GP Connect - FHIR Implementation Guide

Draft Published: 17th May 2016

> Note: this document is a draft implementation guide and does not embody the finalised requirements for the GP Connect First of Type (FoT) FHIR APIs.

Notational Conventions

The keywords "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in RFC 2119.

FHIR Implementation Guidance

The purpose of the implementation guide is to describe adaptations of the base FHIR specification specific for GP Connect First of Type (FoT) implementations. It therefore focuses mainly on any additional constraints and specialisations from the base specification that apply to GP Connect. The complete specification therefore comprises the base FHIR specification plus the constraints and specialisations described herein. Where there is a conflict between the base specification and this document, this document shall take precedence.

FHIR Overview

As outlined on the official HL7 FHIR website:

> FHIR (Fast Health Interoperability Resources) is designed to enable information exchange to support the provision of healthcare in a wide variety of settings. The specification builds on and adapts modern, widely used RESTful practices to enable the provision of integrated healthcare across a wide range of teams and organizations.

FHIR Timelines

> Note: this document is written to accompany the currently released version of FHIR which is **Draft Standard For Trial Use (DSTU) 2, published on the 23 September 2015**.

FHIR is currently a draft standard and as such is expected to evolve and develop over time. As such the HSCIC expects FHIR clients and servers (developed as part of GP Connect) to be maintained and uplifted to newer versions of the FHIR standard as they become available and are staged into the GP Connect programme.

> Note: a pre-release version of FHIR (expected to be the basis of STU3) has been published as the May 2016 release. The FHIR community is expecting the finalised DSTU3 release to be made available during September 2016.
>
> Note: DSTU3 includes the introduction of a new draft `Task` resource which may impact the future implementation direction of the GP Connect Task profile.

FHIR Implementations

The Health Level Seven (HL7) International standards body maintains a list of open source FHIR implementations on their Wiki. Currently five FHIR server implementations and a number of client libraries are available as open source software. These are written in a variety of popular programming languages (e.g. Java, C#.NET, JavaScript and Python).

> Note: the HSCIC recommends software vendors consider utilising (and contributing back to) an existing open source FHIR library to both help accelerate development and to grow/mature the open source FHIR ecosystem.

FHIR In Scope

To help GP Connect FoT implementers deal with the FHIR learning curve the HSCIC has worked to constrain the scope of the FHIR standard that is expected to be implemented in the first tranche of development work as follows:

1. Resource
    1. Resource Identity
    2. Business Identifiers
2. Domain Resource

   1. Common API
      1. Patient profiled as GPConnect-Patient-1-0.
      2. Organization profiled as GPConnect-Organization-1-0.
         1. Location profiled as GPConnect-Location-1-0
   2. Care Record API
      1. Please refer to the Resource Type section of this document.
   3. Bookings API
      1. Schedule profiled as GPConnect-Schedule-1-0
      2. Slot profiled as GPConnect-Slot-1-0
      3. Appointment profiled as GPConnect-Appointment-1-0
   4. Task API
      1. Order profiled as GPConnect-Order-1-0
3. Resource Metadata
   1. profile
   2. version Id
4. Bundle
5. Data Types
   1. Primitive Types
      1. All primitive types SHALL be supported.
   2. Complex Types
      1. The following complex types SHALL be supported.
         1. Identifier
         2. Coding
         3. Codeable Concept
         4. Period
      2. It is expected that further complex types will be introduced as required to model structured data.
6. Interactions
   1. Instance
      1. READ
      2. UPDATE
      3. DELETE
   2. Type Level
      1. CREATE
      2. SEARCH
7. Search
   1. Search Parameter Types
      1. Date
      2. Token
      3. Reference
   2. Parameters for all resources
      1. _query
   3. Search result parameters
      1. _include can be used internally inside a named `_query` operation.
      2. _sort can be used internally inside a named `_query` operation.
   4. Chained parameters (limited to patient identifier only)
8. Operations
   1. Implementation Defined Operations

> Note: it is fully expected that over time new areas of the FHIR standard will be brought into scope as new capabilities are requested and vendors understanding of FHIR standard matures. This is in-line with the agreed iterative/Agile engagement process that has been agreed between the HSCIC and the principle GP system vendors.

FHIR Out Of Scope

GP Connect FoT systems are not expected to implement the following aspects of the FHIR standard under the scope of the first tranche of development work:

1. Operations
   1. Validate a resource

   2. Meta operations
   3. Generate a document
   4. Process message
   5. Find a functional list
   6. ConceptMap operations
   7. Closure Table Maintenance
   8. Fetch records
   9. Questionnaire operations
   10. Terminology operations
2. Interactions
   1. Instance
      1. VREAD
      2. HISTORY
   2. Type Level
      1. HISTORY
   3. Whole System
      1. BATCH TRANSACTION
      2. HISTORY
      3. SEACH
   4. Conditional Interactions
      1. CREATE
      2. UPDATE
      3. DELETE
3. Comparators for Search
   1. equal in range
   2. lt (less than)
   3. le (less or equal to)
   4. gt (great than)
   5. ge (greater or equal to)
   6. ne (not equal to)
4. Search
   1. Parameter Types
      1. Number
      2. String
      3. Quantity
      4. Composite
      5. URL
   2. Parameters for all resources
      1. _language
      2. _lastUpdate
      3. _tag
      4. _profile
      5. _security
      6. _text
      7. _content
      8. _list
      9. _filter
   3. Search result parameters
      1. _revinclude
      2. _count
      3. _summary
      4. _elements
      5. _contained
      6. _containedType
   4. Chained parameters (limited to patient identifier only)
5. Resource Metadata
   1. lastUpdated
   2. security

3. tag

> Note: vendors are free to implement additional FHIR functionality above that mandated under the GP Connect FoT delivery if they so desire.

TODO: do we need to put some words in here about non CA'd interfaces being provided at the risk of the individual vendors?

