Event Streams for FHIR Bulk Data Access

Matt Ping 11/30/2020

FHIR Bulk Data Specification

- The current FHIR Bulk Data specification supports requesting the generation of a bulk data file.

- This implementation asynchronously creates a new line delimited JSON (ndjson) file for processing by a consuming application/server

- The pattern is helpful in bulk file loads and “ETL-style” processing

Current Improvement Opportunities

- Bulk Data Access currently only supports async batch-style processing. This helps in bulk data loads, but is gapped around real-time streaming of larger data payloads (e.g. DaVinci Quality Measures Care Gaps)

- An async response in the current bulk data environment requires an endpoint for requesting data, a polling endpoint for status, and a location to pull a file from. This creates management overhead. In some circumstances this overhead may not be necessary

- Some business use cases may have a need for data to be updated in a (near) real-time scenario

Possible Bulk Data Improvement

- Support event streaming concepts to handle real-time and near real-time data exchange
- This can enable reactive architectures for teams implementing data processing using FHIR
- Standards exist for server streaming APIs. Specifically Server Sent Events (EventSource) - https://en.wikipedia.org/wiki/Server-sent_events
  - W3C specification - https://www.w3.org/TR/eventsource/

Server Sent Events

- EventSource allows for a client to receive a “push” of data when contacting a server.
- The HTTP payload that is returned is a standard specification made up of at least one of `event`, `data`, `id`, and `retry` elements
- Rather than wait for all of the data to be loaded, a server sends back individual small payloads as they are ready
- Since this is based on HTTP headers and data response, it is language independent.

Possible Implementation Details

- Endpoint: `GET [fhir base]/[Resource]/$export`
- Headers:
    - `Accept: application/fhir+json`
    - `Prefer: respond-eventsource`
- QueryParameters:
    - `_since` & `_type` can follow the same recommendations as the existing bulk data specification
    - `_outputFormat` would not be necessary for EventSource implementations

Possible Implementation Details

- Response Payload(s):
    - HTTP Status Code: `200` (Success) `/ 5XX` (Error)
    - HTTP Payload:
        - event: `<FHIR Resource Type>`
        - data: `<Individual FHIR Resource>`

Possible Benefits

- Better support for reactive architectures on solutions
- Support real-time and near real-time data exchange scenarios
- Reduce the number of endpoints to implement a bulk data request
  - No need for Request, Polling, Delete, and File Request capabilities
- More closely aligned single request and bulk request implementation
- Language independent solution

Possible Drawbacks

- Event streams may not make sense for all bulk data scenarios. The implementing solution needs to consider the use cases for the data and their consuming solutions
- Need to watch/monitor how many open connections a server has