INTERWEAVE
CONNECTING CARE

Cookbook for Regional Interoperability Detailed Design Paper #003

Conceptual Design for a FHIR Proxy Server

FOR GENERAL DISTRIBUTION

Version 2.0 – 20th April 2019

Abstract Interoperability Cookbook Anchor Points

Section	Title
3.2.1	FHIR Proxy Service
4.1	FHIR Service Point

Table of Contents

Version Control

Version	Release Date	Released By	Reason for Release
1.0	21/01/2019	R Hickingbotham	Preliminary working draft
1.1	9/02/2019	R Hickingbotham	Internal discussion draft
2.0	20/04/2019	R Hickingbotham	Revision following review

Reviewers

Initials	Name	Organisation
CS	Craig Staniforth	York Teaching Hospital
DM	Dave Maidment	Leeds City Council
GC	Gi Cheung	York Teaching Hospital
GH	George Hope	Leeds City Council
GS	Graham Smith	Leeds City Council
JB	Jonathon Booker	Wakefield CCG
JI	Jolyon Ingles	Harrogate & District
MH	Martin Harvey	Harrogate & District
MT	Martyn Tinsley	Leeds City Council
RC	Richard Cannon	Leeds Teaching Hospitals
RO	Rob Organ	Synanetics
SS	Stephen Stewart	Sheffield Teaching Hospitals
SSh	Steve Shaw	Synanetics
TP	Toby Page	Leeds City Council

Release Notes for this Version

1) Added description of RESTful asynchronous processing.
2) General update for review comments from v 1.1.

1 Introduction

1.1 Purpose of this Document

This document is one of a series of design papers which underpin the Abstract of a Cookbook for Regional Interoperability (the Abstract Cookbook). These papers, in their totality, describe the technical components and the standards which form the YHCR system of systems. They are intended as a basis for developing or procuring software and so are expressed at a level of precision which is intended to avoid ambiguity but with a consequence that they are focussed to technical readers.

Design papers are anchored to topics which are discussed in the Abstract Cookbook. They are elaborations of the concepts which were first introduced by the abstract and new content is further detail rather than variations of previously established core principles.

This document (design paper 003 - "Conceptual Design for a FHIR Proxy Server") describes at a logical level the functioning of interface technology which could be used to FHIR-enable one or more data sources within an organisation acting as a data provider to the YHCR.

1.2 What is a FHIR Proxy Server and Why is it Needed?

All data providers to the YHCR are expected to comply with a minimum technical capability and provide access (sometimes bi-directional) to a precisely defined set of FHIR resources. This capability is aligned with a maturity model which allows it to be built incrementally over time.

It is hoped that system vendors will ultimately take on responsibility for providing native FHIR interfaces which comply to this specification and if they do so, then the need for FHIR proxy technology will be obsoleted. However, our assessment of current vendors is that FHIR support is sporadic and basic at best. Until the situation improves, the onus will be placed on care providers to take responsibility for FHIR-enabling their own data sources. Often this will involve transforming data from existing data warehouses or reporting sources into FHIR resources and implementing an HTTPS REST service. A FHIR Proxy server provides the technical implementation of FHIR, focusing the effort of the care provider on mapping items from the data source onto FHIR resources.

1.3 A Conceptual Design

At its most basic level this document acts a requirements specification for the capability that is expected of the FHR interface: the API which is exposed by a data provider to the YHCR, how it should be secured, and other functionality expected of the interface such as auditing and application of consent rules. Functional specifications are based on the definition of maturity levels and it is intended that an implementor could develop the required functionality incrementally. A separate paper (design paper 023 – The YHCR Maturity Model) draws together maturity definitions from various design papers and aligns them the clinical use cases that they support.

The document goes beyond the specification of requirements and offers a conceptual design for how the functionality might be delivered. This approach is not intended to bind the hands of implementors in any way but, rather, is used as a device for illustrating the requirement in more detail. However, a FHIR Proxy model implementation will be offered by the YHCR for which the development has been driven closely by these designs and this document will act as a guide to users of this software. Model FHIR Proxy software will be available for the following platforms:

- InterSystems Ensemble/HealthShare
- QEWD (Node.JS)

1.4 Relationship of this Document with the HL7 FHIR Specification

This document is complementary to the FHIR specification. It defines the minimum level of support required by the YHCR for FHIR at an endpoint at various levels of maturity but all details of the FHIR implementation are drawn from the Standard for Trial Use 3 (STU3) of the HL7 specification which can be found at: hhtp://www.hl7.org/fhir/STU3.

The HL7 specification should be the primary point of reference for any implementor of the standards in this document.

1.5 Intended Users of the This Document

This paper is particularly relevant to:

- Data providers developing FHIR proxy software.
- Implementors of YHCR model FHIR Proxy servers
- Data consumers wishing to understand the API capability expected of data providers.

1.6 Maturity Levels Referenced in the Document

The Abstract Cookbook defines 6 maturity levels which are aligned with capabilities that support the achievement of programme milestones. This document associates detailed technical capabilities with each of these milestones and further enhances their definition so as to be relevant to the subject matter of this document: local support for the technical FHIR standard.

#	Abstract Cookbook Definition	Definition Applied to Subject
1	Read and search access broadly aligned with GP Connect specification	Direct retrieval of a set of resources representing basic facts about a person. Support for searches limited to a subset of possible search paths using basic syntax. XML and JSON. Resource versioning.
2	Read and search access to care plans, procedures undertaken, test results, diagnoses and other clinical data using local coding systems.	Expanded search path support and support for most search comparators. Support for search directives. Retrieval of current allergies, conditions, medications. Retrieval of medical history.
3	FHIR resources related to pseudonymised patients using local coding systems.	Expanded search path support. Asynchronous searching.
4	Read and search access to appointments and associated resources. Manage referral process through the creation and management of appointments.	Support for all search paths defined by STU3. Creation and update of a small set of FHIR resources. Advanced search comparators. Use of tags in resources

5	Production of agreed compositions (FHIR documents) to regional standards and registration with regional document registry. Support for subscriptions.	Patching FHIR resources. Searching based on the last update time of a resource. Support for chained search parameters.
6	Pseudonymised FHIR resources coded to SNOMED-CT	Application of local consent rules. Support for peer to peer relationships which bypass the regional FHIR Bus.

2 Requirements Specification

The FHIR Proxy server enables compliant client software to interact with one or more clinical data sources via a REST endpoint. Compliance means implementing a minimum subset of the API features which are defined by: http://www.hl7.org/fhir/STU3/http.html. The minimum feature set is defined by maturity level. The API interacts with data sources using resources. The resources supported at each maturity level and the clinical; concepts to which they map are defined by design paper 022 – "FHIR Resource Profiles". The requirements specified here treat a resource as an abstract concept.

2.1 REST Endpoint Specification

The endpoint is offered over mutually authenticated HTTPS and must be secured by a certificate signed by the YHCR certificating authority (design paper 016 – "Securing the YHCR").

Support for the following HTTP verbs is required:

Verb	Function	Maturity

GET	Retrieve an individual resource or search for resources which match a search specification.
	
	`GET /Patient/172364367`
	- retrieves patient # 172364367.
	