Internet Standards

Clients and servers SHALL be conformant to the following Internet Engineering Task Force (IETF) Request for Comments (RFCs) which are the principle technical standards that underpin the design and development of the internet and thus FHIR's APIs.

- Transport level integration SHALL be via HTTP as defined in RFC 2616.
- Transport level security SHALL be via TLS/HTTPS as defined in RFC 5246 and RFC 6176.
- HTTP Strict Transport Security (HSTS) as defined in RFC 6797 SHALL be employed to protect against protocol downgrade attacks and cookie hijacking.

> Note: the HSCIC is currently evaluating how Cross-Origin Resource Sharing (CORS) will be handled for web and mobile based applications.

Endpoint Resolution

Clients SHALL perform a sequence of query operations against existing Spine services to enable FHIR endpoint resolution.

1. Clients SHALL perform (or have previously performed) a PDS lookup for a patient.
   1. Using the PDS results the client SHALL determine the patient's primary GP organisation.
2. Clients SHALL perform (or have previously performed) a SDS lookup using the ODS code of the patient's primary GP organisation.
   1. Using the SDS results the client SHALL determine the Principle GP system responsible for hosting the most up to date GP care record.
      1. EMIS Health
      2. INPS
      3. Micotest
      4. TTP
3. Clients SHALL construct a FHIR Service Root URL suitable for access to a GP vendor's FHIR server.

TODO: check if this is correct or if we will be resolving the endpoint from Spine using some other mechanism.

Security

TLS SHALL be used for all data exchange. The TLS communications are established prior to any HTTP command/response, so the whole FHIR interaction is protected by the TLS communications.

The security of the endpoints of the TLS communications must be risk-managed, so as to prevent inappropriate risks (e.g. audit logging of the GET parameters into an unprotected audit log).

Authentication

FHIR standard specifies that users/clients/servers may be authenticated in any way desired. However, for web-centric use, oAuth (RFC 6749) is recommended but not mandated by the FHIR standard.

For the purpose of GP Connect FoT clients and servers SHALL authenticate using TLS Mutual Authentication (MA) utilising client certificates provided by the HSCIC for this purpose.

TODO: add details of which (if any) PKI client certificate details will need to be included into custom HTTP headers.

Conformance

Servers SHALL provide a FHIR conformance resource that identifies all of the profiles that the server supports for each resource type.

A servers conformance statement SHALL be available using the following conformance interactions:

```
GET [base]/metadata {?_format=[mime-type]}
OPTIONS [base] {?_format=[mime-type]}
```

> Note: HSCIC will be publishing a FHIR conformance profile of type `kind=requirements` which will outline the conformance requirements for GP Connect FoT systems. It is envisaged that this conformance profile template will then be used by GP software vendors to more easily derive

> their own conformance profiles of `kind=instance` for their own systems.

Please refer to Appendix 1. GP Connect FoT Conformance Profile for the conformance profile template.

> Note: HSCIC are evaluating the benefits of providing a centrally hosted FHIR server to act as a profile and conformance repository. This is however out of scope for the GP Connect FoT deployments.

GP Connect API Conformance

GP Connect comprises a number of RESTful API bundles. Each API bundle is intended to support sets of related use cases, for example the Appointment API bundle supports viewing, booking and cancelling GP appointments in a number of scenarios.

Individual API bundles may be provided independently of each other. GP Connect conformance may be claimed in relation to one or more API bundles. A provider claiming to provide an API bundle must be fully conformant (i.e. implement all of the resource profiles and interactions for the API bundle as specified in this document and all of the general requirements described herein).

TODO: include a link to the conformance assessment / testing / go-live process.

RESTful API

The RESTful API described in the FHIR standard is built on top of the Hypertext Transfer Protocol (HTTP) with the same HTTP verbs (`GET`, `POST`, `PUT`, `DELETE`, etc.) commonly used by web browsers. Furthermore, FHIR exposes resources (and operations) as Uniform Resource Identifiers (URIs). For example, a `Patient` resource `/fhir/Patient/1`, can be operated upon using standard HTTP verbs such as `DELETE /fhir/Patient/1` to remove the patient record.

> Note: the FHIR RESTful API Style Guide defines the following URL conventions which are used throughout the remainder of this document:
>
> - URL pattern content surrounded by **[ ]** are mandatory.
> - URL pattern content surrounded by **{ }** are optional.

Service Root URL

The Service Root URL is the address where all of the resources defined by this interface are found. The Service Root URL takes the form of:

```
https://server{/path}
```

The Service Root URL is the `[base]` portion of all FHIR APIs.

> Note: all URLs (and ids that form part of the URL) defined by this specification are case sensitive.

> Note: discussions are ongoing around the use of Semantic Versioning in the server's Service Root URL to provide a clear distinction between API versions that are incompatible (i.e. contain breaking changes) vs. backwards-compatible (i.e. contain no breaking changes).

Resource URL

```
VERB [base]/[type]/[id] {?_format=[mime-type]}
```

Clients and servers constructing URLs SHALL conform to RFC 3986 Section 6 Appendix A which requires percent-encoding for a number of characters that occasionally appear in the URLs (mainly in search parameters).

HTTP Verbs

The following HTTP verbs SHALL be supported to allow RESTful API interactions with the various FHIR resources:

- **GET**
- **POST**
- **PUT**
- **DELETE**
- **OPTION** (which is used to retrieve the servers conformance statement)

> Note: please see later sections for which HTTP verbs are expected to be available for which FHIR resources.

> Note: in a future version of the FHIR standard it is expected that the **PATCH** verb will also be supported.

