2020-09 Argonaut Patient Lists

Patient Lists

SCHEDULE:

KICKOFF: Wednesday Sept 9 4PM Eastern

HANDS-ON Thursday-Friday Sept 10 9-5 Eastern

Breakouts: Thursday 4 Eastern (3 Central, 2 Mountain, 1 Pacific) Location: Argonaut Patient Lists Track Event Stream

Topics: Scenario Discussion Items/TBD

Friday Noon Eastern (11 Central, 10 Mountain, 9 Pacific) Location: Argonaut Patient Lists Track Event Stream

Topics: Scenario Discussion Items/TBD

Wrap-up: Friday 3 Eastern

TRACK REPORT:

Argo Patient Lists Summary Slides

TRACK REPORT:
- Argo Patient Lists Summary Slides

Track overview

Short Description

One of the 2020 Argonaut projects is Patient Lists, which enables a standardized way for software to access lists of patients which are commonly used in an EHR setting. For example, "all patients in a location" or "all the patients on my schedule today."

This track will cover some foundational, early-design operations such as list discovery, selecting list member records, and selection of "extra" patient data using various methods.

Long Description

User-facing apps often need to know things like:

- “who are the patients I’m seeing today,”
- “who are the patients I’m responsible for in the hospital right now,”
- “who are the patients in this ward.”

This is core functionality supported by existing EHR systems. In FHIR, various methods have been used such as the standard search API or assembling the List or Group resource. However, no standard or guidance for creation of, and manipulation of, patient lists currently exists.

Some project goals that will be addressed (and hopefully clarified with this connectathon experience):

- Supporting interoperable and standard exchange of existing EHR supported “user-facing lists”.
    1. user-facing lists include both “system-maintained” and “user-maintained”(which are entirely ad-hoc - the user explicitly selects and manages members) lists
- Defining a general framework using the FHIR API for exposing existing EHR user-facing patient lists so that EHR systems can expose any list they choose to define.
- API would allows user-facing apps to:
    1. Discover user-facing lists including searching on existing lists using limited set of predefined characteristics such as location or careteam.
    2. Fetch the List, allowing apps to enumerate members of a user-facing lists
    3. Provide a framework to convey extra details about the members of a lists

EHR vendors and third party software vendors wishing to use this new API are welcome to participate in this track.

Type

Test the design of a Resource/set of Resources

Submitting Work Group/Project/Accelerator/Affiliate/Implementer Group

Argonaut

Track Leads

Carl Anderson, carl.anderson@microsoft.com

Eric Haas ehaas@healthedatainc.com

Related tracks

Bulk Data

FHIR Version

FHIR v4.0.1

Specification(s) this track uses

The track spec is still under active development, so an existing reference is not fully available. However, much of the relevant work and examples is present in this demo app: https://aka.ms/patient-lists-demo.

There is also a WIP codelab exercise which walks through some of the implementation of the demo app: https://aka.ms/patient-lists-codelab

Further project-management information may be useful to interested parties and can be found here:

- http://bit.ly/argo20-lists
- https://hackmd.io/AfJ9YNb6TNGeDSuAaHIn1g?view

Clinical input requested (if any)

We're running with a handful of example patient lists, such as "all the patients in a location" and "all the patients a provider will see in a day / week" - but if there are other more relevant lists, suggestions from clinicians are welcome.

It has also been suggested that once a list of patients has been identified for selection, there may be relevant pieces of information that are not directly attached to the patient record, such as:

- date of the most recent encounter
- admit date
- scheduled discharge date
- last provider to visit the patient
- current room number
- etc

Any insight into other relevant data points that may be useful when working with patient lists will be most welcome by track participants.

Patient input requested (if any)

Same as above - if there are any well established places where lists of patients are frequently used or useful, patient/caretaker input is welcome.

Expected participants

signup spreadsheet

Server Implementers:

- Epic
- Cerner
- Meditech
- Allscripts?
- MIcrosoft

Client Implementers:

- Apple
- Epic?
- PeraHealth?
- MIcrosoft
- Health eData Inc

Zulip stream

https://chat.fhir.org/#narrow/stream/227046-Argo-Patient.20Lists

Track Orientation

Wednesday, August 26 1-3PM - Eastern (10:00 – 11:00am Pacific) https://meet.jit.si/argo2020.lists

Track details

System Roles

Server implementers (EHR vendors)

Source of User-Facing lists: Provide Group resource endpoints for discovering what lists are available and fetching lists

Clients (third-party software vendors): Application to discover what User-facing lists are available and fetching lists for display, processing etc.

