FHIR – The Future of Interoperability. Now

FHIR for Populations: Understanding the FHIR Bulk Data API Certification Criteria (Part 1)

Diego Kaminker  diego@hl7.org
HL7 DCSIO
Agenda – April 20

• Session Structure
• Part 1
  – The FHIR Toolbox: FHIR Core Spec & IGs + …
  – Scenarios for the FHIR Toolbox
  – Population Health: BULK FHIR / CQL + Measures
  – Intro to Bulk FHIR
  – Bulk FHIR in Regulation
• Part 2
  – Discussion Zone: Example Scenarios
  – Questions/Answers
The FHIR Toolbox

The stack of FHIR family tools from HL7 supporting a learning health care system

- HL7 FHIR Core Spec
- HL7 FHIR Base IGs
- HL7 FHIR IGs
- Smart-on-FHIR
- CDS-Hooks
- Bulk FHIR Data Access
- CQL
FHIR Tool Box: Knowledge

How can we apply medical knowledge in daily practice?
How can we overcome limitations of our EHRs and use/integrate other apps or services?
Which are the different scenarios?
Which FHIR tools apply?
Scenarios

When do we want to apply knowledge?

Individual Patient

(We are attending a patient, or the patient is using a patient portal)

➔ Independent Knowledge (CDS) Service

We want to provide any EHR with a service: given context and data, apply knowledge and recommend an action course

➔ Embedded/Shared User Interface

We want to show information in a different way, integrated with the EHR/Patient Portal
Population Health

We want to analyze data from a (big) set of patients

➔ Data Extraction

We want to extract data from the EHR for research, or apply discovery algorithms, or to calculate quality / clinical / epidemiological measures

Also, for initial load of a native FHIR server

➔ Population Health Measures

We want to use a service to calculate measures on a given population -> (numerator / denominator)

When do we want to apply knowledge?

FHIR BULK

I ask for the data, and then process it

Extractor

EHR Data Server

Requestor

Measure Server

(I ask a FHIR Server to calculate)
"Big" Data Extraction

We need clinical/financial data about patients, to do some kind of analysis: quality measures, public health reporting, data mining, ML training, etc.

➔ Extractor: requesting some data / parameters for extraction (which data?)

➔ Data Server: prepares the data and alert when the data is available (the process is asynchroneous)

We are talking about SEVERAL (HUNDRED) THOUSANDS of records

"Big" Data / "Extraction" examples

- "Recover vitals, demographics, labs of all patients with 'diabetes' in the last 2 years"
- "Recover all procedures for patient covered by XYZ network in the last 3 months"
- "Recover all medication, clinical notes and demographic data for all COVID19 patients"
Data Extraction

Problems?
- How to guarantee security
- Async Processing
- Efficient format
- One API for each EHR?
- Different format for each extraction?

Who is interested?
- EHR Vendors / Data Exporters
- Public Health/Payors/Researchers (EXT: Extractors)

Different API for the same goal

EHR 1: Easy! Call this API
EHR 2: We cannot do that!
EHR 3: Give us 4 months

EXT 1: Please provide with a CSV file with these fields
EXT 2: Create an XLS file with these columns
EXT 3: Call this service with this XML for each case
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Efficient Data Exchange</td>
<td>ND-JSON</td>
</tr>
<tr>
<td>Asynchronous Data Exchange</td>
<td>FHIR $export API</td>
</tr>
<tr>
<td>Clinical Data Exchange</td>
<td>FHIR Resources (+Appropriate data model:IG)</td>
</tr>
</tbody>
</table>
Bulk FHIR

• Register your App

  When registering the server requires a key-pair (Public/Private) and provides:
  • Client_ID
  • Auth URL
  • FHIR Bulk Data Server URL
Authentication

• Create a JWT with iss/sub = client_id, aud = url de auth, digitally signed with our private key

• The 'scope' is 'what I want to access'. Pretty open. But must be as small as possible (remember BOLA)

