OPEN API DOCUMENTATION FOR CLIENT APPLICATION DEVELOPERS

Interoperability Engine 2017

Updated February 15, 2018

This API document is for the interface from Eprosystem Inc., EHR: EproMedical version 3.0.0 to EMR Direct, Interoperability Engine.

1. Introduction

This guide is written for third party developers, including patients, who are developing software applications for accessing Protected Health Information (PHI) based on this documentation of an open API. This documentation allows applications to query a public-facing API enabled by a data holder. Data holders wishing to publish such a public-facing API should have their Health IT vendor register as a developer integrator of EMR Direct Interoperability Engine services at https://www.emrdirect.com/subscribe-developer.
ALWAYS KEEP IN MIND THAT ONLINE DATA TRANSFER IS NOT A SUBSTITUTE FOR PERSON-TO-PERSON COMMUNICATION OF URGENT OR CRITICAL MEDICAL INFORMATION.

This documentation also contains general information and important security information. General information will be marked as “Note:” and important security information will be marked as “IMPORTANT:”.

2. General Concepts

a. Application Access Requests
The API is a read-only RESTful HL7 FHIR® R4 API and follows the syntax described at https://hl7.org/fhir/R4/http.html.

All RESTful data access requests will be in the following format, in which the [base] URL will need to be obtained from the Data Holder, either directly or via directory information: `GET [base]/[resource-specific parameters]…`
FHIR Bulk Data Access Group export is also supported as described in the Bulk Data Access IG.

b. Connecting to the server

The server is accessed by clients through an https connection.

Important: Local customer security policies must be in place to prevent unauthorized monitoring or eavesdropping of connections to the server.
Important: Only TLS 1.2 connections are accepted. All plaintext connections will be refused.

Note: Please limit your connection frequency to a value appropriate for your use case. Connection attempts which are more frequent than permitted by the bandwidth allocation for the data resource are not allowed.

c. Authentication – Obtaining an Access Token

Prior to making API requests, the client application must obtain an Access Token from the associated Authorization Server. The client software must support either the OAuth 2.0 authorization code grant or client credentials grant flow as detailed in RFC 6749. For authorization code flow, a client ID and client secret are required. If the client application does not have a client ID and client secret for this purpose, the client application may obtain a client ID and client secret using the dynamic client registration protocol by submitting the required client information to the registration endpoint as detailed in RFC 7591, by accessing the registration endpoint URL with a browser and completing the online form, by completing the manual registration process at the EMR Direct website, or by using UDAP Dynamic Client Registration. Please see section 3(j) below for the list Client app attributes required for registration. For client credentials flow, a client ID and registered public key are required. Public keys for client credentials grant access may be registered using UDAP Dynamic Client Registration or by contacting the Data Holder directly.
For client applications using authorization code grants, end user authentication is performed using a username and authentication credentials provided by the Data Holder. A healthcare organization may reuse existing patient portal credentials for this purpose, in which case the authenticated username maps to a unique patient portal user on the resource holder's side. The end user should obtain these credentials directly from the Data Holder healthcare organizations from which they wish to access data.

Note: Usernames and authentication credentials are established and reset by the Data Holder.

When an end user is directed to the authorization endpoint, the user will be presented with a login screen where they can enter their credentials for the healthcare organization they are accessing. If the correct credentials are supplied and the end user grants access to the client application, an authorization code will be returned to the client that the client application can use to obtain an access token through the token endpoint as described in RFC 6749. For client credentials flow, the application obtains an access token directly from the token endpoint without user interaction.

Each healthcare organization has a unique base URL that is used to access its Authorization Server. The required endpoint URLs are of the following form:

ENDPOINT	URL
Authorization	https://[baseOAuthURL]/authz

Token	https://[baseOAuthURL]/token
Registration	https://[baseOAuthURL]/register
Manage	https://[baseOAuthURL]/manage
Revoke	https://[baseOAuthURL]/revoke

The complete OAuth 2.0 endpoint URLs for a given FHIR endpoint are published in that Data Holder's well known SMART configuration endpoint, as per Section 4 of the HL7 SMART App Launch specification.

