CrowdHEALTH

**Collective Wisdom Driving Public Health Policies**

# D3.12 Advanced Interoperability Techniques: Software Prototype v1

## Project Deliverable

This project has received funding from the European Union's Horizon 2020 Programme (H2020-SC1-2016-CNECT) under Grant Agreement No. 727560

**D3.12 Advanced Interoperability Techniques: Software Prototype v1**

| | | |
|---|---|---|
| Work Package: | | WP3 |
| Due Date: | | 31/12/2017 |
| Submission Date: | | 11/07/2018 |
| Start Date of Project: | | 01/03/2017 |
| Duration of Project: | | 36 Months |
| Partner Responsible of Deliverable: | | ATOS |
| Version: | | 1.0 |
| Status: | ☒ Final ☐ Draft ☐ Ready for internal Review<br>☐ Task Leader Accepted ☐ WP leader accepted<br>☒ Project Coordinator accepted | |
| Author name(s): | Carlos Cavero (ATOS) | Santiago Aso (ATOS) |
| Reviewer(s): | Mitja Luštrek (JSI) | George Moldovan (SIEMENS) |
| Nature: | ☐ R – Report ☒ D – Demonstrator | |
| Dissemination level: | ☒ PU – Public<br>☐ CO – Confidential<br>☐ RE – Restricted | |

| REVISION HISTORY | | | |
|---|---|---|---|
| Version | Date | Author(s) | Changes made |
| 0.1 | 10/05/2018 | Carlos Cavero (ATOS) | Draft – TOC |
| 0.2 | 16/05/2018 | Usman Wajid (ICE) | Data Converter sections |
| 0.3 | 18/05/2018 | Dimitris Miltiadou (SINGULAR), Thanos Kiourtis (UPRC) | Comments about Rule Engine and the integration into the Data Converter |
| 0.3 | 23/05/2018 | Santiago Aso (ATOS) | Terminology service sections |
| 0.4 | 29/05/2018 | Carlos Cavero (ATOS) | Executive Summary and Conclusions |
| 0.5 | 06/06/2018 | Carlos Cavero (ATOS) | First readable version of the document |
| 0.6 | 20/06/2018 | Quality assurance team (ATOS) | Internal quality assurance |
| 0.7 | 25/06/2018 | Carlos Cavero (ATOS) | First version ready for review |
| 0.8 | 06/07/2018 | Mitja Luštrek (JSI) | First internal review |
| 0.9 | 10/07/2018 | George Moldovan (SIEMENS) | Second internal review |
| 1.0 | 11/07/2018 | Carlos Cavero (ATOS) | Final version ready for submission to the EC and taking into account reviewers comments. |

## List of acronyms

| | |
|---|---|
| API | Application Programming Interface |
| FHIR | HL7 Fast Healthcare Interoperability Resources |
| HHR | Holistic Health Record |
| HTTP | Hypertext Transfer Protocol |
| JSON | JavaScript Object Notation |
| MVP | Minimum Viable Product |
| REST | REpresentational State Transfer |
| XML | Extensible Markup Language |
| WAR | Web ARchive |

## Contents

## List of figures



## List of tables

## Executive Summary

Software prototypes allow providing Minimum Viable Products (MVPs) at the beginning of the project lifetime. With the fast prototyping methodologies even if the components are not fully implemented and the development stage is not finished it is possible to receive early feedback and acquiring valuable feedback from stakeholder partners, validating earlier our assumptions and approaches. Another advantage of this approach is to start the integration stage early in the project easing the business roadmap transition with the possibility to start the exploitation phase before the product is finished thanks to the delivery of MVPs.

This deliverable D3.12 describes the prototype of the advanced interoperability techniques under T3.3 in charge of transforming and mapping data represented under different standards notations and terminologies.

The prototype is based on the efforts and specifications detailed in D3.9 [1] and it is composed by two modules: **Data Converter** and **Terminology Server**. A third module described in D3.9, the **Rule Engine**, has been integrated including the transformation rules into the **Data converter**. The prototype converts data from the origin sources into the Holistic Health Record (HHR) and from HHR to HL7 FHIR using the mapping terminologies already stored.