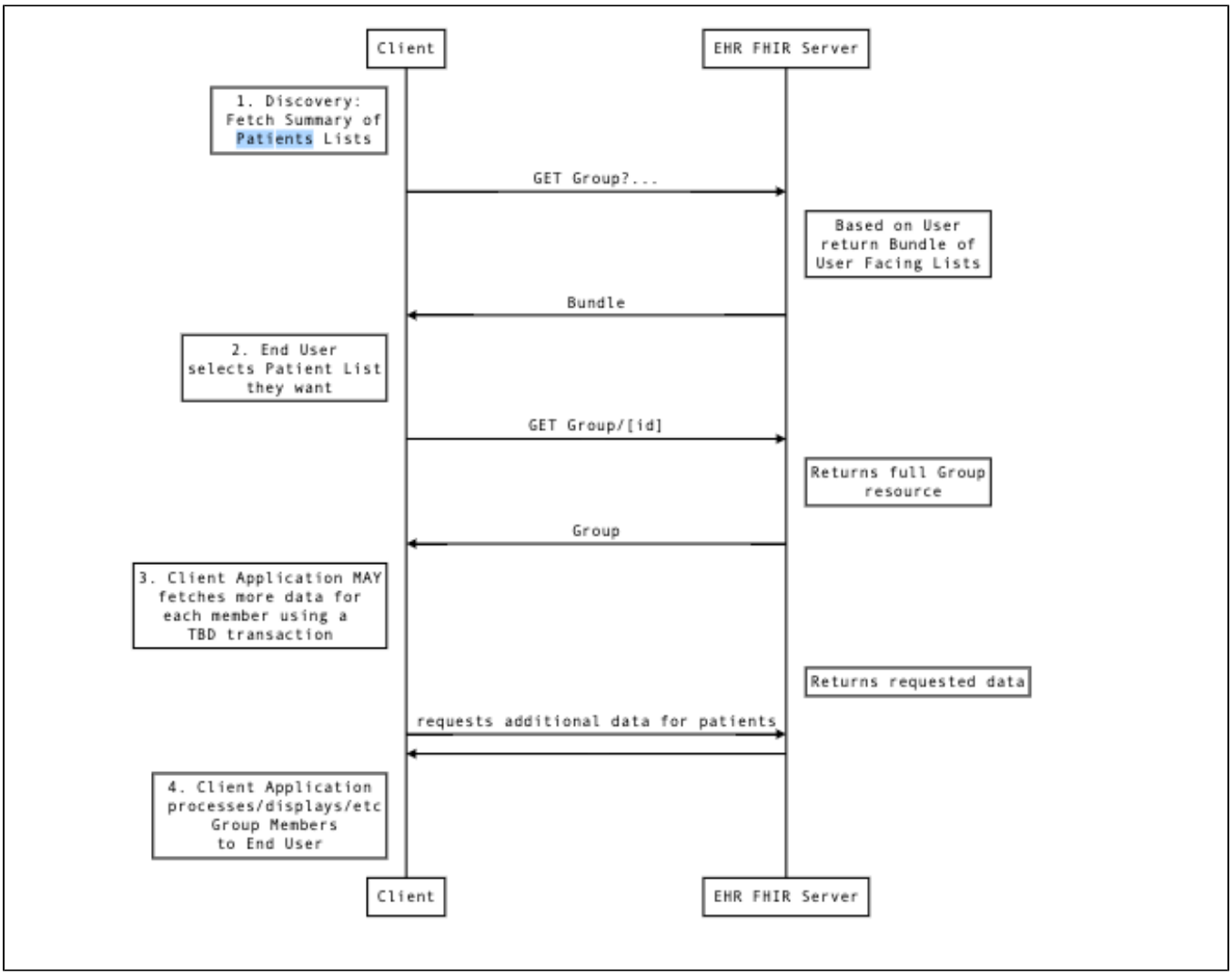
Scenarios (The Argo Patient List Client (http://www.healthedata1.co/) is an interactive app that walks through all the following scenarios.)

