Dipartimento di Ingegneria e Scienza dell'Informazione

– KnowDive Group –

Electronic healthcare record integration

Document Data:	Reference Persons:
March 24, 2022	Aleksa Krsmanović

Index:

1 Purpose and project's resources

1.1 Project's purpose

The aim of the project is to build a knowledge graph (KG) concerning healthcare heterogeneous data about patients, professionals, facilities and other entities from European countries with different healthcare systems. This result could be the foundation to create a service which help the citizens to handle their own healthcare data, in a context in which the citizens moves around Europe. This paradigm is applied in the **Electronic Health Record (EHR)** is the instrument through which citizens can trace and consult the entire history of their health care life in a digital format, sharing it with healthcare professionals to ensure a more effective and efficient service. This great tool can still be improved in aspects as portability due to the different healthcare systems between countries, which we will try to address with our KG.

1.2 Knowledge resources - Teleologies

To pursue the optimal building of the final entity type graph (ETG) we have to follow the iTelos principles and methodologies. The first criterion to mind is that the integration must be purpose driven. The purpose become meaningful if we consider a certain portion of reality, the **domain of interest (DoI)**, which is possible to define thanks to knowledge resources. These allow the representation of purpose specific DoI by using high shareable resource. For this project we referred to three knowledge resources that contain useful vocabs to understand and model in a efficient way a graph regarding this field of knowledge.

- **FHIR:** https://hl7.org/FHIR/. It is the first resource utilized, it provides a detailed classification, ontology, and description for entities of our interest. FHIR is a standard for health care data exchange, a platform specification that defines a set of capabilities use across the healthcare process and in lots of different contexts. We retrieve the terminology necessary for integration, and other essential information, such as entities definition, scope and usage.
- **Schema.org:** https://schema.org/. The goal of this collaborative community effort is to build, maintain, and encourage structured data formats on the Internet. The schemas are a set of types arranged in a hierarchy, each associated with a set of properties. The vocabulary currently consists of 792 Types, 1447 Properties 15 Datatypes, 83 Enumerations and 445 Enumeration members. In the section "schemas" of this tool, among the commonly used types we found the voice "Health and medical types: notes on the health and medical types under MedicalEntity". With this resources we completed the list of types important for our domain of interest.
- **LOV:** https://lov.linkeddata.es/dataset/lov/ This repository of 760 vocabulary allowed us to select the tag "Health" from the 30 category tags available in the home page. There are 12 ontologies related to health and those are in English, Spanish and French. For each of them Ontology Overview, Description with a schema, Classes, Object Properties and Data Properties are available.

1.3 Data resources

Once defined the field of knowledge we are dealing with, we understand that we need medical data. In this project synthetically generated healthcare datasets were used. It is difficult to get access to Electronic Medical Records due to privacy concerns and technical burdens. The advantages of this choice are the possibility to bypass the step of anonymization – since we are not dealing with real subjects' privacy – and the ability to retrieve in a short time large amount of healthcare data. We selected three resources able to provide us a significant number of heterogeneous entities and properties to integrate to satisfy the purpose along the integration process.

1. **EMRBOTS:** http://www.emrbots.org. This dataset describes 10,000 patients, 36,143 admissions to the hospital, and 10,726,505 lab observations made on those patients. Despite the large number of instances, this dataset is quite poor in terms of number of properties describing each instance.

2. **Synthea:** https://github.com/synthetichealth/synthea. This dataset in total contains 15 data tables describing 15 different entities using various properties. In our oppinion, this dataset is the best described one in terms of the number of properties and consistency. The entities described are the following:

   - 1171 patient entities
   - 597 allergies of the patiens
   - 8376 medical conditions of the patients
   - 3483 care plans used by the patients
   - 78 medical devices implanted
   - 53346 medical encounters (appoitments)
   - 855 imaging examinations
   - 15478 immunizations (vaccinations)
   - 42989 medications prescribed to the patients
   - 299697 observations including both labaratory tests and first-hand observations
   - 1119 healthcare provides (hospitals)
   - 10 insurance companies paying for the medical procedures
   - 801 payments from insurance companies or patients to the hoispitals
   - 34981 procedures conducted on patiens
   - 855 doctors employed in various hospitals