Refresh tokens: Client apps that use the authorization code flow and can securely store refresh tokens may request a refresh token by including the `offline_access` scope in the initial authorization endpoint request. Note that the end user may remove this scope during the authorization process. If a refresh token is permitted, it will be returned together with the access token as per RFC 6749, and can be exchanged for a new access token as per RFC 6749. Refresh token lifetime defaults to 90 days (but may be longer or shorter depending on the Data Holder's institutional policies). Each refresh token may be used once; a replacement refresh token will be issued when the client app uses a valid refresh token to obtain a new access token. Note that refresh tokens are not issued to Client apps that support client credentials code flow as these clients can authenticate directly to the token endpoint to obtain a new access token.

PKCE support: Proof Key for Code Exchange (PKCE) as defined by RFC 7636 is supported for Client apps that use the authorization code flow. The use of PKCE by such clients is optional. If used, Client apps must use only the `S256` code challenge method. Please refer to RFC 7636 for details on how to use PKCE in requests to the authorization and token endpoints.

All requests to the API must include the access token transmitted in the Authorization header of the HTTP request as a bearer token as illustrated in RFC 6750. If the access token is missing, expired, or otherwise not valid for the requested operation, the API will return a 401 Unauthorized or 403 Forbidden response.

3. API Details

a. Query a Specific Data Category Resource

Client software must be capable of making HTTPS RESTful requests in accordance with the FHIR specification and consuming the FHIR Resources required by the United States Core Data for Interoperability (USCDI) or the FHIR Bulk Data Access specification.

Resources returned conform to the resource profiles for HL7 FHIR R4 defined in the HL7 FHIR US Core Implementation Guide Release 3.1.1 found at https://hl7.org/fhir/us/core/STU3.1.1. All of the US Core Resource Profiles are supported **except** the optional US Core Medication Profile, as this API uses inline medication codes to represent medications instead of References to separate Medication resources.

General specifications for FHIR resources and the associated data elements can be found at https://hl7.org/fhir/R4/resourcelist.html for other resource types without a US Core profile. Additional information for each required FHIR Resource or Data Element used to represent each data category can be found at the page above. For example, the Patient.name element is found at: https://hl7.org/fhir/R4/patient-definitions.html#Patient.name. The US Core FHIR IG profiles found at https://hl7.org/fhir/us/core/STU3.1.1/profiles.html provide additional information on the meanings of elements and codes used by these profiles.

Choices of data types: The US Core resource profiles permit a number of data types for certain resource elements as indicated by the [x] notation in the corresponding element names. The following table lists the data types supported for elements where a choice of data types is allowed by the US Core FHIR IG. However, please note that a particular Data Holder may not return all supported data types for a given element.

RESOURCE DATA ELEMENT	SUPPORTED CHOICES
AllergyIntolerance.onset[x]	dateTime, Age, Period, Range, string
CarePlan.scheduled[x]	Timing, Period, string
CarePlan.product[x]	CodeableConcept, Reference
Condition.onset[x]	dateTime, Age, Period, Range, string
Condition.abatement[x]	dateTime, Age, Period, Range, string
DiagnosticReport.effective[x]	dateTime, Period
Goal.start[x]	date, CodeableConcept
Goal.detail[x]	Quantity, Range, CodeableConcept, string, boolean, integer, Ratio
Goal.due[x]	date
Immunization.occurrence[x]	dateTime, string
Immunization.doseNumber[x]	positiveInt, string
Immunization.seriesDoses[x]	positiveInt, string
MedicationRequest.reported[x]	boolean, Reference
MedicationRequest.medication[x]	CodeableConcept
MedicationRequest.doseageInstruction.asNeeded[x]	boolean, CodeableConcept
MedicationRequest.doseageInstruction.doseAndRate.dose[x]	Range, SimpleQuantity
MedicationRequest.doseageInstruction.doseAndRate.rate[x]	Ratio, Range, SimpleQuantity
MedicationRequest.doseageInstruction.substitution.allowed[x]	boolean, CodeableConcept
Observation.effective[x]	dateTime, Period
Observation.value[x]	Quantity, CodeableConcept, string, boolean, integer, Range, Ratio, S
Observation.component.value[x]	Quantity, CodeableConcept, string, boolean, integer, Range, Ratio, S
Patient.deceased[x]	boolean, dateTime
Patient.multipleBirth[x]	boolean, integer
PracitionerRole.performed[x]	dateTime, Period
Provenance.ocurred[x]	dateTime, Period

Choices of Reference resource types: The US Core resource profiles and the inherited FHIR R4 Observation Vital Signs profile permit a number of resource types to be referenced for certain "Must Support" resource elements where a value of type Reference is allowed for the corresponding element. The following table lists the resource types supported for such elements where a choice of referenced resource types is allowed. However, please note that a particular Data Holder may not return all supported resource types for a given element.