Resource Type

Clients and servers SHALL support the following FHIR resource types as profiled for use in the GP Connect FoT systems:

- Identification:
  - Patient
  - Practitioner
  - Organization
  - Location
- Clinical:
  - AllergyIntolerance*
  - Condition (Problem & Diagnosis)
  - Procedure*
  - Referral*
  - Medication*
  - MedicationOrder*
  - MedicationAdministration*
  - MedicationDispense*
  - Immunisation*
  - Observation*
- Workflow:
  - Appointment
  - Order (Task)
  - Encounter*
- Conformance:
  - Conformance
  - OperationDefinition
  - SearchParameter

> Note: The list of resources above may not be complete and resources marked with a * are still undergoing active review as part of the work taking place inside GP Connect to reach agreement on the structure of the GP care record.

> Note: The canonical list of resources (and the associated FHIR profiles) are/will be made available on the NHS Health Developer Network website under the GP Soc IM2 landing page.

TODO: write some words around providing fine-grained RESTful APIs for these resources.

Resource ID

This is the `logical Id` of the resource which is assigned by the server responsible for storing it. The logical identity is unique within the space of all resources of the same type on the same server, is case sensitive and can be up to 64 characters long.

Once assigned, the identity SHALL never change. `logical Ids` are always opaque, and external systems need not and should not attempt to determine their internal structure.

> Note: as stated above and in the FHIR standard, `logical Ids` are opaque and other systems should not attempt to determine their structure (or rely on this structure for performing interactions). Furthermore, as they are assigned by each server responsible for storing a resource they are usually implementation specific. For, example:
>
> - NoSQL document stores typically preferring a GUID key (e.g. 0b28be67-dfce-4bb3-a6df-0d0c7b5ab4).
> - Relational database stores typically preferring a integer key (e.g. 2345).

Content Types

Servers SHALL support both formal MIME-types for FHIR resources:

- XML: `application/xml+fhir`
- JSON: `application/json+fhir`

Servers SHALL support the optional `_format` parameter in order to allow the client to specify the response format by its MIME-type. If both are present, the `_format` parameter overrides the `Accept` header value in the request.

Transfer Encoding

Clients and servers SHALL support the HTTP `Transfer-Encoding` header with a value of `chunked`. This indicates that the body of a HTTP response will returned as an unspecified number of data chunks (without an explicit `Content-Length` header).

Content Compression

To improve system performances clients/servers SHALL support GZIP compression.

Compression is requested by setting the `Accept-Encoding` header to `gzip`.

> Note: The HSCIC recognises that applying content compression is key to reducing bandwidth needs and improving battery life for mobile devices.

HTTP Headers

Proxying Headers

Additional HTTP headers SHALL be added into the HTTP request/response for the purpose of allowing the proxy system to disclose information lost in the proxying process (e.g. the originating IP address of a request). Typically, this information is added to proxy forwarding headers as defined in RFC 7239.

Provenance & Audit Headers

In order to meet auditing and provenance requirements (which are expected to be closely aligned with the IM1 requirements), clients SHALL provide an oAuth 2.0 Bearer token in the HTTP Authentication header (as outlined in RFC 6749) in the form of a JSON Web Token (JWT) as defined in RFC 7519.

The following fields SHALL be populated:

- Issuer (iss) – identifies principle that issued the JWT.
- Subject (sub) – identifies the subject of the JWT.
- Audience (aud) – identifies the recipients that the JWT is intended for.
- Issued at (iat) – identifies the time at which the JWT was issued.
- Expiration time (exp) – identifies the time on or after which the JWT MUST NOT be accepted for processing.

> Note: The HSCIC will also be defining a small number of additional headers which are also required to be included in an HSCIC defined custom header.

TODO: add some words that outline what would typically go into these extra headers above and beyond what is expected to be in the JWT.
TODO: add an example of a valid JWT token
TODO: outline that in the GP Connect FoT scope this token will not be centrally issued.

Managing Return Content

Severs SHALL default to the `return=representation` behaviour (i.e. returning the entire resource) for interactions that create or update resources.

Servers SHOULD honour a `return=minimal` or `return=representation` preference indicated in the `Prefer` request header, if present.

Managing Resource Contention

Servers SHALL always return an `ETag` header with each resource including the resources `vesionId`:

```
HTTP 200 OK
Date: Sat, 09 Feb 2013 16:09:50 GMT
Last-Modified: Sat, 02 Feb 2013 12:02:47 GMT
ETag: W/"23"
Content-Type: application/json+fhir
```

`ETag` headers which denote resource `version Id`s SHALL be prefixed with `W/` and enclosed in quotes, for example:

```
ETag: W/"3141"
```

Clients SHALL submit update requests with an `If-Match` header that quotes the `ETag` from the server.

```
PUT /Patient/347 HTTP/1.1
If-Match: W/"23"
```

If the `version Id` given in the `If-Match` header does not match, the server returns a `409` **Conflict** status code instead of updating the resource.

Managing Return Errors

FHIR defines an OperationOutcome resource that can be used to convey specific detailed processable error information. An OperationOutcome may be returned with any HTTP `4xx` or `5xx` response, but is not always required.

FHIR Resources

Resource References

The FHIR resource model includes references from one resource to another.

A reference can be either:

- a relative or absolute URL to a resource managed by the same resource server (a local reference), or
- an absolute URL to a resource managed by another resource server (a remote reference)

A provider’s ability to process a request relating to a resource may depend on its ability to use one or more resource references that the resource contains (i.e. its ability to ‘follow the links’ to other resources).

> Note: GP Connect FoT clients and servers SHALL utilise local relative references only and as such the resources will be expected to reside on the same server.

Resource references SHOULD include a short human-readable `display` field for identification of the resource that is being referenced which can be used for display purposes without needing to pull the entire referenced resource.

TODO: do we want to make providing the display property mandatory and provide guidance around what it should contain? as having this property as a meaningful human readable string would allow clients to not have to pull separately/include in search results the full linked resource.

Resource Metadata

Servers SHALL provide the `profile` metadata for each item, asserting that the content conforms to one of the GP Connect FoT resource profiles.