# 1. Introduction

This deliverable is based on the design decision coming from D3.9 [1]. During the process of releasing this first prototype **Rule Engine** has been integrated into Data converter and therefore it is not included in the prototype description. The deliverable is structured as follows:

- Section 2 describes the prototype covering the main components, the interfaces and the technologies and tools used:
  - **Data Converter** stores mapping rules to transform from a given health information structure to another (from raw XML to HHR, from HHR to FHIR). It converts datasets based on this mapping rules;
  - **Terminology Service** provides different semantic operations with the available terminologies, like the retrieval of concepts, semantic subsumption or translation between terminologies.
  - **Rule engine** as was a part of the initial architecture and is described in deliverable D3.9. However, as the project evolved, the Rule Engine component became obsolete and its functionalities, as described within the context of D3.9, were implemented by the Data Converter tool and are now a part of the existing WP3 flow such as the Raw to HHR rules.
- Section 3 provides the needed information about availability and exploitation of the source code of the aforementioned modules.
- Section 4 describes the conclusions about the current prototype and the future work

# 2. Prototype overview

## 2.1. Main components of the prototype

As it was described in D3.9 [1] the main components are the Data Converter and the Terminology Service.

### 2.1.1. Data Converter

The first prototype of the Data Converter component is composed of two services:

- **HHR Conversion**: The HHR Conversion service performs the specific operation of transforming incoming data (in XML) to the HHR format. The workflow followed by this service is depicted in Figure 1. The first entity in the figure (cra_patient_in) accepts an incoming dataset (in XML) from the Gateway. The incoming data is passed to the tMap_1 component that is responsible for actual transformation of incoming data. The outcome of the transformation operation is then pushed to hhr_patient, which is responsible for sending the data back to the Gateway.

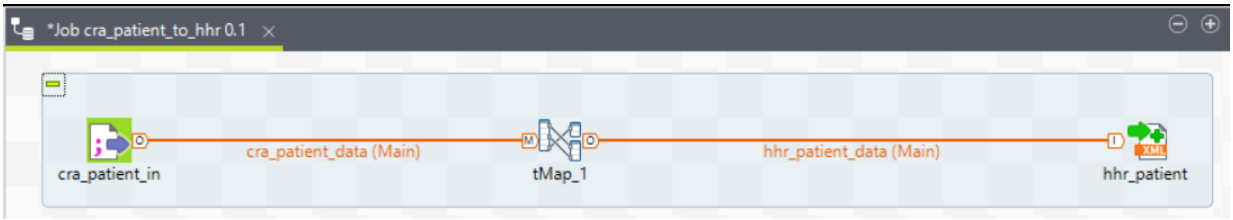



*Figure 1: Internal data flow within HHR Conversion Service*

In this first prototype of the HHR Conversion service, different instances of the service are developed, and exposed as Docker containers (see 2.3 Baseline technologies and tools for further information), to process different types of data coming from the Gateway component. For example, the instance “Cra_comorbidities_to_hhr_0.1” is capable of converting the Comorbidities data of patients from the CRA use-case.

- **FHIR Conversion:** The FHIR conversion service receives an input the data in HHR format and applies transformation rules to produce the data in FHIR format. The FHIR Conversion service is provided as a Dockerized RESTful Web Service. It exposes a REST interface where other components, like the Gateway, can POST HHR documents to be converted to FHIR documents and sent back as a response of the REST call. Its implementation is based on Spring Web MVC, making use of Apache Maven as building tool, which compiles the source code, runs a set of JUnit tests to ensure the integrity, and finally packs the component as a WAR file than can be later deployed on an Apache Tomcat server. A Dockerfile and a set of shell scripts are provided to automate the building, testing, deployment and publishing steps. The project integrates the HHR Java Model developed in task T3.1, but due to the

limitations of the HHR Model, some aspects of the use-case dataset cannot be converted into FHIR in this time. FHIR Conversion service also integrates HAPI FHIR, a Java implementation of HL7 FHIR, to aim for a reliable conversion to the target document schema.

#### 2.1.2. Terminology Service

As explained in the Deliverable D3.9 [1], the terminology service is intended to fulfil the required semantic operations over the terminologies used for the codification of clinical data within the CrowdHEALTH platform. From the already defined (D3.9 [1]) set of operations, the current prototype covers the following operations:

- Concept lookup/decomposition, return additional detail information about a particular system/code.
- Subsume testing.
- Concept translation.