	`GET /Patient?family=Brown`
	- searches for patients with a family name which starts with 'brown'.
		1

POST	Creates a resource.
	
	`POST /Appointment`
	- creates an appointment. The HTTP body contains an appointment resource.
	
	`POST /Appointment?subject=nhs|1234567890&date= 2013-01-14T00:00`
	- creates an appointment if an appointment does not already exist for the patient on this date.
		4

PUT	Updates a single resource.
	
	`PUT /Appointment/638391222`
	- updates appointment #638391222. The HTTP body contains the updated resource.
	
	`PUT /Appointment?subject=nhs|1234567890&date= 2013-01-14T00:00`
	- updates the appointment for the patient on this date.
		4

PATCH	Partially updates one or more resource. `PATCH /Appointment/638391222`	5

	- updates some of the properties of appointment #638391222. The HTTP body contains a patch document. `PATCH /Appointment?patient=/Patient/73537388` - updates some of the properties of all appointments for patient #73537388. The HTTP body contains a patch document.	

Note that DELETE operation is not required for the YHCR.

The body content of HTTP interactions can be either JSON or XML (from maturity level 1. Only JSON patch documents are supported for the patch operation.

Each of the above operations can be modified using directives. Requirements to support optional directives is detailed in the following sections.

HTTP response codes are specified throughout the FHIR API specification and are not repeated here.

2.2 Resource Identifiers and References

The choice of a resource identification scheme is left to the implementor with the following conditions:

- each clinical concept must have a unique resource identifier (for its resource type)
- each clinical concept should have a single resource identifier
- identifiers must be immutable (ie: a clinical concept should be represented by the same identifier for all time).

Here, a clinical concept is used as a generic term for elements of a patient’s medical record and includes, test results, medications, assessments, care plans, diagnoses etc. Mapping of clinical concepts to FHIR resources requires a more comprehensive treatment than is possible here. Some consideration is given to it by design paper 023 – “YHCR FHIR Resource Profiles” but it is an ongoing subject which will be controlled by the Data Architecture Design Authority.

Resource references should be presented as a contained model as per the definition in design paper 001 – “A Unified Distributed Data Model for FHIR”.

Resource references may be absolute or relative. Resource references using business identifiers (an example being an organisation’s ODS code), whilst permitted by the FHIR specification, should be avoided for use in the YHCR.

2.3 Searching for Resources

The sophistication of searching capabilities is closed tied to the maturity model.

Basic searching using a set of resource specific pre-defined search terms is required at maturity level 1. The set of support terms will expand over maturity levels 2 and 3. Search term coverage is defined by design paper 022 – “FHIR Resource Profiles”.

At maturity level 4, implementors are required to offer generic searching across all search terms for supported resources as defined by the FHIR STU3 standard within a single resource. Chained parameters, where search parameters traverse resource references, is required for maturity level 5 (being essential for functionally useful subscription capability). Support for chained parameters will only be mandated across a single resource reference.

Searches are only required against a single resource type. So, the search:

```
/DiagnosticReport?patient.given=john
```

is supported but

```
/DiagnosticReport?patient.organization.name=LTH
```

is not.

There is no requirement to implement cross system searches for all resources with overlapping search parameters.

Support for reverse chained search parameters is not required.

2.3.1 Search Comparators

Support is required for basic equivalence and boolean operators from maturity level 1. So, for example, a data consumer may request the following:

`GET /Patient?address-postalcode=HG4`
- patients with an address in a postcode starting 'hg4'.

`GET /Patient?address-postalcode=HG4&deceased=true`
- patients with an address in a postcode starting 'hg4' who are deceased.

`GET /Patient?address-postalcode=HG4,LS1`
- patients with an address in a postcode starting 'hg4' or 'ls1'.

This final construct is a commonly adopted convention rather than a requirement or the FHIR specification.

The interpretation of a search comparators varies for the type of property on which it operates. Support is specifically required for the following property types:

- Numbers.
- Dates and times.
- Strings.
- Tokens.
- References to Other Resources.
- Composites.
- Quantities.
- URIs.

2.3.2 Tokens and Composites

Tokens are combinations of a coding system and value. The FHIR specification offers syntax which enables searches to be performed against combinations of the system and value. The YHCR requires support for the syntax `parameter=system|code` for Coding, Codable Concepts, Identifiers, and Contact Point data types. The system is implied for code and boolean data types and for these the syntax `parameter=code` is required.

Composites are an amalgamation of elements of a structure which are referenced in a search term as a single condition.

Support for the above property types is required at the maturity level where a search terms requires it as specified in design paper 022 – "FHIR Resource Profiles".

2.3.3 Search Prefixes

Search prefixes expand on basic equivalence to provide support for other comparator operators. The following operators are required for maturity level 2:

• eq:	equals
• ne:	not equals
• gt:	greater than
• lt:	less than
• ge:	greater than or equal
• lt:	less than
• ge:	greater or equals
• le:	less than or equals
• sa:	starts after
• eb:	ends before

Note that the interpretation of these operators depends on the property type and implementors are advised to refer to the HL7 STU3 specification. Particular emphasis is placed on achieving the correct interpretation of comparators for dates.

Note that support is not requires for ap: approximately.

2.3.4 Search Modifiers

Search modifiers also alter the behaviour of a search term. Support is required for the following modifiers:

Modifier	Interpretation	Maturity
:exact	Match a string exactly	1
:contains	Match a string that contains the search term	4
:[type]	Specify the type of a reference to be used on a match	2
:missing	Match a parameter that is present or not present	1
:text	Targets a comparator at the text part of a *CodeableConcept*, *Coding* or *Identifier* property.	1

2.3.5 Using Lists in Searches

Lists can be referenced in search terms in two circumstances which have coincidental syntactical similarities but have very different interpretations:

1) As a technique for obtaining all resources referenced in by FHIR list resource.

```
GET /Patient?_list=1234
```
- returns all patient resources referenced by list 1234

2) As a grammatical embellishment to specify that current allergies, medications, or problems should be returned for a patient.

```
GET /AllergyIntolerance?patient=42&_list=$current-allergies
```
- returns a patient's current allergy intolerance list

Only the second use of the _list directive is required by the YHCR. This is aligned with maturity level 2.

There is another potential use for *Lists* which is not provided for by the FHIR specification. Lists are often used to represent cohorts of patients. In the area of subscriptions, it is a very frequent requirement for a clinician to wish to subscribe to events pertaining to a cohort of patients (e.g. subscribing to ED encounters for patients under my care). This requirement can be met, validly within the constraints of FHIR syntax, by lodging a subscription for each patient in the cohort. A more convenient method might be to register a subscription against a patient list which is maintained regionally but, unfortunately, there is no standard syntax which allows this to be expressed in FHIR. A possible extension to the FHIR STU3 search syntax YHCR FHIR Resource Profiles allow for this is explored in design paper 007 – "Subscription Infrastructure".

2.3.6 Searching Narrative

Free text searching in the narrative of a resource (using the _text and _content annotations) has unresolved issues and is difficult to imagine operating consistently and reliably at a regional level. Text searching is not required by the YHCR at this time.