Servers SHALL provide the `version Id` metadata for each item. This SHALL change each time the content of the resource changes.

Clients SHALL utilise the `version Id` when performing updates to allow management of resource contention and to protect against Lost Updates.

Resource Transactions

When performing an update or create interaction, servers:

- SHALL **validate the content against valid profiles** and business rules before creating/updating the resource.
- MAY apply business rules that alter the content.
- MAY merge updated content with existing content.

Servers SHALL validate the existence of any referenced resources when creating or updating a resource. For example, a `Slot` reference (e.g. `Slot/D497DB00-99AA-11E5-A837-0800200C9A66`) used when creating a new `Appointment` would be checked for existence on the server and an error returned (and the create interaction aborted) if the slot does not exist.

TODO: provide more guidance on payload validation and error behaviours

TODO: publish the FHIR profiles

Servers SHALL provide a read interaction for every resource it accepts update interactions on.

Consumers SHALL follow the pattern described in the Transactional Integrity section of the base FHIR specification, built on top of version-aware updates, for updating resources.

FHIR Resource Interactions

Requests

Servers SHALL be able to consume and process the following requests for GP Connect FoT.

Interaction	Path	Request Verb	Request Content-Type	Body	Prefer	Conditional
`read`	`/[type]/[id]`	`GET`	N/A	N/A	N/A	`ETag`
`update`	`/[type]/[id]`	`PUT`	R	Resource	O	`If-Match`
`delete`	`/[type]/[id]`	`DELETE`	N/A	N/A	N/A	N/A
`create`	`/[type]`	`POST`	R	Resource	O	N/A
`search`	`/[type]?`	`GET`	N/A	N/A	N/A	N/A

`(operation)`	`/[type]/$[name]`
	`/[type]/[id]/$[name]`
		`POST`
		`GET`
			R
			`application/x-www-form-urlencoded`
				Parameters form data	N/A	N/A

> N/A = not present, R = Required, O = Optional

Responses

Servers SHALL be expected to produce the following responses for GP Connect FoT.

Interaction	Response Content-Type	Body	Location	Content-Location	Versioning	Status Codes
`read`	R	R: Resource	N/A	R	`ETag`	`200`, `404`, `410`

`update`	R	R: Resource	N/A	R	`ETag`	`200`, `201`, `400`, `404`
						`405`, `409`, `412`, `422`

`delete`	R	R: Operation Outcome	N/A	N/A	N/A	`200`, `204`, `404`, `405`, `409`, `412`
`create`	R	R: Resource	R	R	`ETag`	`201`, `400`, `404` `405`, `422`
`search`	R	R: Bundle	N/A	N/A	N/A	`200`, `403`
`(operation)`	R	R: Parameters/Resource	N/A	N/A	N/A	`200`

> N/A = not present, R = Required, O = Optional

Response Codes

Servers SHALL produce the following main HTTP Status Codes.

HTTP Status Code	Description
`200`	OK
`201`	Created
`400`	Bad Request

`403`	Forbidden
`404`	Not Found
`405`	Method Not Allowed
`409`	Conflict
`410`	Gone
`412`	Precondition Failed
`422`	Unprocessable Entity
`500`	Internal Server Error

Rejecting Updates

OperationOutcome may be returned with any HTTP `4xx` or `5xx` response, but is not required - many of these errors may be generated by generic server frameworks underlying a FHIR server.

Servers are permitted to reject update interactions because of integrity concerns or other business rules, and return HTTP status codes accordingly (usually a `422`).

As outlined in the FHIR specification, any of these errors SHOULD be accompanied by an OperationOutcome resource that provides additional detail concerning the issue.

TODO: HSCIC to provide a concrete list of error codes and severity levels.

HTTP Status Code	HSCIC Code	Description
`422`	GPC-INVALID-RESOURCE	Submitted resource is not valid.
`422`	GPC-INVALID-PARAMETER	Submitted parameters are not valid.

Compartment Based Access

```
VERB [base]/[compartment_type]/[id]/[type]{?_format=[mime-type]}
```

Each resource type may belong to one or more logical compartments. A compartment is a logical grouping of resources which share a common property.

Servers SHALL support the `Patient` compartment for `Appointment` access.

The patient compartment includes any resources where the `subject` of the resource is the patient.

For example, to retrieve a list of all of a patient's appointments, use the URL:

```
GET [base]/Patient/[id]/Appointment
```

Read Resource

```
GET [base]/[type]/[id]{?_format=[mime-type]}
```

The returned resource SHALL have an `id` element with a value that is the [id].

Servers SHALL return an `ETag` header with the `version Id` of the resource.

Available for Resources

Bundle	Resource(s)

Common	`Patient`, `Practitioner`, `Organization`
Care Record	`AllergyIntolerance`, `Condition`, `Immunisation`, `Medication`, `MedicationOrder`, `MedicationDispense`, `MedicationAdministraion`, `Observation`, `Procedure`, `Referral`
Appointments	`Appointment`, `Schedule`, `Slot`, `Location`
Tasks	`Order`

> Note: In workshop discussions with all principle GP system vendors it has been agreed that record locking (inside the GP system) will not impact on the ability of clients to query the GP Connect APIs and obtain the latest saved/committed clinical and administrative data.

Retrieving a patient resource by logical Id

Request

```
GET [base]/Patient/[id]
```

For example:

```
GET [base]/Patient/1A6E1B1C-6340-4663-926C-9CD1306EAAF8?_format=application/xml+fhir
```

> Note: in this example the response format has been specified in the request URL using the `_format` parameter. This could also have been specified using the HTTP Accept Header mechanism.