RESOURCE DATA ELEMENT	SUPPORTED CHOICES
CareTeam.participant.member	US Core Patient, US Core Practitioner, US Core Organization
DiagnosticReport.performer	US Core Practitioner, US Core Organization
DocumentReference.author	US Core Practitioner, US Core Organization, US Core Patient

MedicationRequest.reportedReference	US Core Patient, US Core Practitioner, US Core Organization
MedicationRequest.requester	US Core Practitioner, US Core Organization, US Core Patient
Observation.hasMember	FHIR R4 Observation Vital Signs
Provenance.agent.who	US Core Practitioner, US Core Patient, US Core Organization

b. Patient Selection

To search for patients, the application should request a bundle of Patient resources matching suitable search criteria. To facilitate this, client apps may include the following optional search parameters in a request for a Patient resource:

RESOURCE DATA ELEMENT	SEARCH PARAMETER
Patient.identifier	identifier
Patient.name	name
Patient.gender	gender
Patient.birthDate	birthdate

Please see the sections relating to search parameters in the US Core Patient Profile for details. Some Data Holders may support additional search parameters. Contact the Data Holder for more information on the supported search parameters or review the capability statement.
For example, to retrieve a bundle of Patient resources to which the app is authorized, where the patient's first or last name is Smith and the patient was born on July 4, 1976, the request could be formatted as:

```
https://[baseURL]/Patient?name=Smith&birthdate=1976-07-04
```

The API will return a bundle of all patients (possibly zero) matching the search criteria. Only patients for which the app has been authorized access will be included in the results. Each patient returned in the search results is assigned a unique patient ID that can be found in the Patient.id element of the corresponding Patient resource. This patient ID can be included in subsequent requests to retrieve additional resources for that specific patient.

c. Query for a Specific Data Category

The Patient resource can be retrieved by specifying a specific patient ID or by performing a search as discussed in section 3(b). The remaining resource types listed in Section 3(a) can be accessed for a specific patient as a Bundle of resources by performing a search by resource type or by patient compartment and specifying the patient ID in the request. For example, to retrieve a bundle of Immunization resources containing all available immunization history for Patient 1234, the request could be formatted as:

```
https://[baseURL]/Immunization?patient=1234
```

The following search terms can be used to isolate results for a single USCDI data class in cases where two or more data classes are represented by a single FHIR resource type:

USCDI DATA CLASS	SEARCH PARAMETER
Health Concerns	category=http://hl7.org/fhir/us/core/CodeSystem/condition-category|heal
Problems	category=http://terminology.hl7.org/CodeSystem/condition-category|prol
Smoking Status	code=http://loinc.org|72166-2

Vital Signs	category=http://terminology.hl7.org/CodeSystem/observation-category|v
Laboratory Tests & Laboratory Values/Results	category=http://terminology.hl7.org/CodeSystem/observation-category|la

For example, to retrieve the Smoking Status for Patient 1234, the request could be formatted as:

```
https://[baseURL]/Patient/1234/Observation?code=http://loinc.org|72166-2
```

Each search request will return a bundle of zero or more results meeting the search criteria.

d. Query for All Data as a CCDA document

CCDA documents can be accessed within DocumentReference resources using the $docref operation as described in the US Core FHIR IG at: https://hl7.org/fhir/us/core/STU3.1.1/OperationDefinition-docref.html. CCDAs are categorized as "Summary of Episode" Notes with LOINC code 34133-9. For example, to request a CCDA document covering all dates for patient 1234, the query could be formatted as:

```
https://[baseURL]/DocumentReference/$docref?patient=1234
```