To determine the prioritization of development activities, the current needs of the CrowdHEALTH platform have been considered. Given that there is currently little use in the creation of 'Value Sets', it has been considered convenient to postpone the development of the operations related to this set of concepts to the next iteration of the prototype.

### 2.2. Interfaces

#### 2.2.1. Data Converter

The interfaces exposed by the two services, composing the Data Converter component are described below:

- **HHR Conversion:** the interfaces exposed by the HHR Conversion service are named following the different types of use-case data they process. The naming convention used for the interfaces is:

  "*use-case acronym*"_"*data type*"_"to"_"hhr"_"*version*"

  Based on the above naming convention, the following interfaces are currently exposed by different instances of HHR Conversion services. These instances of the HHR Conversion service are able to process different types of data being exposed by the CRA use-case in CrowdHEALTH.

  - **Name of HHR Conversion service:** Cra_behaviour_to_hhr_0.1
  - **Service Interface:** POST http://icemain.hopto.org:7025/rawtohhr/cra/behaviour

  Similarly,
  - Cra_comorbidities_to_hhr_0.1: icemain.hopto.org: 7025/rawtohhr/cra/comorbidities
  - Cra_diagnosis_to_hhr_0.1: icemain.hopto.org: 7025/rawtohhr/cra/diagnosis
  - Cra_patient_to_hhr_0.1:

icemain.hopto.org: 7025/rawtohhr/cra/patient
- Cra_sideeffects_to_hhr_0.1: icemain.hopto.org: 7025/rawtohhr/cra/sideeffect
- Cra_treatment_to_hhr_0.1: icemain.hopto.org: 7025/rawtohhr/cra/treatment

Other instances of the HHR Conversion service, capable of dealing with data from other use-cases will be developed in future. Also, a complete list of services in XML can be found in:

http://icemain.hopto.org:7025/servicediscovery

- **FHIR Conversion:** The FHIR Conversion service exposes a REST interface to receive data from the Gateway. A test instance of the service can be accessed at this REST endpoint:

http://icemain.hopto.org:7030/api/convert

This service is currently able to convert HHR data from the CRA use-case (as exposed by the HHR Converter service) into FHIR format.

### 2.2.2. Terminology Service

The terminology service has been implemented following the HL7 FHIR specification. The interfaces provided for the first prototype are the following:[1]

| Interface | |
|---|---|
| Description | Lookup (GET): return additional detail information about a particular system/code[2] |
| Endpoint URL | https://terminology-service.atosresearch.eu/CodeSystem/$lookup |
| HTTP method | GET |
| Parameters | System<br>• http://snomed.info/sct<br>• http://loinc.org<br>• http://hl7.org/fhir/sid/icd-9-cm<br>• http://hl7.org/fhir/sid/icd-10-cm<br>• http://www.whocc.no/atc<br>String: code |
| Request Header/Body | Header:<br>Accept: application/xml<br>Or<br>Accept: application/json<br><br>Body:<br>N/A |

1 https://www.hl7.org/fhir/terminology-service.html

2 https://www.hl7.org/fhir/codesystem-operations.html#lookup

| Interface | |
|---|---|
| Response Body | Serialization of the FHIR resource Parameters[3] in XML or JSON with the outcome of the operation<br>The parameters contain:<br>• Display name of the concept<br>• System and version of the terminology<br>• Relationships to other concepts<br>• A set of parameters that depends on the terminology |

*Table 1- Lookup GET Endpoint*

| Interface | |
|---|---|
| Description | Lookup (POST): return the code system using a Coding object |
| Endpoint URL | https://terminology-service.atosresearch.eu/CodeSystem/$lookup |
| HTTP method | POST |
| Parameters | Application/xml or Application/json: parametersString<br>• Serialization of the FHIR resource Parameters with a value with a Coding[4] FHIR datatype |
| Request Header/Body | Header:<br>Accept: application/xml<br>Or<br>Accept: application/json<br><br>Body:<br><?xml version="1.0" encoding="UTF-8"?><br><Parameters><br><parameter><br><name value="coding"/><br><valueCoding><br><system value="http://snomed.info/sct"/><br><code value="235856003"/><br></valueCoding><br></parameter><br></Parameters> |
| Response Body | Serialization of the FHIR resource Parameters[5] in xml or json with the outcome of the operation<br>The parameters contain:<br>• Display name of the concept<br>• System and version of the terminology<br>• Relationships to other concepts<br>A set of parameters that depends on the terminology |