Response

```
<Patient xmlns="http://hl7.org/fhir">
    <id value="1A6E1B1C-6340-4663-926C-9CD1306EAAF8" />
    <meta>
        <profile value="http://fhir.nhs.net/StructureDefinition/gpconnect-patient-1-0" />
    </meta>
    <identifier>
        <system value="http://fhir.nhs.net/Id/nhs-number" />
        <value value="9900002831" />
    </identifier>
    <identifier>
        <type>
            <coding>
                <system value="http://fhir.nhs.net/ValueSet/gpconnect-patient-identifier-type-1-0" />
                <code value="Local" />
                <display value="Local identifier" />
            </coding>
        </type>
        <system value="http://fhir.nhs.net/Id/local-identifier" />
        <value value="L12345" />
    </identifier>
    <name>
        <use value="usual" />
        <family value="Taylor" />
        <given value="Sally" />
        <prefix value="Mrs" />
    </name>
    <birthDate value="1947-06-09" />
    <address>
        <use value="home" />
        <type value="both" />
        <line value="42, Grove Street" />
        <city value="Overtown" />
        <district value="West Yorkshire" />
        <postalCode value="LS21 1PF" />
    </address>
</Patient>
```

Create Resource

To create a new resource a RESTful **POST** operation with a request body SHALL be utilised.

```
POST [base]/[resourcetype]
```

When the server creates a new resource and returns a `201` **Created** HTTP status code, it SHALL also return a `Location` header which contains the new `logical Id` and `version Id` of the created resource.

```
Location: [base]/[type]/[id]/_history/[vid]
```

[id] and [vid] are the newly created `logical Id` and `version Id` for the resource version. An `ETag` header with the `version Id` and a `Last-Modified` header will also be included in the response.

> Note: GP Connect FoT servers are not required to implement the VREAD operation which would be required to retrieve a given version of a resource using the `Location` header details.

Available for Resources

Bundle	Resource(s)
Common	
Care Record	
Appointments	`Appointment`
Tasks	`Order`

> Note: In workshop discussions with all principle GP system vendors it has been agreed that record locking (inside the GP CIS) will not impact on the ability of clients to query the GP Connect APIs and obtain the latest saved patient data.

Create Example: Book an appointment for a patient

Request

```
POST [base]/Appointment
```

Request Body

```
<Appointment>
</Appointment>
```

Response

```
<Appointment>
</Appointment>
```

Update Resource

To update an existing resource, a RESTful **PUT** operation with a request body SHALL be utilised.

```
PUT [base]/[resourcetype]/[id]
```

Bundle	Resource(s)	Field(s)
Common		

Care Record		
Appointments	`Appointment`	reason
Tasks		

Update Example: Modify the appointment reason for a patient as their condition has changed

Request

```
PUT [base]/Appointment/2A6FFB1C-5540-1613-926C-7DD1306EBBF8
```

Request Body

```
<Appointment>
</Appointment>
```

TODO: we need to decide if we should remove the $cancel operation in favour of a more RESTful PUT which changes the appointment `status` and adds a `cancellationReason`.

Response

```
<Appointment>
</Appointment>
```

> Note: The Server SHALL return an error and abort updating the appointment resource if any details other than the `reason` field are modified.

Delete Resource

To delete an existing resource, a RESTful **DELETE** operation with no request body SHALL be utilised.

```
DELETE [base]/[resourcetype]/[id]
```

Bundle	Resource(s)
Common	
Care Record	
Appointments	
Tasks	

> Note: GP Connect FoT clients and servers are currently not expected to utilise the ability to delete a resource using the RESTful DELETE operation. However, this section is included as implementers should ensure that their implementation choices don't preclude the use of this HTTP verb in future release.

Operations

Operations are used (a) where the server needs to play an active role in formulating the content of the response, not merely return existing information, or (b) where the intended purpose is to cause side effects such as the modification of existing resources, or creation of new resources.

Operation Example: ?

TODO: give an example of an operation.

Request

```
POST [base]/Patient/1A6E1B1C-6340-4663-926C-9CD1306EAAF8/$getcarerecord
```

Request Body

```
<Operation>
</Operation>
```

Response

```
<OperationOutcome>
</OperationOutcome>
```

> Note: as outlined in the Extend and Restricting the API section of the FHIR standard, the HSCIC is considering prefixing it's operation names with a short prefix followed by a "." to reduce the likelihood of name conflicts.

Search Resources

A simple search is executed by performing a `GET` request optionally accompanied by zero or more name-value URL encoded parameters:

```
GET [base]/[resourcetype]?name=value&...
```

In order to enable searching by date/time range, servers SHALL support the following prefixes as defined in the base FHIR specification for date parameters: eq, gt, lt, ge, le.

TODO: check if we need to support the search prefixes if we're just using dates

Chained Parameters

Servers SHALL support searching by a chained `Patient` identifier parameter for references to `Patient` resources that conform to the `GP-Patient` profile (and therefore have an NHS Number identifier). For example:

```
[base]/AllergyIntolerance?patient.identifier=http://fhir.nhs.net/Id/nhs-number|1234569876
```

> Note: GP Connect FoT clients and servers are not expected to support arbitrary adhoc searching.

Search Example: Search for a patient resource by business Id

Request

```
GET [base]/Patient?identifier=http://fhir.nhs.net/Id/nhs-number|9900002831
```

If a Patient resource for NHS number 9900002831 exists then the server SHALL return a Bundle containing all Patient resources with the specified NHS number identifier.