The Base64 encoded CCDA XML data can be found in the DocumentReference.content.attachment.data element of the returned DocumentReference resource within the returned Bundle, if any.

e. Query for a Subset of Data using Search Parameters

The USCDI data classes or CCDA documents returned by the API may be filtered by supplying one or more optional search parameters, as detailed in the search parameter sections of the US Core Profiles for the corresponding resource type. The following table lists the supported search parameters for this purpose:

RESOURCE DATA ELEMENT	SEARCH PARAMETER
CarePlan.category	category
CareTeam.status	status
DiagnosticReport.code	code
DiagnosticReport.category	category
DiagnosticReport.effective	date
DocumentReference.type	type
DocumentReference.category	category
DocumentReference.date	date
Condition.category	category
MedicationStatement.status	status
MedicationRequest.intent	intent
Observation.code	code
Observation.category	category
Observation.effective	date
Patient.identifier	identifier
Patient.name	name
Patient.gender	gender

Patient.birthDate	birthdate
Condition.category	category
Procedure.performed	date

Some Data Holders may support additional search parameters. Contact the Data Holder for more information on the supported search parameters or review the capability statement.

f. Error Handling

If the access token used in the request is invalid, expired, or the user has not been authorized to access the requested Resource, the API will return a 401 Unauthorized or 403 Forbidden HTTP response.

If the request cannot be processed for other reasons (temporarily unavailable, unsupported resource type, system error, etc.), the API will return a 400 Bad Request HTTP response containing an OperationOutcome Resource with additional information regarding the issue contained in the OperationOutcome.issue element.

Handling of OAuth-related errors is detailed in RFC 6749.
Handling of dynamic client registration errors is detailed in RFC 7591.

g. Supported clinical scopes

Apps identify the types of clinical data they would like to access by specifying one or more clinical OAuth scopes in their registration and authorization requests. Clinical scopes follow the format defined in the SMART App Launch Framework 1.0.0. Please see https://www.hl7.org/fhir/smart-app-launch/1.0.0/scopes-and-launch-context/index.html#scopes-for-requesting-clinical-data for additional details. Only read scopes are supported. Write scopes will be rejected. Applications may request patient-level, user-level, or system-level scopes for the following resource types:

RESOURCE
AllergyIntolerance
CarePlan
CareTeam
Condition
Device
DiagnosticReport
DocumentReference
Encounter
Goal
Group
Immunization
Location
MedicationRequest
Observation
Organization
Patient

Practitioner
PractitionerRole
Procedure
Provenance

Some Data Holders may support additional resource types and clinical scopes. Contact the Data Holder for more information on the supported scopes or review the endpoint's capability statement to identify other resource types supported by that endpoint.

Applications may also request "wildcard" scopes, e.g.: `user/*.read`

h. OpenID Connect

For client apps that need to authenticate the end user using OpenID Connect, the server supports the required OpenID Connect functionality as specified in the SMART App Launch Framework version 1.0.0. Please refer to the following link for details: https://www.hl7.org/fhir/smart-app-launch/1.0.0/scopes-and-launch-context/index.html#scopes-for-requesting-identity-data

i. Other Security Considerations

Revoking access to patient data: End users can sign in to the management endpoint listed above to terminate access that the user has previously granted to a client application. Client applications may request revocation of tokens by making a request to the Data Holder's revocation endpoint as described in RFC 7009.

j. Attributes required for registration

To register a client for authorization code flow, the following information is required:

Developer Name	the name of the entity that created the app
Developer Representative	the name of the person completing this registration
Client App Name	the human-readable name of the app
Redirect URI	the app's redirection URI (https only)
Client App URI (optional)	a web page about the client app (http or https)
Logo URI (optional)	a URI to retrieve an image file with the app's logo (https only)
Terms of Service URI (optional)	a web page to view the app's terms of service (http or https)
Privacy Policy URI (optional)	a web page to view the app's privacy policy (http or https)
Contact email	an email address where the app developer may be contacted (use mailto: scheme)

Additionally, the client app developer will select the applicable characteristics of their application from the following:

- The client application is a confidential client capable of storing a client secret.
- The client application is a native application capable of securing a refresh token.

Please see Section 2.1 of RFC 6749 for the definitions of 'confidential' and 'native application' as used above.