3. **FHIR SMART data:** https://github.com/smart-on-fhir/sample-patients. This dataset describes 11 entities with various properties, however it is the smalles in terms of number of each propery. The entities described are the following:

   - 67 patients
   - 62 allergies of those patients

- 15 family history entities describing past
- 9 imaging examinations of the described patients
- 27 immunizations
- 3513 labs tests
- 361 medical prescriptions to the described patiens
- 561 problems occuring to the patients
- 22 procedures
- 49 social history entities of the described patients
- 1035 vitals function and measurements of the patients taken from the i2b2 demo data

If properties of the described datasets need to be examined in detail, each data table described is available in our GitHub repository in textual, tab/comma separated value format.

1.4 Metadata

Here we present some basic characteristic and properties of our datasets. These information aim to help user to achieve a better description of the problem, understanding of the data available, as well as the reson of generating synthetic medical records.

1. **EMRBOTS:**. The database was created by Uri Kartoun, PhD, in 2015 to allow non-commercial entities to use the artificial patient repositories to practice statistical analysis and machine-learning algorithms. The database contains the same characteristics that exist in a real medical database such as patients' admission details, demographics, socioeconomic details, medications, while relying only minimally on real patient data.
2. **Synthea:**. This synthetic data generation tool was developed by the MITRE corporation in 2016. It is an open-source Synthetic Patient Population Simulator. The goal is to output synthetic, realistic patient data and associated health records in a variety of formats. It is one of the most popular datasets in this domain and it is used for various purposes such as machine learning, deterministic predictive model tuning, and data integration projects.
3. **FHIR SMART data:**. This tool is provided by SMART corporation. This generator creates syntetic data about 67 patients, it uses a python script that select the data files in a directory to generate FHIR test patients.

1.5 Project domain of interest, personas and scenarios

In this subchapter we will discuss which are the problem to be solved by this knowledge graph developed using iTelos methodology. Moreover, examples of potential users will be given as well as use-case scenarios.

1.5.1 Project domain of interest

It is easy to realize how numerous inconveniences, additional costs and difficulties in life-or-death situations may arise from the fact that a system that allows the usage one's medical data abroad does not exist. These problems arise from the representation diversity in all levels. In this moment, a number of projects with the goal of integrating EHRs are run and transitioning to an integrated system will take years of effort. One of the most relevant ones worldwide, also in line with the goals of our project, is the eHealth Digital Service Infrastructure (eHDSI) of the European Union. We will use this project as a main reference and inspiration since the goals are shared.

Briefly speaking, eHDSI main goal is the creation of a European format that will allow electronic health records being shared in a secure manner at the same time as adhering to data protection rules. This allows citizens to access and transfer their medical data, as well as get necessary treatments or medication in member countries of the EU. It is, also, an essential requirement that the application level utilizing this konwledge graph needs to have multi-language support. Currently this project enforces two cross-border health services:

1. **ePrescription and eDispensation:** allows a transfer of person's electronic prescription from their country of residence in order to obtain medication in a pharmacy located in another EU country.

2. **Patient Summaries:** provides information on important health related aspects such as allergies, current medication, previous illness and surgeries. It is meant to be used by doctors of the health provider of another EU country.

3. **Medical images, lab results and hospital discharge reports:** these are planned to be integrated in the later stages of the project.

Our downstream analysis and implementation will focus on creating a knowledge and data resource in line with eHDSI which will integrate electronic medical data from different source datasets (representing different countries of the European Union) while being easy to exploit and expand. This improves the potential of treating any citizen of the EU, decreases medical expenses for both citizens and hospitals, as well as enhances academic research opportunities due to the resource availability. Additionally, the amount of data can be used for creating artificial intelligence systems that can aid in medical decision making.

1.5.2 Personas and Scenarios

In this subchapter we aim to give examples of different personas and hypothetical situations in which they can benefit from our solution. All five different types of users have different reasons for utilizing our resource.