Response

```
<Bundle>
    <Patient xmlns="http://hl7.org/fhir">
        <id value="1A6E1B1C-6340-4663-926C-9CD1306EAAF8" />
        <meta>
            <profile value="http://fhir.nhs.net/StructureDefinition/gpconnect-patient-1-0" />
        </meta>
        <identifier>
            <system value="http://fhir.nhs.net/Id/nhs-number" />
            <value value="9900002831" />
        </identifier>
        <name>
            <use value="usual" />
            <family value="Smith" />
            <given value="Mike" />
            <prefix value="Mr" />
        </name>
        <birthDate value="1977-01-09" />
        <address>
            <use value="home" />
            <type value="both" />
            <line value="2, The Green" />
            <city value="Harrogate" />
            <district value="Yorkshire" />
            <postalCode value="HG1 4AF" />
        </address>
    </Patient>
</Bundle>
```

Advanced Search

Servers SHALL implement the `_query` search parameter to enable custom named search profiles to be defined and used which describe a specific query operation.

```
GET [base]/[resourcetype]?_query=[query_name]&name=value&...
```

Servers SHOULD implement the standard search equivalent of the advanced custom named search for queries defined under the GP Connect FoT.

Request

```
GET [base]/Schedule?_query=getscheduledetails&date>=2016-05-12&date<=2016-05-26
```

Response

```
<Bundle xmlns="http://hl7.org/fhir">
     <type value="searchset"/>
</Bundle>
```

TODO: run a named query on a public FHIR server and grab the searchset as an example.

GP Connect APIs

Common Interactions

Find A Patient

Resolve (zero or more) `Patient` resources using an NHS number.

```
GET /Patient?identifier=http://fhir.nhs.net/Id/nhs-number|9900002831
```

TODO: determine if we need to align with the Spine Mini Service and also get clients to send the patient's DOB.

Find A Practitioner

Resolve (zero or more) `Practitioner` resources using an SDS User ID.

Request

```
GET /Practitioner?identifier=http://fhir.nhs.net/Id/sds-user-id|55443322
```

Response

Find An Organisation

Resolve (zero or more) `Organisation` resources using ODS code.

Request

```
GET /Organization?identifier=http://fhir.nhs.net/Id/ods-organization-code|Y123456789
```

Response

Find A Location

Resolve (zero or more) `Location` resources using ODS code OR ODS site code.

Request

```
GET /Location?identifier=http://fhir.nhs.net/Id/ods-organization-code|Y123456789
```

OR

```
GET /Location?identifier=http://fhir.nhs.net/Id/ods-site-code|789
```

Response

TODO: determine if it really makes any sense to search by site code only or whether we should mandate that all Locations are traceable back to a top-level ODS code.

TODO: determine if managingOrganization should really be mandatory; shouldn't all locations have an organisation responsible for the provisioning and upkeep?

> Note: all the above search operations using business identifiers can potentially return multiple resources.

As outlined in the FHIR standard, clients SHALL handle servers returning a bundle of requested resources, instead of just the resource itself, as shown below:

```
<Bundle>
  ...
  <entry>
    <fullUrl value="/Patient/1A6E1B1C-6340-4663-926C-9CD1306EAAF8"/>
    <resource>
      <Patient>
        <id value="1A6E1B1C-6340-4663-926C-9CD1306EAAF8">
        ...
      </Patient>
    </resource>
    <search>
      <mode value="match"/>
    </search>
  </entry>
</Bundle>
```

TODO: check the above output against a publicly available FHIR server.

Care Record Interactions

Retrieve a HTML care record for a patient

The typical flow to retrieve a patient's HTML care record is:

1. Search by `NHS number` for, or otherwise obtain, a `Patient` resource.
2. Determine which record sections are to be retrieved (see table below).
3. Retrieve the care record as HTML using the `$getcarerecord` operation.

Section Code	Section Name
SUM	Summary

ENC	Encounters
CIT	Clinical Items
AIT	Administrative Items
REF	Referrals
IMM	Immunisations
PRB	Problems
MED	Medications
OBS	Observations
INV	Investigations
ALL	Allergies and Sensitivities

TODO: need to agree on the codes as these are ones I've just made up and won't align with say segment codes in HL7v2 or in fact the original GPConnect-Record-Section-1-0 headings.

> Note: In workshop discussions with all principle GP system vendors it has been agreed that within the scope of GP Connect FoT minimal structured `Patient`, `Practitioner` and `Organization` resources will also be included in the searchset bundle.

Request

```
POST [base]/Patient/$getcarerecord
```

Request Body

```
<Operation>
</Operation>
```

Input Parameters

Parameter		Validation
patientNHSNumber	R	Has the correct format and check-digit to be a valid NHS number.
recordSection	O	Is a list of valid record sections.
timePeriod	O	`start` is equal to or lower (earlier) than `end`, if both are present.

> R = Required, O = Optional

TODO: decide if we should also provide this operation at the instance level.
TODO: decide how we specify give me everything or if this should even be allowed.
TODO: decide if unknown section names should cause an error.
TODO: decide if we're going to support multiple timePeriods, one per recordSection.
TODO: decide on what the behaviour is if the optional fields are not supplied (i.e. default timePeriods

Response

Output Parameters

Parameter	Description
return	A `Bundle` of type `searchset`.

TODO: add a better description of what is returned in the resulting bundle.

```
<Bundle>
</Bundle>
```

TODO: add an example fragment of a searchset bundle.

Errors

Servers SHALL return errors in retrieving the GP care record as a `OperationOutcome` with `severity=error`.

```
<OperationOutcome xmlns="http://hl7.org/fhir">
    <id value="A3B55ECB-9B98-453C-B463-C6001ECE3C36" />
    <meta>
        <profile value="http://fhir.nhs.net/StructureDefinition/gpconnect-operationoutcome-1-0" />
    </meta>
    <issue>
        <severity value="error" />
        <code value="invalid" />
        <details>
            <coding>
                <system value="http://fhir.nhs.net/ValueSet/gpconnect-response-code-1-0" />
                <code value="GPC-0001" />
            </coding>
        </details>
        <diagnostics value="NHS Number Invalid" />
    </issue>
</OperationOutcome>
```

HTTP Status Code	HSCIC Code	Description
TODO	GCR-0001	NHS number invalid
TODO	GCR-0002	No record found

TODO: need to harmonize these error codes defined in the gpconnect-response-code-1-0 with those in other parts of this and other documents.