2.3.7 Search Content Directives

By default, the search result is a bundle which contains full resources in an indeterminant order. The following directives modify this behaviour and are required by the YHCR.

Directive	Interpretation	Maturity
_sort	Resources in the bundle are sorted using the search terms listed in the _sort expression.	2
_count	Paginates results and returns a maximum of the specified _count with references to pervious and next page urls.	2
_include	Includes referenced resources of specified types. The :recurse modifier applies the inclusion recursively to included resources. Implementors can choose to limit the number of recursions. If doing so they should behave as specified by FHIR STU3.	2
_revinclude	Includes resources of the specified types which reference resources in the result set. The :recurse modifier applies as above.	2
_summary	Returns only those properties of a resource required for the summary view.	2
_elements	Specifies the properties of the resource to include in the search results.	2

The recursive elements of the above should be supported from maturity level 4.

The effect of the search directives _contained and _containedType can be achieved using the _include directive and so support is not required by the YHCR.

2.3.8 Resource Tags

Tags are meta data about resources which associates them to a position in a workflow. Whilst there is no design for a tag system at this stage, it is likely that use cases will emerge when addressing co-ordination of care objectives which are aligned with maturity level 4. Search using the _tag annotation should be included in a implementors roadmap.

2.3.9 Other Search Features

The ability to search on the last update time of a resource is closely correlated with implementation of subscriptions is required from maturity level 5.

```
GET /Observation?_lastUpdated=gt2017-01-01
```
- retrieves observations modified after 1 January 2017

The following annotations are specified by FHIR STU3 but are not required by the YHCR at this time.

_profile	Searching against meta data assertions that a resource belongs to a particular profile, the application of which will be explored alongside emerging use cases.
_security	Searching against meta data connections to a security policy, the application of which will be explored alongside emerging use cases.
_has	A mechanism for reverse chaining the effect of which can be achieved (albeit slightly less efficiently) the _include directive.
_type	A mechanism for restricting non-resource specific searches to specified resource types.
_query	Enables a complex search term to be expressed the application of which will be explored alongside emerging use cases.
_filter	An alternative method of providing a search string.

2.3.10 Asynchronous Searching

For the purposes of direct care, most data consumers will be patient-centric user interfaces which will acquire data on-demand for specific patients in sufficiently small quantities for them to be easily assimilable by an end-user. Such consumers will be targeting searches at a small number of data points or will use pagination to reduce the amount of data returned in any one query. Assuming data sources are well indexed, such searches will return in real time and the results will be included, synchronously, in the HTTP body of the response to the search request.

With the advent of requirements for population health management, a different class of search can be anticipated: searches which are intended to acquire data in bulk. These could place a significant computational load on the data provider were it to attempt to service the search in real time. Search result volumes may also be too great to reliably return in a single HTTP response body. For this class of query, a data consumer should use asynchronous searching. The FHIR standard adopts an approach offered by RFC 7240. This allows a consumer to request a search to be performed but for the response to be generated at a convenient time for the data provider and to be picked up by the consumer at some later point in time. FHIR also specifies the mechanism for a consumer to interrogate the status of an asynchronous request and to receive the results in one or more batches. The precis of the processing requirements here are not intended as an extension to the FHIR standard and the standard should guide implementations.

1. A consumer requests asynchronous processing by including the following header in the HTTP request:

   ```
   Prefer: respond-async
   ```

2. The FHIR proxy responds immediately with a URL for the consumer to interrogate for status updates in a HTTP response header as follows:

```
Content-location: [url]
```

3. The consumer periodically interrogates the service at the above URL and the response code indicates the progress status of the request.
4. When the request has completed the progress status service returns in the HTTP body a list of URLs from which fragments of the response can be obtained.
5. A bundle of search results can be obtained from each of these URLs.

The approach to implementing asynchronous processing is at the discretion of the data provider but it is anticipated that most approaches will involve:

- placing the request on a queue;
- paginating or segmenting the request search in some way;
- executing searches at times of low processing load;
- purging search results once the consumer has collected them.

A data provider can refuse a synchronous search request which should be executed asynchronously by issuing the HTTP response:

```
504 Too Expensive
```

The ability to search in bulk is a prerequisite for the population health management workstream and is aligned with working with pseudonymised data at maturity level 3.

2.4 Creating and Modifying Resources

One key objective for the LHCRE is to improve collaboration of care across care settings. At maturity level 4 it is anticipated that a data provider will be able to offer certain items for collaborative management to users in other care settings and this point the FHIR endpoint becomes bi-directional.

For local care letting t is likely that this capability will be restricted to a small section of the patient record, and will focus on workflow and patient administration. Such topics typically involve the management of Tasks, Appointments, ProcessRequests , ProcedureRequests and ReferalRequests.

Data management at a regional level is a broader topic and the ability manage *Questionnaires*, *Care Plans*, current *Conditions* etc. affords many opportunities to improve collaboration in care. However, responsibility for management of these data items may take place at a regional level and local capability is not mandated.

It is well recognised that that there are not many fully functional inbound interfaces into the regions Electronic Patient Record or Client Management Systems and care settings will be limited in their ability to automatically process inbound resources. At a minimum capability a data provider, at maturity level 4, should be able to receive and store a defined set of FHIR resources. It should offer management functionality through updates or patches to a set of properties defined by a corresponding FHIR management resource profile. Resources created from outside an organisation’s boundary should be available to users in the organisation through a portal or other user interface and business processes established to action this data.

2.5 Securing the REST Service

A FHIR endpoint may be used both by local applications and by the YHCR and may operate different security mechanisms for each. The YHCR does not concern itself with local security policies but does stipulate that any connections the FHIR endpoint from outside the local network must contain an HTTP Authorization header of type 'Bearer'. The header value must be a JSON Web Token (JWT) which is signed by the YHCR Identity and Access Management Service (a public key will be provided by the YHCR). At a minimum the FHIR proxy must validate that the JWT is active (based on the expiry date embedded in the JWT). Optionally, the FHIR proxy may use other properties of the JWT to restrict access (the User Identity, User Role, User Organisation). The format of the JWT is described by design paper 005 – "Identity and Access Management Server".

Optionally, the FHIR Proxy may use a service implemented by IAM to validate JWT. Superficially the service offers an additional security measure in that ii is able to detect a misappropriated JWT that has been revoked. Although, in practise, discovery of misappropriation is unlikely within the lifetime of a JWT (probably 15 minutes) and the use of this service is discouraged for performance reasons.

The FHIR endpoint must be secured for external access by a firewall. At maturity level 1 the firewall must only permit external access from the IP address of the regional FHIR bus. At maturity level 6 the firewall may permit access from other accredited peer data consumers.

2.6 Audit

The FHIR proxy must audit:

- attempted security breaches.
- resources served, created, and modified.
- subscription fulfilment (see design paper 007 – "Subscriptions Infrastructure")

Audit logs can either be persisted locally or at the regional FHIR store. The former is preferred for performance reasons.

Locally persisted audit logs must be accessible as *AuditEvent* FHIR resources via the FHIR proxy using standard FHIR retrieval and search interactions.

The content of the *AuditEvent* resource and a definition of the minimum support for search terms is in design paper 009 – "Auditing".