1. Giorgio: An Italian 70 year old man with Crohn's disease. His family lives in the Poland and he visits them often. Due to his condition he needs to take anti-inflammatory medication

every day. He only speaks Italian.
Scenario: During his stay in Poland, he ran out of medication. This type of drug is obtainable only with a medical prescription. Instead of going to the doctor in Poland, or returning to Italy, Giorgio would benefit from accessing his prescription history. In these situations it is possible that this specific drug is not sold in Poland. In this case, a drug of equivalent effect may be sold to Giorgio upon presenting a proof of medical conditions.

2. **Miloš**: A Czech man suffering from Amyotrophic lateral sclerosis - a currently untreatable disease. He is being treated in The Czech Republic but is looking for a clinical trial of an experimental drug to enroll into. Neither him or his doctor do not not speak any other language.
Scenario: He was able to enroll into a clinical trail in Austria and now he visits Vienna once a week to receive an experimental therapy. The medical team from Vienna is not able to communicate properly to with his regular doctor. A way of transferring medical data back and forth between two institutions is needed.

3. **Sophie**: A French anesthesiologists. She is often in the situation where she has to make life-or-death decisions before a person with a serious condition is operated on.
Scenario: A Spanish unconscious man with critical injuries is brought into the hospital after a car crash and has to undergo surgery. In order to know what cocktail of drugs she needs to administer prior to the surgery, she must know the patients previous medical history.

4. **Francesco**: A researcher at the University of Trento with a PhD in life sciences. He is often in a need of large amounts of data to ensure his analysis has enough statistical power to yield valid results.
Scenario: To conduct his research project on the correlations of the dominant industries of a region with medical conditions of inhabitants of that region he needs data coming from many cities across Europe.

5. **Johanes**: A security worker at the Brussels airport in Belgium.
Scenario: Due to the current pandemic situation and the strict rules employed by the Belgium government, his job requires him to control vaccination certificates and PCR test results. Either one of those is required to enter the country. Different formats of certificates and test results from various countries make his job difficult, moreover fake documents are not rarely encountered. A single standard of reporting certificates and test results adopted by all EU countries would make this procedure significantly easier.

6. **Maria**: A Spanish government official involved in strategic planning of handling the COVID situation.
Scenario: To make the right decisions she needs to know day-to-day number of cases, recovery rates of patients, doctors available and their specialization for multiple geographical locations in Spain. She would also like to know which blood test is the most prognostic for a mild infection in adults.

7. **Cristina**: A world renowned practitioner of orthopedic medicine.

Scenario: In order to promote her new research work on spine injuries in children she would like to organize a scientific meeting in Barcelona and invite all practitioner of orthopedic medicine for that region. In order to invite them, she would like to know their home addresses so she can send them a paper invitation.

2 Inception

In this chapter we will describe the the first phase of the iTelos integration methodology - the Inception phase. We will start by introducing the competency questions - queries that the hypothetical users of our resource might request. Afterwards, a description of source datasets will be given, along it's classification into three main types of resources. Finally, we will evaluate the results obtained in this phase.

2.1 Definition of competency queries

To create the initial competency question, we will rely on the hypothetical situation that have been presented in the previous chapter. Every given scenario will be assessed in order to generate multiple queries needed for that specific situation.

Number	Person	Requirement	Returns
1.1	Giorgio	List medical prescriptions given patient ID [in Polish]	A list of current medical prescription given patient's ID. Return the results alongside the doctors who prescribed them, healthcare providers where the prescriptions were made and theirs locations.
1.2	Giorgio	List medical conditions given patient ID [in Polish]	A list of current medical conditions given patient's ID. Return the results alongside the doctors who diagnosed them, healthcare providers where the diagnosis were made and their locations.
2.1	Miloš	List medical conditions given patient ID [in German]	A list of medical conditions given patient's ID. Return the results alongside the doctors who diagnosed them, healthcare providers where the diagnosis were made and their locations.
2.2	Miloš	List patient allergies given patient ID [in German]	A list of allergies given patient's ID. Return the results alongside the doctors who diagnosed them, healthcare providers where the diagnosis were made and their locations.
2.3	Miloš	List family disease history given patient ID [in German]	A list of tuples (family member - disease) given a patients ID.
2.4	Miloš	List medical prescriptions given patient ID [in Czech]	A list of medical prescription given patient's ID. Return the results alongside the doctors who prescribed them, healthcare providers where the prescriptions were made and theirs locations.
2.5	Miloš	List medical laboratory results given patient ID [in Czech]	A list of laboratory results given a patient's ID. Return the results alongside the doctors who prescribed them, healthcare providers where the test was done and their location.
2.6	Miloš	List all medical appointments of a patient given his ID [in German]	List metadata of all medical encounters of a patient given his ID, including information on doctors and health providers who conducted these.
3.1	Sophie	Retrieve patient metadata given his/her ID [in French]	A list of non-medical patient attributes given his/her ID.

