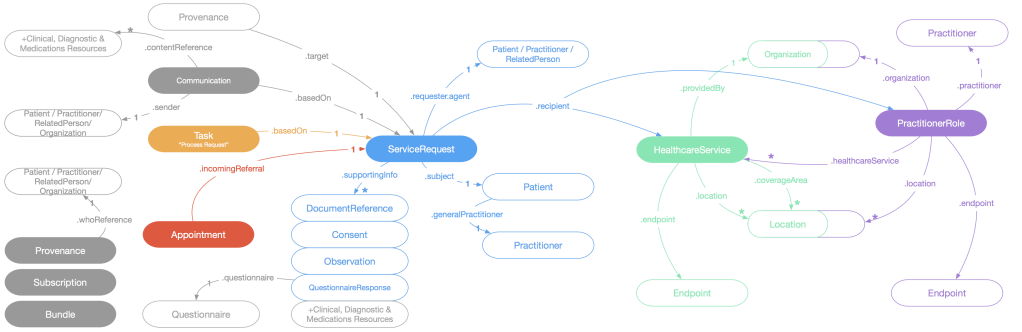
CANADIAN FHIR eREFERRAL SPECIFICATION - DESIGN CANVAS

Sponsored by Canada Health Infoway

eReferral Resources Relationships

This diagram illustrates the relationships between all resources used in an eReferral.
Dark filled resources are the primary resources used in API interactions, white filled resources are ancillary connected resources.



Process & Workflow

- a "Process Request" task is used to track the referral status
- a "Request Communication Response" task is used to make requests for others to send /Communication resources.
- an /Appointment can be attached to a ServiceRequest, either at the time of submission or later.

eReferral Payload

- /ServiceRequest contains all of the information needed to process a referral
- .supportingInfo references additional resources that the requester thinks will be useful for the provider to process the referral (such as attachments, observations, etc…)
- Observation resource should be prioritized before QuiresitonnaireResponse for h holding additional data, as it is better suitable for computable interoperability

Healthcare Service Directory

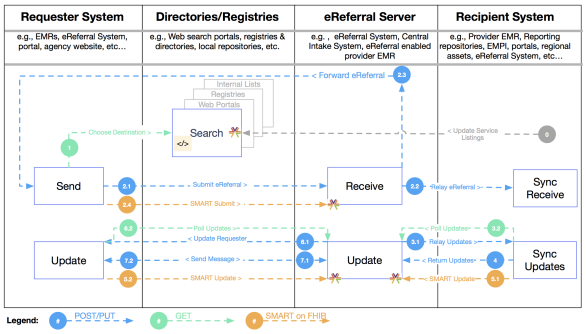
A "Healthcare Service Directory" is service oriented.
e.g., find an immunization clinic, find an adult day program, etc…
Search returns a /HealthareService resource.

Provider Registry

A "Provider Registry" is practitioner oriented.
e.g., find a cardiologist, find Dr.Smith

Search returns a /PractitionerRole resource.

eReferral Flows



Central Intake Notes

Method 1 - Chaining Referrals:
- "Central Intake" is a type of service that receives referrals, analyzes and then decides where to direct that referral. Central intake may add additional data to a referral.
- A "central intake" service should usually have its own service listing, distinct from the services it is connecting referrals to.
- When central intake "forwards" a referral to another services (or services), a new / ServiceRequest resource is created for each destination.
- The new "Forwarded" referrals reference the source referral using the ServiceRequest.basedOn field. This can create a chain of referrals.
- All participants in the forwarded referral chain should have access to each ServiceRequest in the chain.
- If some participants should NOT see some ServiceRequests in the chain, then
a) consider breaking the chain and creating a new referral that is not .basedOn a previous referral or
b) the eReferral server can hide the reference to the chained ServiceRequest values from the parties that should not see them.

Method 2 - Modifying Original Referral:
- An alternate method to support a central intake process would be to modify the original ServiceRequest to reflect the decisions made by the router. It is recommend that his method only be used when referrals are routed within the same organization as the original referral, or for routing actives such as choosing a specific specialist within a target specialty.
- Practically, this method is the same as the process of editing a referral, and should require no additional logic to support from the requester system or recipient system.

eReferral Operations

0 Update Service Listing @ Recipient System
API Operations
PUT /healthcareService, /Organization, /Location /PractitionerRole, /Location

1 Choose Destination @Requester System
(from a /HealthcareService Directory)
API Operations
GET /HealthcareService/
GET /HealthcareService/?<search>
Note: Can _include & /Organization and Location

1 Choose Destination
(from a /ProviderRole Directory)
API Operations
GET /PractitionerRole
GET /PractitionerRole/?<search>

— The assumption is that an eReferral should always be directed to a /HealthcareService — More discussion required

2.1 *Submit eReferral @Requester System
API Operations
POST /Bundle (ServiceRequest, Patient, Practitioner, .supportingInfo.*)

Note 1: The target /healthcareService must be included in the / ServiceRequest, and transmitted to the service's /Endpoint
Note 2: On reception of new /ServiceRequest, create the "Process Request" /Task @eReferral Server
Note 3: Include a text/html summery of the entire referral in the narrative of the /ServiceRequest
Note 4: Put additional data that is not captured in the resources into the .note element (or as supportingInfo?).