- A dockerized HAPI server will be provided that is populated with a canonical set of Synthea-generated FHIR data. This server image, and the scripts used to generate it, will be made available both to server implementers and third party vendors for testing purposes prior to the connectathon. Instructions will be provided in github and in the introductory video.
- A script will also be made available that can generate the Synthea data and load it into an empty server, which will be useful for the server implementer participants. Instructions to BYOS (bring your own server) will be made available.

Sample Data:

FHIR Transaction Bundle of sample data used for this connectathon on the UHN HAPI Server:

Note that the IDs mentioned in the scenarios below are illustrative only



1 - Patient List Discovery

As a side note, all of these scenarios below assume that the server supports, and the client specifies, _summary search parameter in queries for patient list Groups from the server and implements as discussed:

- Use _summary and require that Servers **Shall** return all the Group summary elements **+ Group.characteristic element**
  - (Based on summary behavior in base specification)

1.1 - Basic Discovery all User Facing Lists

A server is able to produce a list of the available Patient Lists known. A client is able to render a list of known Patient Lists.

Action: A client issues a GET request to a server:

```
GET Group?_summary=true&type=person
```

Server Success Criteria

The server responds with a complete Bundle of the available Group entries with type person.

Client Success Criteria

The list of lists is processed, e.g., displayed in HTML.

1.2 - Organization-Managed Lists

A server is able to provide a collection of patient lists that are managed by a particular organization. A client is able to query for lists, specifying the managing Organization in the request.

Action: A client issues a GET request to a server:

```
GET Group?_summary=true&type=person&managingEntity=Organization/42
```

Server Success Criteria

The server responds with a Bundle of Group entries where each entry is managed by an Organization with an ID of '42'.

Client Success Criteria

The client provides a selector listing available Organizations. When selected, a query returns the patient lists that are managed by the selected Organization.

1.3 - Discovery by Characteristic

A server is able to provide a collection of patient lists that all have a common characteristic. A client is able to query for lists, specifying one of the following characteristic parameters in the request:

Characteristic	code	Characteristic Value	Value
characteristic	location	value-reference	Location/[location_id]
characteristic	attributed-to	value-reference	Practitioner/[practitioner_id]
characteristic	attributed-to	value-reference	Organization/[organization_id]
characteristic	team	value-reference	CareTeam/[careteam_id]

value-reference is a custom SearchParameter formally defined here:

Action: A client issues a GET request to a server:

```
GET Group?_summary=true&type=person&characteristic=[Code value]&characteristic-reference=[Value value]
```

Server Success Criteria

The server responds with a Bundle of Group entries where each entry has the same characterisic code/value pair (e.g., location=Location/[location_id])

Client Success Criteria

The client provides a selector listing above characteristic name-value pairs. When selected, a query returns the patient lists that are managed by the selected Organization.

2 - Patient Lists Members

Fetch a Group containing a list a Patient who are members of the Group. A server returns a Group resource which contains references to patients.

Action: A client issues a GET request to a server:

```
GET Group/123
```

Server Success Criteria

The server responds with a Group resource with ID '123'.

Client Success Criteria

The client queries for a particular list of patients and processes them e.g., displayed in HTML as a list of ids.

3 - Getting Extra Details about the patients

3.1 - Patient Lists - Extra Details via Base FHIR RESTful API Search

The Simplest approach for the client to do a series of queries on the Server to Fetch additional data:

When requested, a server can provides patient details for each for each of the members in the Group resource via a series of FHIR RESTful queries for other resources about that patient as described in the base specification and US Core.

Action: A client issues a GET request, fetching a Group resource. Then, for each Group.member (aka patient), and **Patient** resource and a Observation attribute (the most recent lab result) is requested, which is not directly part of the patient resource.

```
GET Group/123
for each patientRef in Group.member:
  GET Patient/ID
  GET Observation/$lastn?patient=Patient/ID&category=laboratory
```

Server Success Criteria

The server responds with a search Bundle for each query.

Client Success Criteria

The request bundle is properly prepared. All fetched 'extra details' are then also processes them e.g.,displayed in HTML.

Discussion 1) Some queries can be combined using the OR search parameter to reduce the number of Client queries.

```
GET Patient?_id=ID1,ID2,ID3,...
GET Observation/$lastn?patient=ID1,ID2,ID3,...&category=laboratory
```

Server Success Criteria

The server responds with a search Bundle for each query.