Retrieve a structured care record for a patient

The typical flow to retrieve a patient's structured care record is:

1. Search by `NHS number` for, or otherwise obtain, a `Patient` resource.
2. Determine which record sections are to be retrieved.
3. Retrieve the care record as HTML using the `$getcarerecord` operation.

TODO: decide what this operation would look like.

Request

```
POST [base]/Patient/$getcarerecord
```

Request Body

```
<Operation>
</Operation>
```

Response

```
<Bundle>
</Bundle>
```

Appointment Interactions

Create an appointment for a patient at any location

The typical flow to book an appointment is:

1. Search by `NHS number` for, or otherwise obtain, a `Patient` resource.
2. Search for available `Slot` resources, by some combination of:
    1. preferred date range
    2. preferred `Location`
    3. preferred session type
3. Create an `Appointment` for the chosen `Slot` and `Patient` resources.

Server SHALL maintain resource state in accordance with their own internal integrity constraints, including the state of any associated resources, such as `Slots`, `Schedules`, etc.

Hence, a `201` **Created** HTTP response code implies that the server has checked and, if necessary, updated the state of associated resources, such as the requested `Slot` and/or associated `Schedule`.

TODO: determine what granularity we want to be able to search for schedules and slots. I think in the first instance being able to search by ODS code and optionally ODS site code might be enough? However, we should also consider adding an optional date range for booking in the future.

TODO: define the operation definition for this interaction and make sure it makes it back into the data standards doco AND the above description of what you can search on.

Request

```
POST [base]/Organization/$getschedule
```

Input Parameters

Parameter		Description
odsCode	R	
odsSiteCode	O	
timePeriod	O	`start` is equal to or lower (earlier) than `end`, if both are present.

> R = Required, O = Optional

Output Parameters

Parameter	Description
return	A `searchset` bundle of `Schedule`, `Slot`, `Location`

Request Body

```
<Operation>
</Operation>
```

TODO: add an example request body.

Response

```
<Bundle>
</Bundle>
```

TODO: add an example response fragment.

Update an existing appointment for a patient

The typical flow to cancel an appointment is:

1. Search by `NHS number` for, or otherwise obtain, a `Patient` resource.
2. Search for `Appointment` resources for the Patient resource.
3. Choose an `Appointment` resource and update it.

Request

```
PUT [base]/Appointment/5F6E2C1A-7340-4763-922C-91D1306FBBF8
```

Request Body

```
<Appointment>
</Appointment>
```

TODO: add an example request body (noting only a small subset of fields can be modified).

Response

```
<Appointment>
</Appointment>
```

TODO: add an example response fragment.

Cancel an existing appointment for a patient from any location

The typical flow to cancel an appointment is:

1. Search by `NHS number` for, or otherwise obtain, a `Patient` resource.
2. Search for `Appointment` resources for the `Patient` resource.
3. Choose an `Appointment` resource and cancel it.

A status response `200` **OK** implies that the state of any resources affected by the appointment cancellation (i.e. the associated `Slot`) subsequently reflects the cancellation (for example, `Appointment.status`, `Slot.freeBusyType` are updated inline with any internal integrity constraints).

TODO: we need to check if we really want to copy onto the appointment the schedule practitioner and location details as not having these details will make querying harder down the line and departs substantially from FHIR standard.

TODO: should we provide more guidance on how the participant-actor should be used as this is a mandatory field and we'd realistically expect that if you know the patient then that should always be populated and required? and should default the status to accepted? etc.

Request

```
POST [base]/Appointment/[id]/$cancel
```

Input Parameters

Parameter		Type	Description
reason	O	`string`	Text describing the reason for cancellation

> R = Required, O = Optional

Output Parameters

Parameter	Type	Description
return	`OperationOutcome`	The result of processing the cancellation request

Request Body

```
<Operation>
```

```
</Operation>
```

TODO: add an example request body.

Response

```
<OperationOutcome>
</OperationOutcome>
```

TODO: add an example operation outcome response.

Errors

The server response SHALL contain either an `OperationOutcome` as the single output parameter, or a `5xx` HTTP status code. Servers SHOULD return an `OperationOutcome` in the case of `5xx` HTTP status code.

Task Interactions

> Note: In workshop discussions with all principle GP system vendors it has been agreed that within the scope of GP Connect FoT a Task will be regarded as a one-way "fire and forget" interaction, and thus they are stateless (i.e. there is no mechanism to accept or decline a task) and there is no mechanism in place to directly respond to a task (as might be the case in a threaded discussion or fully fledged workflow system).

Send a task to an organisation requesting an activity relating to a patient

The typical flow to send a task (order) is:

1. Search by `NHS number` for, or otherwise obtain, a `Patient` resource reference.
2. Search by `ODS number` for, or otherwise obtain, an `Organization` resource reference for the task recipient.
3. Search by `SDS number` for, or otherwise obtain, an `Practitioner` resource reference for the task recipient.
4. Create the `Task` for the chosen `Patient`, `Organization` and `Practitioner` resources with the current date (and time?), free text task `description` and coded `reason` (e.g. Action needed).

> Note: it is assumed that you will already have references for your own Organization and Practitioner resources which will be added onto the Task as the `sender` references.