*Table 2- Lookup POST Endpoint*

| Interface | |
|---|---|
| Description | Subsumes (GET): return the relationship between codeA and codeB[6] |
| Endpoint URL | https://terminology-service.atosresearch.eu/CodeSystem/$subsumes |

3 http://hl7.org/fhir/parameters.html
4 http://hl7.org/fhir/datatypes.html#coding
5 http://hl7.org/fhir/parameters.html
6 https://www.hl7.org/fhir/codesystem.html#subsumption

| Interface | |
|---|---|
| HTTP method | GET |
| Parameters | String: system<br>• http://snomed.info/sct<br>• http://loinc.org<br>• http://hl7.org/fhir/sid/icd-9-cm<br>• http://hl7.org/fhir/sid/icd-10-cm<br>• http://www.whocc.no/atc<br>String: codeA<br>String: codeB |
| Request Header/Body | Header:<br>Accept: application/xml<br>Or<br>Accept: application/json<br><br>Body:<br>N/A |
| Response Body | Serialization of the FHIR resource Parameters in xml or json with the outcome of the operation<br>• Subsumes<br>• Subsumes-by<br>• Equivalent<br>• Not-subsumed |

*Table 3- Subsumes GET Endpoint*

| Interface | |
|---|---|
| Description | Subsumes (POST): return the relationship between two codes given a Coding object. |
| Endpoint URL | https://terminology-service.atosresearch.eu/CodeSystem/$subsumes |
| HTTP method | POST |
| Parameters | Application/xml or Application/json: parametersString<br>• Serialization of the FHIR resource Parameters with:<br>○ a system value with a Uri[7] FHIR datatype<br>○ a codeA value with a Coding FHIR datatype<br>○ a codeB value with a Coding FHIR datatype |
| Request Header/Body | Header:<br>Accept: application/xml<br>Or<br>Accept: application/json<br><br>Body:<br><?xml version="1.0" encoding="UTF-8"?><br><Parameters xmlns="http://hl7.org/fhir"><br><parameter><br><name value="system"/><br><valueUri value="http://snomed.info/sct"/><br></parameter><br><parameter><br><name value="codingA"/> |

7 http://hl7.org/fhir/datatypes.html#uri

| Interface | |
|---|---|
| | <valueCoding><br><system value="http://snomed.info/sct"/><br><code value="235856003"/><br></valueCoding><br></parameter><br><parameter><br><name value="codingB"/><br><valueCoding><br><system value="http://snomed.info/sct"/><br><code value="3738000"/><br></valueCoding><br></parameter><br></Parameters> |
| Response Body | Serialization of the FHIR resource Parameters in xml or json with the outcome of the operation<br>• Subsumes<br>• Subsumes-by<br>• Equivalent<br>• Not-subsumed |

*Table 4- Subsumes POST Endpoint*

| Interface | |
|---|---|
| Description | Translate (GET): translate a concept from one value set to another code system[8] |
| Endpoint URL | https://terminology-service.atosresearch.eu/ConceptMap/$translate |
| HTTP method | GET |
| Parameters | String: system<br>• http://snomed.info/sct<br>• http://loinc.org<br>• http://hl7.org/fhir/sid/icd-9-cm<br>• http://hl7.org/fhir/sid/icd-10-cm<br>• http://www.whocc.no/atc<br>String: code<br>String: source. Source valueset context url for the translation<br>String: target. Target valueset context url for the translation<br>String: target system<br>• http://snomed.info/sct<br>• http://loinc.org<br>• http://hl7.org/fhir/sid/icd-9-cm<br>• http://hl7.org/fhir/sid/icd-10-cm<br>• http://www.whocc.no/atc |
| Request Header/Body | Header:<br>Accept: application/xml<br>Or<br>Accept: application/json<br>Body:<br>N/A |