2.7 Consent

A full discussion of the operation of consent management is left to design paper 008 – "Consent Management". Here, focus is placed on the processing requirements of the FHIR Proxy which allows consent to be enforced at a data provider's boundary.

From maturity level 1, consent rules will be applied by the regional FHIR Bus and data released through requests serviced by regional infrastructure will comply with the consent wishes recorded by a patient on the regional consent server or national opt-out service.

Applying consent rules at a data provider's boundary enables:

- Local consent policies to be applied that override or supplement regional policies.
- Peer to peer relationships whereby data consumers interact directly with a data provider and bypass regional infrastructure.

Local consent management is required at maturity level 6.

Consent is modelled using FHIR *Consent* resources. In broad terms the *Consent* resource records a patient's opt-in or opt-out of a consent policy. The policy describes:

- The situation to which it applies such as role of the clinician accessing the record, the organisation from which it is accessed, o the reason for accessing the record (these factors can be derived from properties of the JWT).
- The data to which it pertains (a set of FHIR search paths).
- The precedence the policy takes in relationship to other policies.

Policies can be accessed using a REST service implemented by the regional Consent Server. A FHIR Bus should maintain a local cache of regional consent policies which is refreshed according to a periodicity specified in the YHCR Operations Guide.

Local policies may further constrain regional policies. It is the responsibility of the locality to define how local policies interact with regional policies and how precedence applies where there is conflict.

A FHIR Proxy must undertake consent processing when local policies are in place or when interacting directly with a peer data consumer without using the regional FHIR Bus as an intermediary.

Consent processing may result in resources being withheld from a search result set or a resource retrieval. The consent policy will determine whether information is provided to the service user that resources were withheld. Depending on the configuration of the policy, the fact that data has been withheld may be included, as meta-data in the result set resource bundle, and for a resource retrieval in the HTTP response message. Further detail is provided by design paper 008 – "Consent Management".

Consent processing acts on a set of resources which would otherwise be released from a FHIR endpoint. Logically, each resource in the set is processed as follows:

i. Identify the patient that is the subject of the resource.
ii. Query the regional FHIR bus for Consent resources which pertain to the patient.
iii. Query the local consent repository for Consent resources which pertain to the patient.
iv. Merge policies referenced by regional and local resources and order by precedence.
v. For each policy and for each FHIR search term in the policy definition, if it references the current resource type then construct a search which includes the resource id and execute the search against the local FHIR endpoint.
vi. Depending on the search result, the patient's opt-in/opt-out status for the policy and the policy definition then the search will indicate whether the resource should be released.
vii. After executing all searches for all policies, a final determination can be made whether to withhold the resource.

2.8 Accessing Data

The role of the FHIR Proxy Server is to front one or more data sources with the technical FHIR implementation describe above. When responding to HTTP GET requests the server will extract data from one or more data sources and map the data to the appropriate resource structure. At early stages of maturity, the mapping will be simple, and local coding system will be the basis for codable concepts. At later stages, local coding systems will be mapped to SNOMED-CT, DM+D and other regional/national vocabularies. The level of de-duplication of data will also increase over maturity levels and the FHIR Proxy will be expected to become increasingly selective over its choice as to from where data is acquired.

The strategy that a FHIR Proxy uses to acquire data is in the hands of the implementor. The following approaches, or combination of these, can be imagined:

- FHIR resources are constructed on demand from a relational database. Resources identifiers may to a primary key on a table from which a query can be constructed to assemble data for the resource. Search terms for resource map to clauses in a query and resources can be assembled from the query results.
- A batch extract of data is periodically taken from a system and an Extract-Transform-Load process constructs FHIR resources which a re stored in a document database with well indexed search terms.
- Event driven transactions emitted from a system are processed in an integration engine and transformed into FHIR resources which are stored in a document database.
- An API is called on a source system in response to a search or retrieval request. Data obtained in manner is converted to FHIR resources.

2.9 FHIR History and Versioning

Having the ability to recreate a view of a patient record presented to a clinical at a point in time is considered a critical feature for the YHCR and this requires tracking of FHIR resource versions. From maturity level 1 a FHIR Proxy server is required to populate the VersionId of the resource meta structure. The VersionId should only change between consecutive resource retrievals if one or more of the properties of the resource has changed.

The Proxy server is required to support the version specific resource retrieval:

```
GET /Patient/8464394/_history/23
```
- retrieve patient # 8464394 version number 23

FHIR allows for any standards compliant identification scheme to be used as a version identifier. The YHCR requires the version id uses integers which are consecutively allocated to the resource being served.

At this time, there is no requirement for a FHIR Proxy to support the history interaction.

The approach taken to implementing FHIR versioning is left to the implementor. An approach, which is expanded on the conceptual design, is to persist FHIR resources which are served by a FHIR Proxy. Prior to serving a new instance of the resource, the FHIR proxy compares it to the last instance persisted and if there is a difference then uplifts the version number and re-persists the resource.

Responsibility for reproducing the view of the patient record seen at a particular point in time lies with the data consumer. Of course, this can be an ad-hoc exercise, triggered by a particular investigative need rather than a feature of data consuming software. The exercise would be supported by the audit record of resources served which are available from data providers and the regional FHIR aggregator

2.10 Subscriptions

The ability to subscribe to a data point in a locality and support for a messaging of data which match the subscription are crucial capabilities for the YHCR but are of a breadth and depth which necessitates consideration being deferred to other documents: design paper 006 – "Messaging Infrastructure" and design paper 007 – "Subscriptions".

Support for subscriptions was assigned by the Abstract Cookbook to maturity level 5. However, preliminary work with pilot organisations has identified it as a capability which is needed to support some basic use cases and the expectation will be that data providers offer this facility much earlier in the evolutionary cycle.

2.11 Transactions and Batches

Support is not mandated at a local level for transactions and batches.

2.12 Publishing Capabilities

Data providers will register their maturity with a regional service and the FHIR aggregator will use this information to mediate with data consumers. These topics are detailed in design papers 020 – "Onboarding Data Providers" and 010 –"FHIR Aggregator".

Data providers will also publish a CapabilityStatement from the FHIR Proxy. The capabilities published here must be at least conformant with the capabilities required for their registered maturity level.