Request

```
POST /Order
```

Request Body

```
<Order>
    <meta>
        <profile value="http://fhir.nhs.net/StructureDefinition/gpconnect-task-order-1-0"/>
    </meta>
    <extension url="http://fhir.nhs.net/StructureDefinition/extension-gpconnect-task-description-1-0">
        <valueString value="Investigate physiotherapy exercise options to assist with dense right sided weakness"/>
    </extension>
    <identifier>
        <value value="4D559C01-C2FE-4614-B8B6-168BC796FE51"/>
    </identifier>
    <date value="2014-12-04"/>
    <subject>
        <reference value="Patient/1A6E1B1C-6340-4663-926C-9CD1306EAAF8"/>
        <display value="Sally Taylor"/>
    </subject>
    <source>
        <reference value="Practitioner/5BF84050-8308-4271-BBC6-F296801714CE"/>
        <display value="Dr. Tom Wood"/>
    </source>
    <target>
        <reference value="Practitioner/30D59E6F-09CF-4FCD-8772-CFA020211B60"/>
        <display value="Mr. Bob Jones"/>
    </target>
    <reasonCodeableConcept>
        <coding>
            <system value="http://fhir.nhs.net/ValueSet/gpconnect-reason-type-1-0"/>
            <code value="2"/>
            <display value="Action needed"/>
        </coding>
    </reasonCodeableConcept>
</Order>
```

Response

> Note: if no `return=minimal` header is specified in the POST request the `return=representation` behavour will be used (as specified earlier in this document) and the response body will thus contain the full details of the newly created resource.

```
<Order>
    <id value="172F53D6-65A0-4BB5-8F34-4BC3EAD8ADEA"/>
    <meta>
        <profile value="http://fhir.nhs.net/StructureDefinition/gpconnect-task-order-1-0"/>
    </meta>
    <extension url="http://fhir.nhs.net/StructureDefinition/extension-gpconnect-task-description-1-0">
        <valueString value="Investigate physiotherapy exercise options to assist with dense right sided weakness"/>
    </extension>
    <identifier>
        <value value="4D559C01-C2FE-4614-B8B6-168BC796FE51"/>
    </identifier>
    <date value="2014-12-04"/>
    <subject>
        <reference value="Patient/1A6E1B1C-6340-4663-926C-9CD1306EAAF8"/>
        <display value="Sally Taylor"/>
    </subject>
    <source>
        <reference value="Practitioner/5BF84050-8308-4271-BBC6-F296801714CE"/>
        <display value="Dr. Tom Wood"/>
    </source>
    <target>
        <reference value="Practitioner/30D59E6F-09CF-4FCD-8772-CFA020211B60"/>
        <display value="Mr. Bob Jones"/>
    </target>
    <reasonCodeableConcept>
        <coding>
            <system value="http://fhir.nhs.net/ValueSet/gpconnect-reason-type-1-0"/>
            <code value="2"/>
            <display value="Action needed"/>
        </coding>
    </reasonCodeableConcept>
</Order>
```

TODO: determine if we should remove the encounter reference as we don't currently have any way to search for encounters.

Appendix 1. GP Connect FoT Conformance Profile

TODO: add in the GP Connect FoT Conformance Profile.

GP Connect FoT Conformance Profile of Type Kind

```
<ConformanceProfile>
</ConformanceProfile>
```

Appendix 2. Code Systems

Business Identifers

Identifier	Resource	System
NHS Number	Patient	http://fhir.nhs.net/Id/nhs-number
Local Identifier	Resource	http://fhir.nhs.net/Id/local-identifier
ODS Code	Organisation	http://fhir.nhs.net/Id/ods-organization-code
ODS Site Code	Organisation	http://fhir.nhs.net/Id/ods-site-code
SDS User ID	Practitioner	http://fhir.nhs.net/Id/sds-user-id

Appendix 3. Integration Prerequistes

Spine Mini Service

Clients SHALL be capable of performing Patient Demographic Service (PDS) tracing of patients to obtain their NHS Number, Date of Birth and current GP organisation.

Clients MAY utilise the Spine Mini Service (SMS) as a lighter weight alternative to integrating with the full Spine2 Patient Demographic Service (PDS).

As the SMS does not return multiple possible matches for the patient it is typically only suitable to be used where there is enough information to achieve a single matched trace.

Spine Directory Service

TODO: get details on how this needs to be integrated for FoT (i.e. LDAP).

Appendix 4. Operation Definitions

The OperationDefinition resource provides a formal computable definition of an operation or a named query. In the scope of GP Connect FoT two operations have been defined as follows:

TODO: include details of the operation definitions for GP Connect FoT.

Get Care Record Operation Definition

```
<OperationDefinition>
</OperationDefinition>
```

TODO: add the operation definition for retrieving an appointment.

Cancel Appointment Operation Definition

```
<OperationDefinition>
</OperationDefinition>
```

TODO: add the operation definition for cancelling an appointment.

Search For Schedules & Slots Operation Definition

```
<OperationDefinition>
</OperationDefinition>
```

TODO: add the operation definition for searching for schedules and slots for a given organisation.

Appendix 5. Terms and Abbreviations

Term	Description
API	Application Programming Interface
Attended consumer	An end consumer that sends an API request within the context of a specific end user of the consumer application.
Consumer	An application that sends an API request to a provider and receives and processes an API response from the provider.
End consumer	A consumer that is not acting as an intermediary, in the context of a given API request
End provider	A provider that is not acting as an intermediary, in the context of a given API request.
Intermediary / Proxy	An application that mediates access to resources exposed by an API, acting as both a provider and consumer. It receives requests from consumers, forwards each request to some other provider and forwards the response from that provider to the original consumer.
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP over TLS
JSON	JavaScript Object Notation
JWS	JSON Web Signature
JWT	JSON Web Token
PKI	Public Key Infrastructure
Provider	An application that receives and processes an API request from a consumer and sends an API response to the consumer.
Consumer	As application that issues a request to an API and receives the API response.
TLS	Transport Layer Security
TLS MA	TLS Mutual Authentication
Unattended consumer	An end consumer that sends an API request outside the context of a specific end user of the consumer application.

Appendix 6. Open Source FHIR Clients

The following open source FHIR client libraries are available on GitHub:

- C#.NET FHIR NET API
- Java HAPI FHIR
- JavaScript Smart on FHIR
- Python Smart on FHIR