8 https://www.hl7.org/fhir/conceptmap-operations.html#translate

| Interface | |
|---|---|
| Response Body | Serialization of the FHIR resource Parameters in xml or json with the outcome of the operation<br><br>A match in a translation is found with:<br>• An equivalence<br>  • wider<br>  • equivalent<br>  • specializes<br>  • inexact<br>• The code in the target system |

*Table 5- Translate GET Endpoint*

| Interface | |
|---|---|
| Description | Translate (POST): translate the concept to another code system given a Coding object. |
| Endpoint URL | https://terminology-service.atosresearch.eu/ConceptMap/$translate |
| HTTP method | POST |
| Parameters | Application/xml or Application/json: parametersString<br>• Serialization of the FHIR resource Parameters with:<br>  o a code value with a Coding FHIR datatype<br>  o a targetsystem value with a Uri FHIR datatype |
| Request Header/Body | Header:<br>Accept: application/xml<br>Or<br>Accept: application/json<br><br>Body:<br><?xml version="1.0" encoding="UTF-8"?><br><Parameters><br><parameter><br><name value="coding"/><br><valueCoding><br><system value="http://snomed.info/sct"/><br><code value="235856003"/><br></valueCoding><br></parameter><br><parameter><br><name value="targetsystem"/><br><valueUri value="http://hl7.org/fhir/sid/icd-10-cm"/><br></parameter><br></Parameters> |
| Response Body | Serialization of the FHIR resource Parameters in xml or json with the outcome of the operation<br><br>A match in a translation is found with:<br>• A equivalence<br>  • wider<br>  • equivalent<br>  • specializes<br>  • inexact<br>• The code in the target system |

*Table 6- Translate POST Endpoint*

## 2.3. Baseline technologies and tools

### 2.3.1. Data Converter

The following baseline technologies and tools are used by the first prototype of the Data Converter component.

- **ICE Data Platform** (IDP) is built upon open-source technologies from Talend and Apache to facilitate the design and development of data transformation operations. IDP operates as a code generator, producing data-transformation scripts and underlying programs in Java. Its GUI gives access to a metadata repository and to a graphical designer. Users can design individual data transformation jobs using graphical components. Crucially it is possible to develop and export data transformation jobs as (a) standalone Docker containers that can be deployed as micro-service or Jar, or (b) Jar files that can be readily integrated in a work/data flow. In the first prototype of the Data Converter component, IDP is used to develop the HHR Conversion service.

- **FHIR HAPI[9] Server** is an open-source implementation of the FHIR specification in Java. HAPI provides a built-in mechanism for adding FHIR's RESTful Server capabilities to the FHIR Conversion service. The HAPI RESTful Server is Servlet based, so it makes it easy to deploy to any of the many compliant containers that exist e.g. Docker.

- **Spring Framework,** is an application framework [10] and inversion of control[11] container[12] for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE (Enterprise Edition) platform. The FHIR Conversion service uses the Spring Web model-view-controller (MVC) framework to dispatch requests to handlers that use configurable handler mappings to perform the transformation from HHR to FHIR format.

- **Docker** is an open source project that allows the deployment of applications or services inside containers, adding a layer of abstraction. CI, Version Control, Portability, Isolation and Security. Containers are virtual machines running an operating system and environment of the developers choice, working this way allows for continuous integration and version control with the ability to quickly deploy and roll back versions, portability as docker is widely available on a number of host operating systems, the ability to isolate sub components within their own containers leading to

9 http://hapifhir.io/
10 https://en.wikipedia.org/wiki/Application_framework
11 https://en.wikipedia.org/wiki/Inversion_of_control
12 https://en.wikipedia.org/wiki/Servlet_container

well defined interfaces and also enhancing system security. These features allow a stable deployment environment Data Converter services.

### 2.3.2. Terminology Service

The Terminology Service is a set of REST services as described in the HL7 FHIR specification. The base technologies used for the implementation are mainly open source tools:

- **Spring Framework:** as stated in the IDP section the component is developed upon Spring Framework that eases the deployment of REST services including Swagger description.