3 Conceptual Design

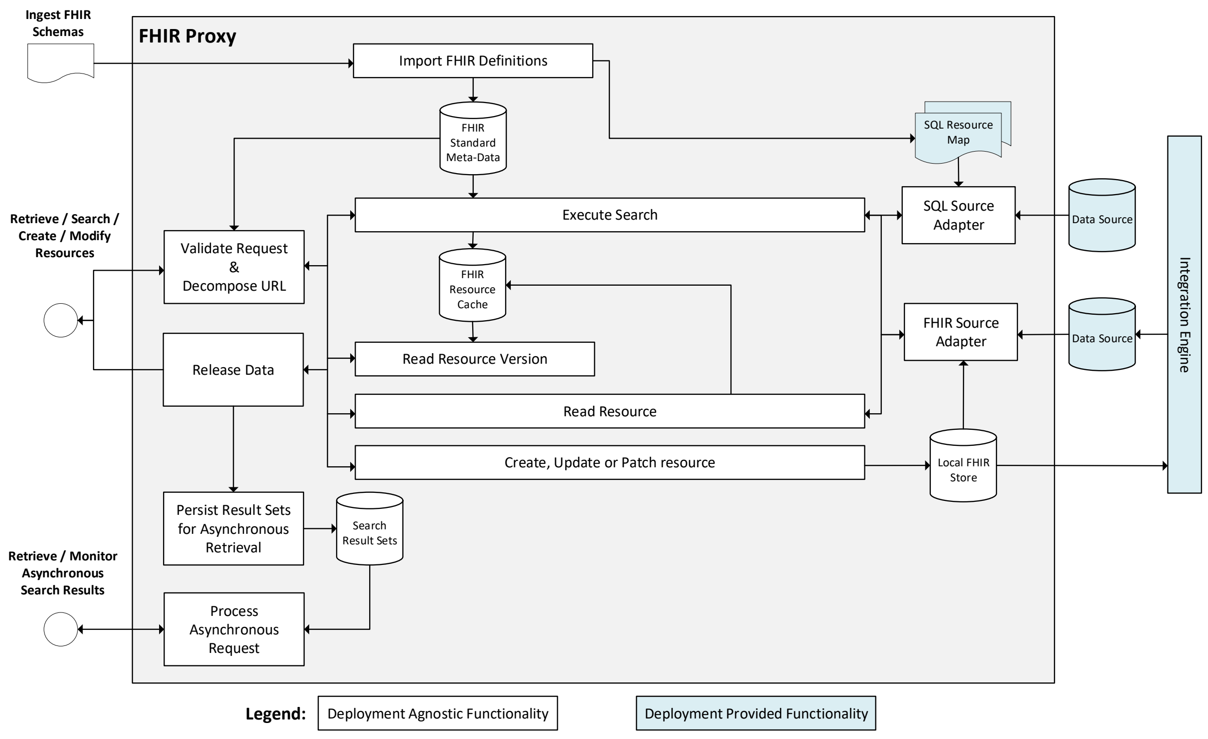
The following sections detail a conceptual design for a FHIR proxy. This design is intended to inform the development of model software. The primary intent of model software is to abstract its users from the technical implementation of the FHIR standard and to focus their effort on mapping data from local data sources into FHIR resources.

The design assumes that the FHIR Proxy is being used to front one or more data sources. A data source might by synonymous with a local system, such as a radiology system, or may the result of an internal aggregation process, such as a data warehouse. Data sources might be updated in real-time or be refreshed periodically.

It is assumed that each data source is either as a SQL database or a FHIR document database. Clearly there are many systems for which neither of these options are available and, in these circumstances, it is assumed that integration technology could mediate between the capabilities of the system and the requirements of the FHIR Proxy. So:

- Transactions from systems which emit event triggered HL7 messages could be routed through an integration engine, disassembled, and inserted into a SQL database which is the source of data for the FHIR Proxy.
- Interfaces to systems which offer an API through which data can be accessed on-demand could be invoked at the point that data is requested, and the data converted to a FHIR Document format which is the source of data for the FHIR Proxy.

The design is based on the following processing model.



Ingest FHIR Schemas
FHIR Proxy
Import FHIR Definitions
FHIR Standard Meta-Data
SQL Resource Map
Retrieve / Search / Create / Modify Resources
Validate Request & Decompose URL
Execute Search
SQL Source Adapter
Data Source
FHIR Resource Cache
FHIR Source Adapter
Data Source
Integration Engine
Release Data
Read Resource Version
Read Resource
Create, Update or Patch resource
Local FHIR Store
Persist Result Sets for Asynchronous Retrieval
Search Result Sets
Retrieve / Monitor Asynchronous Search Results
Process Asynchronous Request
Legend:
Deployment Agnostic Functionality
Deployment Provided Functionality

In this model the FHIR Proxy presents two HTTPS listeners. The first is a standard FHIR endpoint which interacts with FHIR resources through GET POST, PUT and PATCH requests. The second supports asynchronous interactions, again, according to the FHIR standard.

Requests on the listeners are serviced through various message “pathways”. These ultimately result in an interaction with external data sources and/or data which is held internally to the FHIR Proxy.

The precise processing performed in message pathways depends knowledge about the specifics of the FHIR standard. Knowledge of:

- resource structures are needed to map data.
- search term definitions are needed to interpret search strings.
- code system definitions are needed to validate inbound data.

FHIR is self-describing: FHIR resources are used describe elements of the standard itself. Schemas are also published which describe the structure of FHIR resources. The processing model described here uses the metadata about the standard to abstract processing details from the definition of the standard. Processing is defined independently from the particulars of the standard and, to some extent, is future proofed against changes to the standard.

A key objective of the model is to segregate core functionality from implementation detail. Most of the units of functionality are standard and can be used without modification by different deployments. Specifics of the data sources over which the FHIR Proxy will operate are clearly important to local implementation but the design strives to isolate data mapping work to small well-defined parts of the overall solution.

Data sources might be databases which native to applications or databases which are built from transactional feeds flowing through an integration engine. The model is bi-directional: resources can be created and modified. However, these actions act only on a FHIR Store which is local to the FHIR Bus. If the effect of these operations needs to propagate out to other systems, then the model assumes that an integration engine will monitor the local database and take appropriate action as data changes.

The model FHIR Proxy will support *Subscriptions* and Messaging. Conceptual design for these concepts are presented in the design papers 006 – “Messaging Infrastructure” and 007 – “Subscriptions”.

3.1 Import FHIR Definitions

This pathway reads FHIR schemas and other metadata files and populates internal data structures. The metadata, which reflects Y&H FHIR profiles, will be published in JSON format by the Data Architecture Design Authority (DADA. These documents being profiled versions of those published by HL7 itself.

Schema files provide explanatory narrative and describe the properties of resources and the data structures which are used by resources. Properties are specified in terms of:

- a descriptive narrative;
- its data type (or for an array, the type of its elements);
- a possible list of values if appropriate;
- any constraint of the data content.

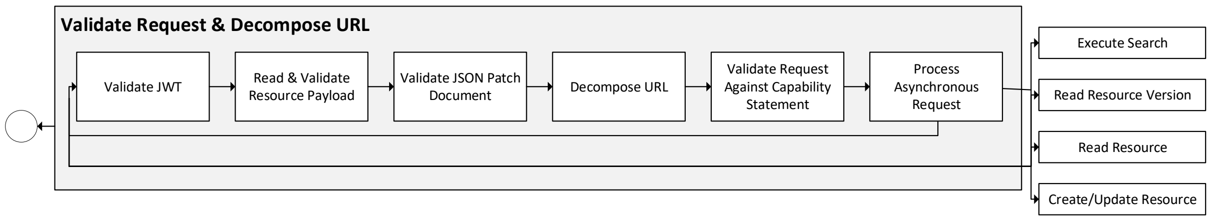
The model implementation of the FHIR proxy uses resource schemas to create template SQL resource maps which can be used for basis of data mapping to a data source (see below). Schema definitions are also used for validating the content of resources created or modified by the FHIR Proxy.

