addition to the information itself, the semantics of the behaviour and policies also need to be agreed upon.

### **A.2.2 Consistent**

The Architecture must promote a consistent e-health environment. Consistency is necessary for interoperability, as well as to facilitate better management and control of information.

One important aspect of consistency involves identifying authoritative sources of information, and ensuring that other systems which rely on that information are synchronised with the authoritative source. This applies to both information and policies.

Another aspect of consistency is to favour the adoption of existing standards over developing new ones that reproduce the functionality that already exists. The corollary of this is to develop standards which can be adopted by others.

### **A.2.3 Connected**

The Architecture must favour a connected e-health environment. This is to ensure that all participants are able to share and exchange information on an equal basis, and not to have multiple-tiers where some participants are disadvantaged over others.

The e-health environment needs to favour direct connections which are always available. Broadband connections are already available in many locations, their adoption is increasing, and they will be generally available in the future.

The Architecture should take advantage of this connectivity when it is available. Depending on the environment of a solution, this level of connectivity may not always be available, so the solution will need to handle this in an appropriate manner.

It is recognised that alternatives are also necessary to support disconnected modes of operation. Some participants will still be reliant on intermittent dial-up connections, and network failures can never be eliminated. However, the preferred mode of operation should be targeted at a fully connected environment.

One important aspect of being connected is the ability of any provider to be able to communicate with any other provider when they need to. The e-health environment cannot be effective if there are islands of participants who are disconnected from each other.

### **A.2.4 Adaptable**

The Architecture must be adaptable enough to support a diverse range of requirements as well as being able to evolve to meet new requirements that will arise in the future.

The e-health environment is currently very diverse. There are many different types of processes in healthcare, as well as many different types of tools and techniques being used.

The e-health environment is also constantly changing. These changes could come from within the healthcare (e.g. new medical processes or policy changes), or from outside of healthcare (e.g. changes in technology). The Architecture must be able to handle these changes. The systems built using the Architecture should allow for changes, so that they can remain useful in a changing environment.

The Architecture also serves to discourage short-sighted solutions from being adopted. Short-sighted solutions are those that meet immediate needs, but fail to meet future needs. This can lead to solutions which become obsolete quicker, and are difficult or impossible to integrate with.

### **A.2.5 Scalable**

The Architecture needs to facilitate scalable system design to meet the growing needs of the e-health environment. There will be ongoing growth in the range of systems and functions that it needs to support. There will also be continued growth in the volume of tasks being conducted in the e-health environment.

### **A.2.6 Open and vendor neutral**

The Architecture needs to be based on open standards which are vendor and product neutral.

The Architecture must support a heterogeneous environment consisting of multiple vendors and multiple platforms. It is important to ensure that vendor lock-in will not limit the growth of the e-health environment. It is also important to allow the systems to evolve to meet future needs and to take advantage of newer technologies when they become available.

### **A.2.7 Cost effective**

Finally, the Architecture needs to be practical and cost effective to implement. Aspects of this include favouring the reuse of components (instead of unnecessary duplication), and the use of industry standard technology (instead of custom solutions).

# Appendix B: References

[NIF2006] NEHTA, Interoperability Framework, 1.0, 01-04-2006.