2.2 Relay eReferral @eReferral Server
API Operations (via /subscription)
POST <Same as "2.1 Submit eReferral">

Note: The eReferral server may modify the eReferral before relaying it to the Recipient System, such as assigning a resource identifier, assigning a service, etc…

2.3 Forward eReferral @eReferral Server
API Operations
POST <Same as "2.1 Submit eReferral">

Note 1: The eReferral server acts as a requester system to create a new "forwarded" ServiceRequest (e.g., central intake process)
Note 2: Set ServiceRequest.BasedOn to reference the original /ServiceRequest. This can create a "chain" of related eReferrals

2.4 SMART Submit eReferal @Requester System
Resources Requested: /Patient, /Practitioner (+ clinical resources?)

Note 1: The SMART app (eReferral server) will present the eReferral business logic interface. The client's in-context /Patient (or / ServiceRequest) resource will be requested for use in the eReferral (e.g., pre-fill a form, perform matching criteria).
Note 2: (??) A /ReferralRequest resource will be POSTed back to the calling system. If this is not supported, a text version of the ServiceRequest can be POSTed to a /DocumentReference resource

3.1 Relay Updates @eReferral Server
API Operations (via /subscription)
a) PUT /ServiceRequest
b) PUT /Task (task.code.text="Process Request")
c) POST /Bundle (Communication, DocumentReference)
d) PUT/POST /Appointment (when eReferral server manages appointments)

3.2 Poll Updates (by recipient) @Recipient System
API Operations (via polling eReferral Server)
a) GET /Task/?_lastUpdated=<date range>
b) GET /Task/?_authored-on=<date range>
c) GET /Task/?<search>$related

Note: $related option returns all resources connected to /Task except / HealthcareService & /DocumentReference

4 Return Updates @Recipient System
API Operations
a) PUT /Task (Update "Process Request")
a) POST /Appointment, /Communication, /DocumentReference
b) PUT /Appointment, /ServiceRequest, /Patient, /Practitioner

5.1 SMART Update @Recipient System
Resources Requested: /Patient or /ServiceRequest, /Practitioner

Note: Open the in-context patient's eReferrals, or in-context eReferral, user can update as per business logic available in SMART app.

5.2 SMART Update @Requester System
- Same as 5.1 "SMART Update" -

6.1 Update Requester (subscription) @eReferral Server
API Operations
a) PUT /ServiceRequest
b) PUT /Task (task.code.text="Process Request")
c) POST /Communication, /DocumentReference
d) PUT/POST /Appointment

6.2 Poll Updates (by Requestor) @Requester System
- Same as 3.2 "Poll Updates (by recipient)" -

7.1 Send Message @eReferral Server
API Operations
a) POST /Bundle (Communication, DocumentReference)

Note: When the task (eReferral) is completed, an outcome is sent back via /Communication, and referenced by task.output.value

7.2 Send Message @Requester Server
- Same as 7.1 "Send Message", excluding Note -

eReferral Supported Tasks

The eReferral process supports the following task codes definition to support specific workflows.

"**Process Request**": Track the status of an eReferral status over its lifecycle. This task is created by the eReferral server on submission, and closed when the referral processing is complete. The task.status updates in parallel with the similar (but more broadly defined) ServiceRequest.status. In addition, Task.BusinessStatus can also be used to communicate custom deployment specific statuses (e.g., waitlisted, waiting for call-back, pending assessment completion, etc…)

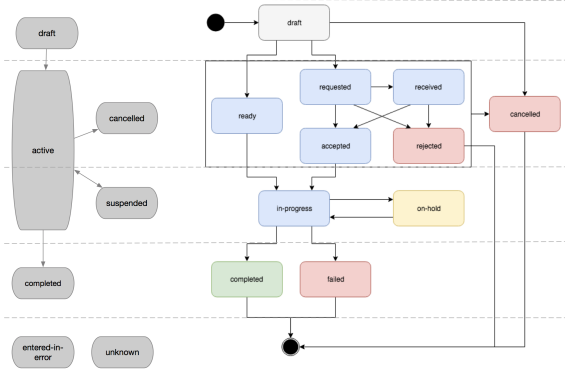
"**Request Communication Response**" (a.k.a., "Request for Information"): Send requests and track the status of requests for information between eReferral requesters and receivers. The request is made in the form of a /Task (e.g., "please send Assessment X"), and the response comes in the form of a /Communication which may be text/html and include /DocumentReference with attachments.

/Task: "Process Request"

Task.BasedOn = reference(ReferralRequest)
Task.Code.Text = "Process Request"
Task.BusinessStatus = <custom_deployment_dependant>
Task.Status = <see table below>

ServiceRequest Statuses
ServiceRequest.status = _

Task State Machine
Task.status = _



ServiceRequest .status =	*Task .Status =*
draft	**draft** - when a referral has not yet been submitted
active	**ready** - after submitted, if provider has no "acceptance" step
active	**requested** - if provider has a accept/reject process
active	**received** - if the provider acknowledged receipt
active	**accepted** - if provider accepts referral for full processing
active	**rejected** - if provider rejects referral
cancelled	**cancelled** - if provider OR requester cancels (but wasn't rejected)
active	**in-progress** - while the provider is seeing the patient (rarely used)
suspended	**on hold** - while patient is in a "wait list" state
completed	**completed** - referral was fully processed
completed	**failed** - could not complete referral (e.g., no show, insufficient documentation, etc.)