Other important metadata files define the mapping of search terms into properties (which are used for the interpretation of a search request), and value sets and code tables (for use when validating the content of FHIR resources).

Importing a schema importing is an occasional operation.

3.2 Validate Request & Decompose URL

This pathway performs security and sanity checks which are common to all other processing pathways.


Validate Request & Decompose URL
Validate JWT
Read & Validate Resource Payload
Validate JSON Patch Document
Decompose URL
Validate Request Against Capability Statement
Process Asynchronous Request
Execute Search
Read Resource Version
Read Resource
Create/Update Resource

3.2.1 Validate JWT

i. Verify the JWT signature using the YHCR public key.
ii. Verify the expire date on the JWT.
iii. Validate the structure of the JWT.
iv. Optionally invoke the regional Identity and Access Management Server's Validate JWT method.

A request failing JWT validation is immediately rejected with a 401 response code and an *AuditEvent* resource written to the local FHIR store (not shown in the processing model).

3.2.2 Read and Validate Resource Payload

i. For PUT and POST operations (to create and update resources), extract the HTTP body and de-serialise from XML or JSON into a language neutral graph representation.
ii. Validate that the payload contains a single resource and the resource type is supported.
iii. Validate that the resource structure corresponds to the resource schema by iteratively traversing branches of the graph and validating properties in turn against corresponding schema definitions. The property must exist in the resource or structure definition, be of the correct data type, validate against a content definition or be a member of the specified value set.
iv. Validate mandatory properties in the resource schema by iteratively traversing the branches of the graph structure which describes it. For mandatory properties validate the existence of a corresponding property in the resource payload.

A request failing validation is immediately rejected with a 400 response code and an *AuditEvent* resource written to the local FHIR store (not shown in the processing model).

3.2.3 Validate JSON Patch Document

i. For PATCH operations which are used to selectively update one or more resource properties, extract the HTTP body and de-serialize into a language neutral graph representation.

Validate that the object represents a patch document with an array of instructions composed of an operation path and value.

ii. Iterate through the instructions and validate the operations, that paths exist in the resource schema definitions, values comply with format constraints and value sets, and that remove operations are not operating on mandatory parameters.

3.2.4 Decompose URL

i. Extract the resource path, search string, search modifiers and directives from the URL and populate an internal data structure.

3.2.5 Validate Request Against Capability Statement

i. Validate that the requested operation is supported within the implementation of the FHIR Proxy.

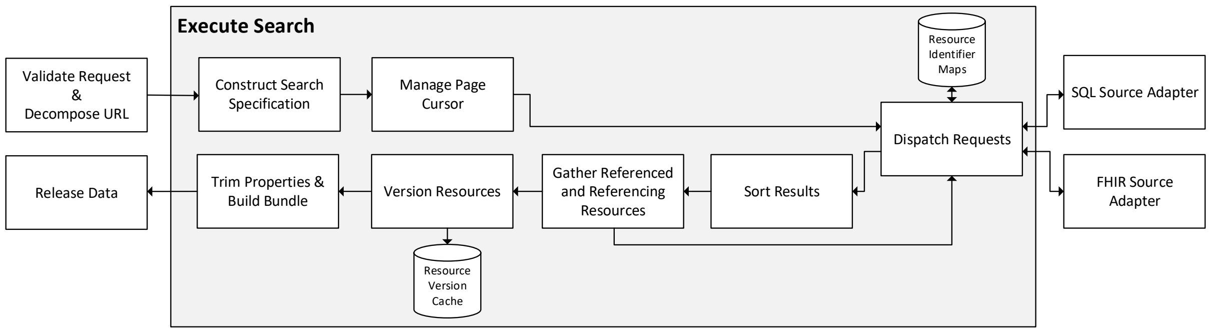
3.2.6 Process Asynchronous Request

If the request indicated that an asynchronous response is preferred, then:

i. generate a unique identifier for the request.
ii. respond immediately with a 202 HTTP response code and content location constructed from the URL of the asynchronous processing endpoint and the unique request identifier.
iii. add an appropriate pagination to the decomposed URL.
iv. pass the request to the Execute Search pathway for processing.

3.3 Execute Search

A search request is executed across all data sources (including a FHIR store which is internal to the FHIR Proxy). Results are amalgamated and processed according to search directives. A bundle containing an array of serialised resources is returned.



3.3.1 Construct Search Specification

The search string in the invoking URL makes use of search terms which are defined in the FHIR specifications. Search term mapping to FHIR properties (and conditions on properties) is defined in the metadata imported in section 3.1.

i. Populate a language neutral representation of the search by decomposing the search string into an array of comparison.
ii. Express each comparison in terms of a condition of a resource property using the FHIR standard metadata. For chained search terms collect resource join conditions.

Similar processing is performed for the _sort directive to relate search terms to properties.

3.3.2 Manage Page Cursor

Searches involving the :count directive are returned as discrete pages of results. If the FHIR proxy is operating over more than one data source, then the FHIR proxy will need to operate a cursor: an data structure (possibly in memory) that queues data collected from primary sources. Backward page movements can be served directly from the cache without returning to the data source. Forward page movements may be serviceable from the cache if there is sufficient data in the cache from each data source to guarantee that search order will be preserved.

If there are insufficient results in the cache then the query is passed to primary data sources and returned sorted results merged into cache before a subset are returned as a page.

Page cursors are expunged if not used for a period of time.

3.3.3 Dispatch Requests

Queries are dispatched to each of the data sources which are being fronted by the FHIR proxy. Specifically, this includes the local FHIR store. Queries are dispatched in parallel to optimise performance. Data sources are interrogated using an adapter (described below) which returns resource references as business identifiers. For most resource types the business identifier is local to the data source and has no generic meaning. For instance, a reference to an encounter may be the ROWID or primary key of the encounters table in the data source. The exceptions are concepts which should have common representation across all data sources. Examples include:

- patient;
- organisation;
- practitioner;
- episode of care;
- location;
- organisation.

The precise list of concepts which can be standardised across data sources (common resources) will vary between deployments and a generic implementation may choose to make shared resource concepts configurable.

For shared resource concepts, the business identifier has a common interpretation. Examples include a patient hospital number or a code belonging to a generally used vocabulary such as DM+D.

Translation of Business Identifiers to Resource Identifiers

The Dispatcher translates business identifiers into resource identifiers and replaces business identifier references with a relative resource reference URL.

The translation is performed using a simple map:

```
DATA SOURCE + BUSINESS IDENTIFIER => RESOURCE IDENTIFIER
```

Resource identifiers are created, and new entries inserted into the map as needed.

A reverse mapping is also maintained:

```
RESOURCE IDENTIFIER => DATA SOURCE + BUSINESS IDENTIFIER.
```

For common resources, the data source is not necessarily the data source which served the reference. Each common resource is associated with a ‘home’ data source which is used instead of the referencing data source in the map. The implication is that common resources will always be served from same data source regardless of the data source that refers to them.

For example, a pathology test result may be served as a *DiagnosticReport* from a laboratory system. The *DiagnosticReport* refers to a patient using an NHS Number. The source for patient resources is patient administration system (PS) and the resource maps link the patient's unique resource identifier to the PAS.