Client Success Criteria

The request bundle is properly prepared. All fetched 'extra details' are then also processes them, e.g., displayed in HTML.

Note that US Core does not require servers to support multipleOr for these queries.

Should this be part of the basic Patient list API - e.g., Servers SHALL/SHOULD/MAY Support?

Discussion 2) These queries can be done separately as described above or as a single batch interaction

```
  POST [base]

{
  "resourceType": "Bundle",
  "id": "bundle-request-groupdetails",
  "type": "batch",
  "entry": [
    {
      "request": {
        "method": "GET",
        "url": "Patient/123"
      }
    },
    {
      "request": {
        "method": "GET",
        "url": "/Observation/$lastn?patient=Patient/123&category=laboratory"
      }
    },
    {
      "request": {
        "method": "GET",
        "url": "Patient/456"
      }
    },
    {
      "request": {
        "method": "GET",
        "url": "/Observation/$lastn?patient=Patient/456&category=laboratory"
      }
    },
    ...
  ]
}
```

Discussion 3)

Should this be part of the basic Patient list API - e.g., Servers SHALL/SHOULD/MAY Support?

Server Success Criteria

The server SHALL return a Bundle with type set to `batch-response` that contains the request resource for each entry in the batch request, in the same order, with the outcome of processing the entry.

Client Success Criteria

The request bundle is properly prepared. All fetched 'extra details' are then also processes them, e.g., displayed in HTML.

3.2 - Using _include:

Support _include for Group so that the Patient resource attributes such as Name, Age, DOB, Gender, Height, Weight, etc can be fetched in a single interaction (this is functionally akin to using the `_list` parameter)

```
GET [base]/Group/1234?_include=Group:member
```

Server Success Criteria

The server responds with a complete Bundle of Patient entries, all members of the requested Group with ID '123' AND a Patient resource for each Group.member.

Client Success Criteria

The client queries for a particular list of patients and processes them e.g.,displayed in HTML as a table of patient resource attributes such as Name, Age, DOB, Gender, MRN, Contact info.

Discussion

Should this be part of the basic Patient list API - e.g., Servers SHALL/SHOULD/MAY Support?

3.3 - Patient Lists - Extra Details via Questionnaire

See full example of this exchange: https://hackmd.io/AfJ9YNb6TNGeDSuAaHIn1g?view#Patients-with-column-data

When requested, a server provides patient details that are not present in the patient resource directly *via a Questionnaire and QuestionnaireResponse as follows:*

1. A client issues a GET request, fetching a patient list (aka Group resource). The Group resource has an extension that references a Questionnaire url which defines the extra data to be returned for that group of patients and a second extension for each Patient entry in the Group referencing a QuestionnaireResponse resource containing the patient-level data. *(See example extension).*
2. The server is able to populate corresponding QuestionnaireResponses with the appropriate data for each patient.
3. Then, for each patient in the list, the client may fetch the QuestionnaireResponse for the above Questionnaire.

```
GET Group/123
```

Server Success Criteria

The server responds with a Group resource with ID '123' that has the two extensions: a Questionnaire resource and a QuestionnaireResponse resource for each patient.

Client Success Criteria

The client queries for a particular list of patients. The Client make All extra patient details are extracted from a QuestionnaireResponse resource.

Option 1: Get QR by resolving an extension QR ID

```
for each patient in Group/123.bundle.entry:
  GET QuestionnaireResponse/[QuestionnaireResponse resource id from Group.member.entity.
extension]
```

Option 2: Get all QR for groups based on Q url

```
GET QuestionnaireResponse?q=/[Questionnaire url from Group.extension]
```

Server Success Criteria

The server responds with search Bundle with each entry a QuestionnaireResponse resources for each patient containing the requested patient attributes.

Client Success Criteria

The Client extracts the extra patient details from the QuestionnaireResponse resources and processes them e.g., populates a table for display.

Discussion

1. identify how the appropriate questionnaire is determined for a group or particular context.

2. This information could easily be transmitted as a binary (csv) or in the Group.member.display element as a delimited string. Is the added complexity worth it and what are the real benefits here?

TestScript(s)

A demo app is provided here: https://microsoft-healthcare-madison.github.io/patient-lists-demo/

The source for the demo lives here: https://github.com/microsoft-healthcare-madison/patient-lists-demo

Instructions for generating and loading sample data can be found in the demo app's README.

Security and Privacy Considerations

N/A