- **HAPI FHIR libraries:** has been used mainly for the serialization and deserialization of FHIR resources.

- **Terminologies data sources:** in order to provide availability for the concepts management, data sources have been used to import them into MySQL database.

# 3. Source code

## 3.1. Availability

### 3.1.1. Data Converter

The source code of the Data Converter services is available for use and integration in the CrowdHEALTH platform.

- **HHR Conversion Service:** CrowdHEALTH GIT repository**:** git@crowdhealthtasks.ddns.net:CrowdHEALTH/DataConverterRaw.git
- **FHIR Conversion Service:** CrowdHEALTH GIT repository: git@crowdhealthtasks.ddns.net/CrowdHEALTH/DataConverter.git

### 3.1.2. Terminology Service

The source code is available at ATOS GitLab that will be uploaded soon into GitHub code repository. The strategy is to open source the project and offer to the HAPI community in order to get closely involved in the implementation, refinement and exploitation.

https://gitlab.atosresearch.eu/ari/terminology-service

## 3.2. Exploitation

### 3.2.1. Data Converter

The Data Converter services are currently deployed on partner resources and can be accessed through the specified address.

- **HHR Conversion Service:** With Postman or any other general REST client, it is possible to use the HHR Conversion service by posting an XML document. As an example scenario, the incoming data shown in Figure 2 (below) is processed by the "Cra_comorbidities_to_hhr_0.1" service (running at http://icemain.hopto.org:7025/rawtohhr/cra/comorbidities) to produce the HHR document shown in Figure 3.

```
<data>
    <item>
        <id>1</id>
        <userId>3</userId>
        <value>Cholesterol</value>
    </item>
    <item>
        <id>2</id>
        <userId>4</userId>
        <value>IBS</value>
    </item>
</data>
```

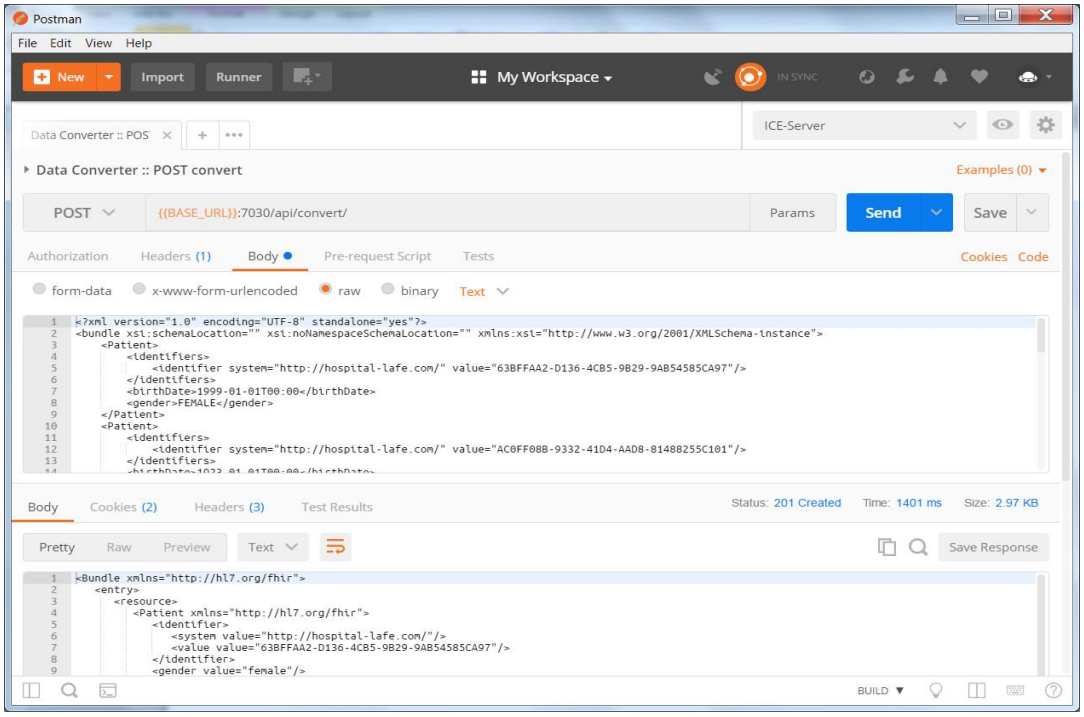
*Figure 2: Incoming document from CRA use-case – input to HHR Conversion service*

```
<?xml version="1.0" encoding="UTF-8"?>

<bundle>
  <Condition>
    <identifiers>
      <identifier system="https://www.careacross.com/comorbidities" value="1"/>
    </identifiers>
    <recorder xsi:type="Patient">
      <identifiers>
        <identifier system="https://www.careacross.com/patient" value="3"/>
      </identifiers>
    </recorder>
    <subject>
      <identifiers>
        <identifier system="https://www.careacross.com/patient" value="3"/>
      </identifiers>
    </subject>
    <performers>
      <performer xsi:type="Patient">
        <identifiers>
          <identifier system="https://www.careacross.com/patient" value="3"/>
        </identifiers>
      </performer>
    </performers>
    <asserter xsi:type="Patient">
      <identifiers>
        <identifier system="https://www.careacross.com/patient" value="3"/>
      </identifiers>
    </asserter>
```