The operation of the dispatcher ensures that search results are always returned with resource references expressed as URLs which can be serviced by the FHIR Proxy. Referencing URLs embed a resource identifier which is mapped to a data source in the resource map. For common resource types the mapped data source is a nominated 'master' source. For other resource types it is the data source from which the search result was obtained.

3.3.4 Sort Results

The data source adapter results are sorted according to the search terms specified in the request.

3.3.5 Gather Referenced and Referencing Resources

Additional resources may be added to the search result bundle through the use of the _include and _revinclude directives.

The search result set is traversed and referenced and referencing resources are retrieved by placing additional requests on the Dispatcher. The process may be recursive (if the :recurse modifier was specified in the search) and additional results returned from the Dispatcher are traversed up to a configurable number of recursions.

3.3.6 Version Resources

This module stamps version numbers onto resources. It maintains a cache of all resource versions released by the FHIR bus. For each resource in the result set, it determines whether it has properties which are identical to those for a previously released version. If not, then it is a new version and the version number is incremented and the resource version is persisted in a resource cache.

A policy for resource version retention will be published in the YHCR Operations Guide. The time period for retention is likely to be expressed in years.

3.3.7 Trim Resource Properties and Prepare Bundle

Properties may be explicitly requested using the _elements directive or implicitly using the _summary directive. If _summary is specified, then the model assembles an element list from the metadata loaded in 3.1.

i. For each resource in the search result set (as distinct from resources added to the result set from the _include and _revinclude directives) trim properties to those referred to be the elements list.
ii. Construct a bundle structure and populate with resources.

3.4 SQL Source Adapter

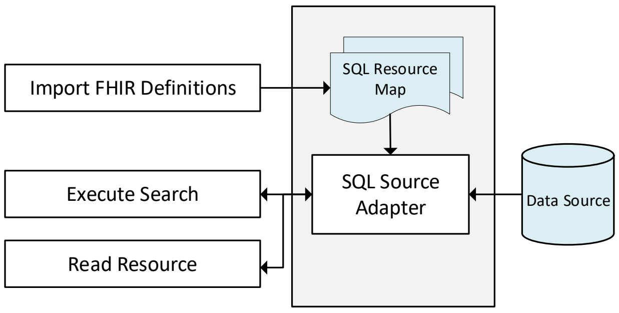
The SQL Source Adapter reads and searches for data in an SQL database and transforms results into FHIR resources. The processing model described here aims to isolate deployment specific configurations from standard functionality which applies to all deployments.

Note that the SQL Adapter will be expected to service requests for a single resource and searches composed of boolean operations using search paths and comparators with absolute values. Processing of more complex search modifiers such as _includes and _revincludes has been placed downstream in the processing model at, for instance, "Gather Referenced and Referencing Resources".

The logic to transform from SQL tables to FHIR resources is entirely dependent on the on the database schema and inevitably work is needed to apply the model implementation to the specifics of a local application's database.

In some cases, the logic will be complex, and SQL Source Adapter will act solely as an abstract class which defines placeholders for functionality which is implemented by the deployment team through programme code.

In other situations, mappings between SQL columns and FHIR properties may be relatively straightforward and can be expressed in simple expressions from which SQL statements can be constructed using a common framework. The overarching processing model depicts this latter case.



SQL Resource Maps describe for individual resource types:

1) For each property of a resource, the table columns need used to compute the resource property and a definition of the computation which needs to be applied to them.
2) For each search term, a SQL clause which represents the condition and a definition of a computation which needs to be applied to a search parameter in the clause.
3) The tables, or tables which form the basis of derived SQL statements with associated join conditions.
4) A SQL clause which can be used to select a resource from its business identifier and the computations which are applied to the identifier for use in the clause.

The computations defined in the map use library functions for performing standard data operations such as applying a code table translation. Library functions are extensible for deployment purposes.

The maps are used by the SQL Source Adaptor to generate SQL on demand for

- reading a resource given its business identifier;
- searching for resources given a search specification.

The "Import FHIR Definitions" module produces a template map for each resource which is completed by the implementor for each SQL Adapter.

A number of different SQL Source Adaptors can be deployed against different databases, with each being configured with an ODBC or JDBC connection string.

3.5 FHIR Source Adapter

This component represents a set of adaptors, each targeted at a specific NoSQL Database such as:

- MongoDB;
- InterSystems Caché;
- YottaDB.

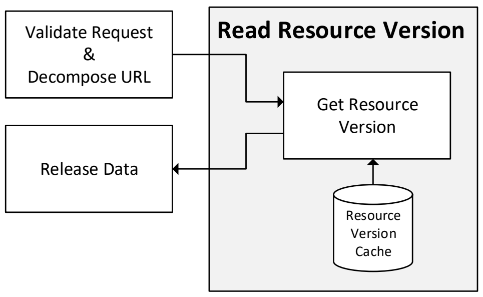
The adaptor provides methods for reading a resource from its business identifier (which in this case is its local resource identifier), and for executing a search from a search specification.

The search algorithm is constructed in a technology specific manner based on the native searching capability of the database.

A FHIR Adaptor is used to interface with a local FHIR store. In this instance, the adaptor also supports create, update and patch operations.

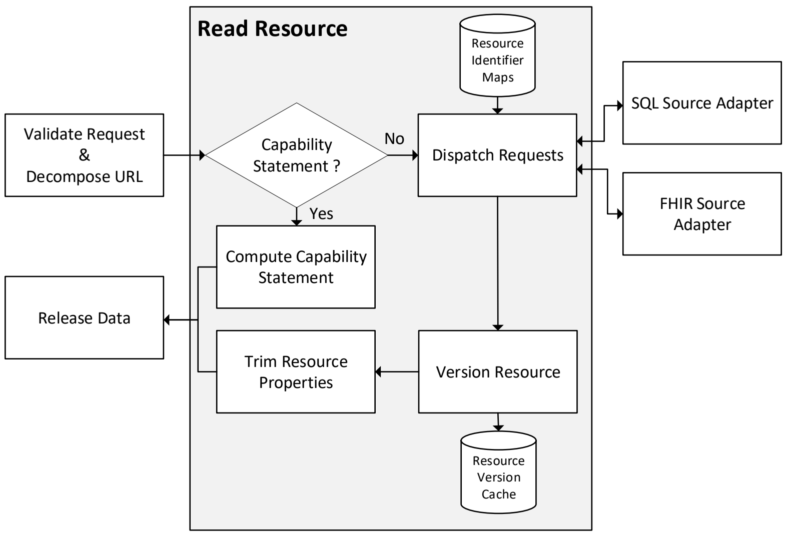
3.6 Read Resource Version

Resource versions are read from the resource version cache which is implemented locally by the FHIR Bus.



3.7 Read Resource

This pathway processes a read request and returns the FHIR resource which corresponds to the resource identifier which is included in the request.



3.7.1 Compute CapabilityStatement

A special use case for the read resource pathway is serving a *CapabilityStatement*. The content *CapabilityStatement* can be computed from the configuration of the FHIR Proxy and never persisted. The FHIR Proxy generates this resource on demand.