Connection with the endpoint of authorization

(SMART Backend Services OAuth Profile)

```javascript
# Agrego el jwt con el claim utilizando el client_id y la clave privada
jwt_claims = {
  'iss': client_id,
  'sub': client_id,
  'aud': token_endpoint,
  'exp': datetime.datetime.utcnow() + datetime.timedelta(minutes=5),
  'jti': uuid.uuid4().hex
}

client_assertion = jose.jwt.encode(
  jwt_claims,
  signing_key,
  algorithm='RS256',
  headers={'kid': signing_key['kid']})

token_request = {
  'scope': 'system:read',
  'grant_type': 'client_credentials',
  'client_assertion': client_assertion
}

# Solicito el token de acceso
token_response = requests.post(token_endpoint, data=token_request)
```
Kick-off request

• Standardized Operation ($export)
• Which kind of resources we want
• Asynchronous: Prefer: respond-async
• Export all patients, a group of patients, or all the resources.
  • [FHIR Server Base]/Patient/$export
  • [FHIR Server Base]/Group/[group id]/$export
  • [FHIR Server Base]/$export

Groups are arbitrary and can be pre-negotiated
# Kick-off request parameters

- **$export parameter (v1)** *(Mandatory)*
  
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_outputFormat</td>
<td>Only ndjson</td>
</tr>
<tr>
<td>_since</td>
<td>Last update date</td>
</tr>
<tr>
<td>_type</td>
<td>Resource classes, comma separated</td>
</tr>
<tr>
<td>_typeFilter</td>
<td>Optional Filtering criteria for resources</td>
</tr>
</tbody>
</table>

- **New in V2 (optional)**
  
  - _typeFilter: (criteria)
  - _elements: (only these)
    - Patient (sp.patients ref)
    - includeAssociatedData (last/all provenance)

---

```python
# Lanzamos la solicitud con una operación $export
kickoff_request_headers = {
    'Prefer': 'respond-asyn',
    'Accept': 'application/fhir+json',
    'Authorization': 'Bearer ' + token_response.json()['access_token']
}
kickoff_request_params = {'_type': 'Observation,Condition,Patint'}
kickoff_response = requests.get(fhir_endpoint + '/Patient/$export', params=kickoff_request_params, headers=kickoff_request_headers)
```
It will return 202 (Accepted) and an URL where we can ask when our 300,000,000 records will be ready for download...

{"Server": "Cowboy", 'Connection': 'keep-alive', 'X-Powered-By': 'Express', 'Content-Location': 'https://bulk-data.smarthealthit.org/fhir/bulkstatus/cfe81d727b7f2e08e35ad77efdefe6ad', 'Content-Type': 'application/json; charset=utf-8', 'Content-Length': '644', 'Etag': 'W/"284-F3AUtGqq00JHYRvr+vui14KUsQE"', 'Date': 'Sun, 14 Nov 2021 23:03:33 GMT', 'Via': '1.1 vegur"}
Polling

We ask in Content-Location until we get a 200 instead of 202

Status: 202 Accepted
X-Progress: “50% complete”
Retry-After: 120
Polling

And when we get 200...

Response code: 202
Response code: 200
Response body: 

```json
{
  "transactionTime": "2021-11-14T23:03:33.337Z",
  "request": 
    "https://bulk-data.smarthealthit.org/eyJlcnIiOiJiLCJwYWdlIjoxMDAwMCwiZHVyIj...SI6MSwic3R1Ijo0LCJkZGVhc29ufHJvc3Rpbmc6c3RlbjI9MCwia2l0aCI6NjM5LCJzZWN1cmUiOnRydWV9/fhir/Patient/$export
    ?_type=Observation%2CCondition%2CPatient"
,  "requiresAccessToken": true,
  "output":
    [  
      {  
        "type": "Condition",
        "count": 639,
        "url": 
          "https://bulk-data.smarthealthit.org/eyJpZCI6ImNmZTgxZDcyN2I3ZjJlMDhlMzVhZD...2Zmc2V0IjowLCJsaW1pdCI6NjM5LCJzZWN1cmUiOnRydWV9/fhir/bulkfiles
            /1.Condition.ndjson"  
      }  
    ]
}
```

The manifest tells us how many resources per file and the url for the files
Data Access

**FHIR resources**

(But not just ANY FHIR resource : constrained!)