*Figure 3: HHR formatted document - output of HHR Conversion service*

- **FHIR Conversion Service:** With Postman or any other general REST client, it is possible to use the FHIR Conversion service (running at http://icemain.hopto.org:7030/api/convert) by posting an HHR XML (as shown in Figure 4). This service generates a response 201 CREATED HTTP Response with a FHIR representation of the same data attached, or a 200 OK HTTP Response, in the case that the format is not recognisable (not an XML document, or not HHR), with the same data attached as it was posted.



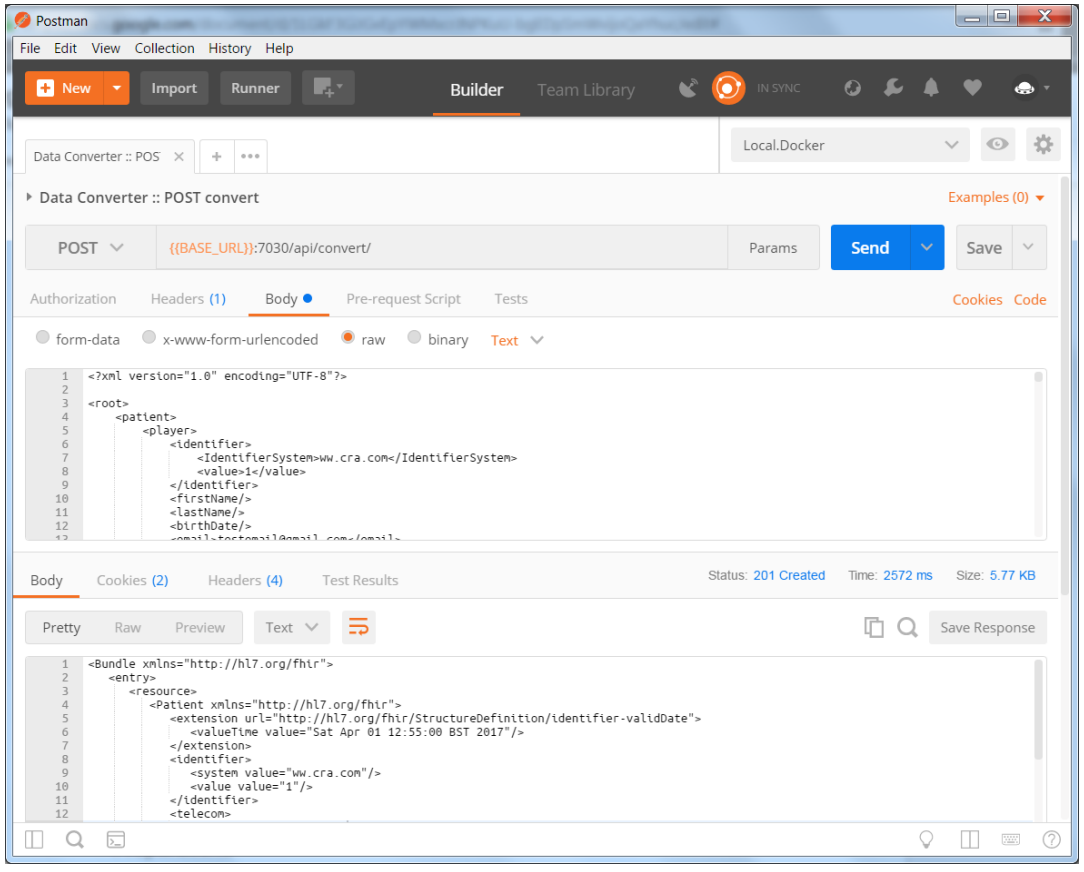




*Figure 4: Snapshot of testing FHIR Conversion service through Postman tool*

### 3.2.2. Terminology Service

Swagger[13] has been used to describe the APIs following the specification-based standards. This way users can test the different functionalities through the Swagger Graphical User Interface. Figure 5 shows the Swagger description including all the APIs available at:

https://terminology-service.atosresearch.eu/swagger-ui.html