3.7.2 Dispatch Request

This module was introduced in section 3.3.3. When processing a read request, the module uses the Resource Identifier Map to determine the data source from which the resource may be obtained and the business identifier. The request is dispatched to the source adapter associated with the data source.

3.7.3 Version Resources

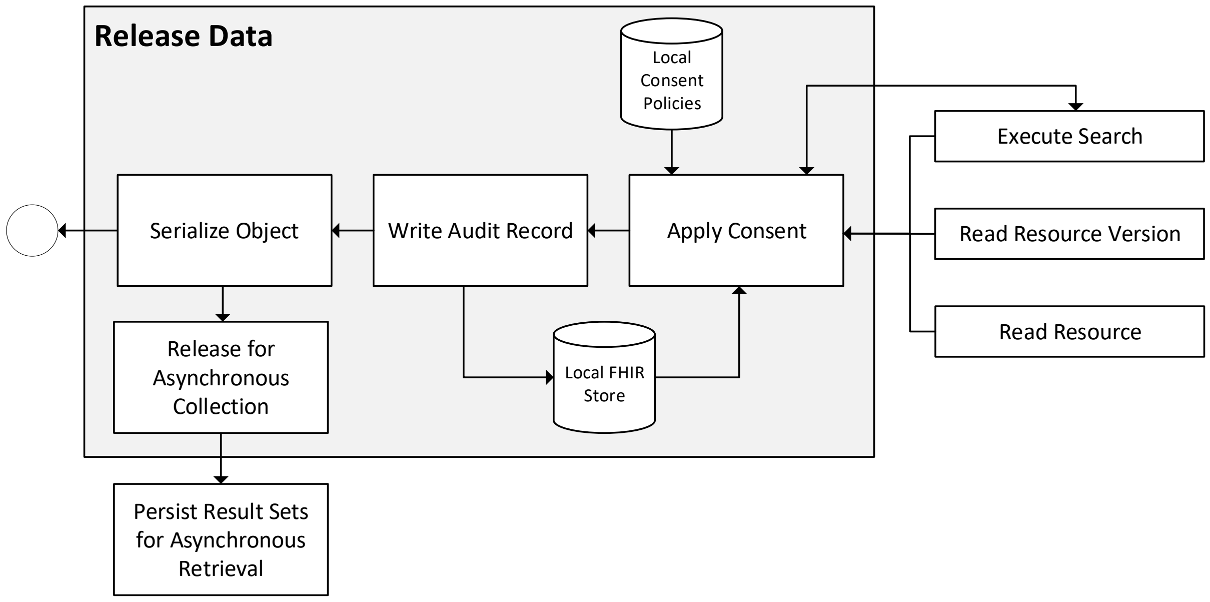
This module was introduced in section 3.3.6 and performs identically in the read pathway as does in the search pathway, in this instance operating on a single resource.

3.7.4 Trim Resource Properties

This module was introduced in section 3.3.7 and performs identically in the read pathway as does in the search pathway, in this instance applying the _summary directive to a single resource.

3.8 Release Data

This pathway is used by all resource read and search pathways to ensure common controls are applied to data released by the FHIR Proxy. Data is either released synchronously over the socket connection which initiated the request or persisted locally for asynchronous collection.



3.8.1 Apply Consent

Requirements for consent modelling are described by design paper 008 – "Consent Management" and the focus here is on the implementation of the principles set out section 2.7 which, under certain circumstances, require local enforcement of consent rules. Local consent enforcement is not needed if there are no local policies in force (over and above those enforced regionally) and the FHIR Proxy is configured to only accept requests from the regional FHIR aggregator (i.e.: there are no direct relationships with trusted peer data consumer which bypass the mediation of regional infrastructure).

This model assumes that citizen consent is recorded as FHIR *Consent* resources. Resources are persisted in the Local FHIR Store. *Consent* resources reference consent policies. A policy is defined in terms of:

- the context in which the policy applies (why data is being accessed and by whom);
- the data points which the policy covers (search terms which can be applied to resources to determine whether they are within the scope of a policy).

The role of this component is to determine whether any of the local consent policies apply given the context in which the resource is being accessed (context is defined by the data consumer's role, reason for access, and organisation from which access is requested: these data items being embedded in the JWT used for authorisation). If a policy applies and the patient that is the subject of data being released has opted in or out of the policy, then the data must be tested for conformance with the policy before being released.

Some resources are not patient identifiable, and consent does not apply to these. Patient identifiable resources can be associated with a subject patient and it these on which the processing described here relates. Design paper 005 – "Identity and Access Management" distinguishes the different categories of resources).

The following processing relies on consent rules being defined which cause resources to be explicitly included or excluded from a result set based on a search path. For instance:

```
Exclude /Obsevation?context.type=MentalHealth
```

might be used to exclude all observations captured in encounters classified as relating to mental health from a result set. Any resource can be tested to determine compliance with the rule by executing a search for the resource id including the rules search term i.e.:

```
/Obsevation?context.type=MentalHealth&_id=1234
```

i. For each patient identifiable resource being released determine the policies to which the patient has an opt-in or opt-out
ii. From the policy definition determine the context in which the policy applies.
iii. If the context matches properties supplied in the JWT used in the request, then:
iv. Use the id of the resource to construct search strings from the policy definition.
v. Execute the search pathway for each search in turn and use the results to determine whether the resource should be withheld.
vi. If so, then remove the resource from the response.

Consent policies should have an order of precedence which allows one policy to take priority over another.

3.8.2 Write Audit Record

Auditing is discussed in design paper 009 – "Auditing" and the principles set out in section 2.6.

i. Create an *AuditEvent* resource in the local FHIR store for each resource released.

3.8.3 Serialize Object

The REST request stipulates the format in which data is to be returned. Supported options are XML or JSON. This module serialises an internal object graph structure into the required representation.

3.8.4 Release for Asynchronous Collection

The option to collect asynchronously was established by the invoker of the service and a unique identifier was assigned to the request. Search results arrive in pages and each page is persisted and indexed by the request identifier.

3.9 Persist Result Set for Asyncronous Retrieval

Remove redundant fields from the search bundle and persists the paginated result set. Each page is assigned a unique identifier and can be retrieved independently.

3.10 Process Asynchronous Request

Handles the two categories of request which are supported by the asynchronous processing endpoint:

1. Request status update.
2. Retrieve result page.

Status update requests quote a unique request identifier and if validly formed the possible responses are an HTTP 202 response code for requests which are in progress or an HTTP 200 response code for requests where the search is complete, and results may be collected.

If issuing an HTTP 200 response, then the HTTP is a structure which includes an array of URLs from which individual search pages can be retrieved.

Result page retrieval requests are made against one of the advertised URLs. The response is a bundle of resources retrieved from the original search request. Once retrieved the persisted bundle is marked as such and will be purged by a batch process (not shown in the processing model). An attempt to retrieve a purged bundle will result in a 404 HTTP response.

3.11 Create, Update and Patch Resource

All manageable data is held in the local FHIR Store. This pathway either creates or updates an object in this database. Objects are either updated in their entirety or by applying a sequence of operations from a patch document.

An *AuditResource* is created for each transaction.