Number	Person	Query	Return values
3.2	Sophie	List patient allergies given patient ID [in French]	A list of allergies given patient's ID. Return the results alongside the information is the allergy ongoing.
3.3	Sophie	List medical prescriptions given patient ID [in French]	A list of current medical prescription given patient's ID. Return the results alongside the doctors who prescribed them, healthcare providers where the prescriptions were made and theirs locations.
3.4	Sophie	List medical conditions given patient ID [in French]	A list of current medical conditions given patient's ID. Return the results alongside the doctors who diagnosed them, healthcare providers where the diagnosis were made and their locations.
4.1	Francesco	List patient metadata in a defined region for a defined time period [in Italian]	A list of non-medical patient data of multiple patients at a defined location, for a defined time interval in Italian.
4.2	Francesco	List medical conditions of multiple users in a defined region for a defined time period [in Italian]	A list of medical condition of multiple patients at a defined location, for a defined time interval, alongside the doctor who diagnosed it, diagnosis date and the location of the healthcare provider where the diagnosis was made.
4.3	Francesco	List allergies of multiple users in a defined region for a defined time period [in Italian]	A list of allergies of multiple patients at a defined location, for a defined time interval. Return the results alongside the doctors who diagnosed them, healthcare providers where the diagnosis were made and their locations.
4.4	Francesco	List family disease history of multiple patients in a defined region, at a defined time period [in Italian]	A list of tuples (family member - disease) for multiple patients of a defined region and a defined time period.
5.1	Johanes	Retrieve patient metadata given his/her ID [in Belgium]	A list of non-medical patient attributes given his/her ID.
5.2	Johanes	List SARS-CoV-2 immunization confirmations given ID of a person [in Belgium]	A confirmation in that a person is vaccinated against SARS-CoV-2 and the date of vaccination. Return the results alongside the healthcare provider where the immunization is done.

Number	Person	Query	Return values
5.3	Johanes	List SARS-CoV-2 PCR test confirmation given ID of a person [in Belgium]	A confirmation in that a person is has tested negative for SARS-CoV-2 and the date of the testing. Return the results alongside the healthcare provider where this test was performed.
6.1	Maria	Count the number of COVID diagnosis being made for each hospital in Sevilla.	Return counts of cases diagnosed.
6.2	Maria	Find the average sickness time for COVID patients under Ivermectin therapy	Return the average.
6.3	Maria	Identify the hospital with the highest number of of pulmonology practicioners.	Returns the hospital alongside it's location.
6.4	Maria	List blood results for patients in age category 25 - 60 that have had COVID, but the disease did not last more than 5 days	Returns blood test result and it's metadata as well as the diagnosis metadata of those patients.
7.1	Cristina	List all house addresses of orthopedic doctors working in Barcelona	Returns a list of addresses along other doctor properties.