**Structure: FHIR Resources**
Specific profile so we know what to expect on content/terminology

**Efficient: ndjson**

Resources in the same file. No Spaces or CR/LF

http://ndjson.org/

**FHIR core**
- Patient.identifier: 0..*
- Patient.name 0..*
- Patient.gender: 0..1

**'Our' FHIR**
- Patient.identifier: 1..*
- Patient.name: 1..*
- Patient.gender: 1..*
- Observation.code.system = LOINC
And then…?

The standarization "succeds" when we get our ndjson files

- We can upload our FHIR resources to any JSON / FHIR native repository (Google BigQuery - example) and then use them
- Can be also ELK Elastic/Logstash/Kibana

```sql
SELECT
  json_extract ( Observation.json , '$. subject.reference ' ) paciente,
  json_extract ( Observation.json , '$. effectiveDateTime ' ) fecha,
  json_extract ( coding.value , '$.code' ) test,
  json_extract ( Observation.json , '$. valueQuantity.value ' ) resultado
FROM
  Observation
```
But how?

BULK FHIR to SQL Example

• [https://bit.ly/3xmChS1](https://bit.ly/3xmChS1)

• Obtains Patient / Observations for all patients in the EHR

• Create an Histogram for Total Cholesterol
The Standard

• 2019 – HL7 Publishes the first version of the Bulk Data Access FHIR Implementation Guide (STU1)

• 2021 – Second version of Bulk Data Access (STU2) published

Regulation (a)

(1) ONC’s Cures Act Final Rule added a new certification criterion to the ONC Health IT Certification Program at §170.315(g)(10) for standardized APIs for patient and population services.

(2) ONC also adopted a suite of regulatory provisions that address health IT developers’ business practices associated with certified API technology through the API Condition of Certification. Health IT developers seeking certification to this criterion are required to implement an API using the HL7 FHIR standard to make patient data available to users.
Regulation (b)

- The criterion also requires developers to build capabilities for standardized export of large quantities of health information according to the Bulk Data IG.
- Support of STU1 is mandatory, but STU2 can be also supported.
- Adoption of “bulk data” APIs by certified health IT products represents the culmination of more than five years of work to enable more efficient access to large sets of health data.
- Support for FHIR API for Patient Access
- Support for FHIR Bulk API for Provider/Payor Access
Implementers Tools (I)

- **Bulk Data Server** – A configurable reference implementation of a bulk data server conforming to the Bulk Data IG.

- **Bulk Data Client** – A sample client application using a command-line interface to test bulk data servers.
Implementers Tools (II)

- **Bulk Data Test Suite**— A web-based online tester for bulk data servers that contains a test suite and test runner. The test suite has been integrated into the Inferno test tool.

- **Bulk Data Tools**— A library for working with and converting bulk data into different formats (json, ndjson, csv, tsv)
Discussion (1)

https://bcda.cms.gov/

Beneficiary Claims Data API

Provides FHIR-formatted bulk data files to an ACO for all of the beneficiaries eligible to a given Shared Savings Program ACO. BCDA does not require individual beneficiary authorization but does allow a process for patients to opt out of data sharing.

BCDA’s Parameters

Parameters are options that can be passed to the endpoint to influence the response. Parameters can be used to filter or select for certain desired data. We would like to highlight two query parameters used in BCDA.

1. The type parameter

The _type parameter allows you to request different Resource Types from the API. Instead of receiving data from all three Resource Types when no _type parameter is specified, you will be able to use the _type parameter to submit one or more Resource Types. The API will then produce data from the specified Resource Types.

To learn more about the Resource Types see the FHIR Resource Types section above.

2. The since parameter

The _since parameter allows you to apply a data parameter to your bulk data requests. Instead of receiving the full record of historical data every time you request data from an endpoint, you will be able to use _since to submit a date. The API will then produce claims data from the bulk data endpoints that have been loaded since the entered date.

To learn more about the differences using _since between the /Patient and /Group endpoints see the FHIR endpoints section above.

BCDA Version 2 (V2)

Version 2 of the Beneficiary Claims Data API (BCDA V2) is available as of Summer 2021.
Discussion (2)

• What if?

Bulk FHIR!

Our discussion should be limited to:

Which patients?
Which FHIR resources?
Which elements?
Which criteria?