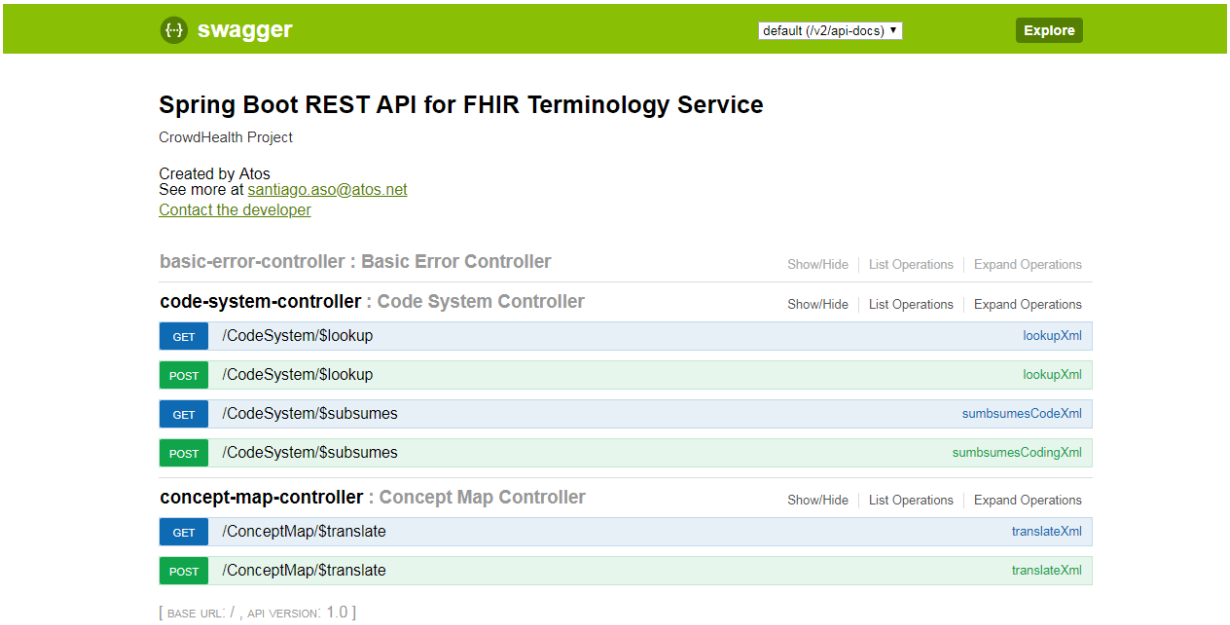



*Figure 5: Terminology Service swagger description*

13 https://swagger.io/

## 4. Conclusions

This deliverable described the first prototype of the advanced interoperability techniques. The early prototyping allows us to receive feedback from users early in the project. Two different components have been explained including the interfaces, the base technologies, where they are available and how to exploit them.

The prototype is built upon open source technologies being developed by solid communities. Next releases will include new functionalities that will be explained in forthcoming deliverables such as:

- Value Set expansion.
- Value Set validation.
- Closure table maintenance.
- Batch format of all operations.

Part of the component will also be open source and offered to the community to take advantage of the work force available in the community to enhance the project.

## 5. References

[1] D3.9 - Advanced Interoperability Techniques: Design and Open Specification