In the downstream analysis we will assume all translation is done at application level, independently of our solution. By analyzing these competency questions we then classified each concept involved as common, core or cotextual. The following table describes this:

Number	Core	Common	Contextual
1.1	Patient, Patient ID, Hospital, Doctor	Location	Encounter, Prescription, Prescription end date, Prescription start date, Prescription drug name, Prescription dosage description
1.2	Patient, Patient ID, Doctor, Hospital	Location	Encounter, Condition, Condition start date, Condition end date, Condition type
2.1	Patient, Patient ID, Hospital, Doctor	Location	Encounter, Condition, Condition start date, Condition end date, Condition type
2.2	Patient, Patient ID, Hospital, Doctor	Location	Allergy, Encounter, Allergy start date, Allergy end date, Allergy description
2.3	Patient, Patient ID		Family disease history
2.4	Patient, Patient ID, Hospital, Doctor	Location	Encounter, Prescription, Prescription end date, Prescription start date, Prescription drug name, Prescription dosage description
2.5	Patient, Patient ID, Doctor, Hospital	Location	Encounter, Lab result, Lab result type, Lab result date, Lab result objective
2.6	Patient, Doctor, Hospital	Location	Encounter, Encounter start time, Encounter end time, Encounter description
3.1	Patient, Patient ID, Patient age, Date of Birth, Patient address, Patient gender, Patient phone, Patient email		
3.2	Patient, Patient ID, Hospital, Doctor	Location	Allergy, Encounter, Allergy start date, Allergy end date, Allergy description
3.3	Patient, Patient ID, Hospital, Doctor	Location	Encounter, Prescription, Prescription end date, Prescription start date, Prescription drug name, Prescription dosage description

Number	Core	Common	Contextual
3.4	Patient, Patient ID, Doctor, Hospital	Location	Encounter, Condition, Condition start date, Condition end date, Condition type
4.1	Patient, Patient ID, Date of birth, Date of death	Location	
4.2	Patient, Patient ID, Date of birth, Date of death , Hospital	Location	Condition, Condition type
4.3	Patient, Patient ID, Date of birth, Date of death, Hospital, Doctor	Location	Allergy , Allergy description
4.4	Patient, Patient ID, Date of birth, Date of death	Location	Family disease history
5.1	Patient, Patient ID, Patient age, Date of Birth, Patient address		
5.2	Patient, Patient ID , Hospital	Location	Immunization, Immunization date, Immunization disease
5.3	Patient, Patient ID, Hospital	Location	Lab result, Lab result type, Lab result date, Lab result objective
6.1	Hospital	Location	Condition, Condition start date, Condition end date, Condition type
6.2	Patient		Condition, Condition start date, Condition end date, Condition type, Prescription drug name, Prescription dosage description, Prescription start date, Prescription end date
6.3	Hospital, Doctor	Location	Doctor specialization
6.4	Patient, Patient age		Lab result, Lab result, Lab result type, Lab result date, Lab result objective, Condition start date, Condition end date, Condition type
7.1	Doctor, Doctor address, Doctor name, Doctor contact , Hospital	Location	Doctor specialization

It should be noted that in the case of competency questions 5.x we are referring to a Person as a Patient because from the perspective of the system these are the same.

2.2 Resource collection

Data resources described in the previous chapter are fairly easy to access. We were able to retrieve the intended datasets in textual formats (tab or comma separated value formats), without a need of performing scraping.

- **EMRBOTS:** From the official website we collected .tsv files containing 10,000 patients, 36,143 admissions, and 10,726,505 lab observations.
- **Synthea:** From the official website we retrieved .csv files containing 1,000 patients, 53,347 admissions and 299,698 lab observations.
- **FIHR generated data:** Using the tool provided by SMART corporation we generated synthetic data of 67 patients, 561 clinical observations and 3,513 lab observations also in .csv format.

Due to this large size difference between retrieved datasets we will subsample data of 70 patients from the first two resources for sake of convenience and continue developing our knowledge graph with the reduced and uniform number of datapoints.

2.3 Resource classification

Considering the purpose which drives the creation of our solution, the data and knowledge resources collected we classify resources into one of 3 categories - core, common or contextual.

Core resources: This category includes resources related to the more fundamental features of DoI:

- Patient
- Patient ID
- Patient age
- Patient address
- Patient date of birth
- Patient date of death
- Patient gender
- Patient phone
- Patient email

- Doctor
- Doctor address
- Doctor name
- Doctor contact
- Hospital

Common resources: This category contains resources related to aspects that are shared by all domains.

- Location

Contextual resources: This class comprehend specific and particular information about DoI:

- Encounter
- Encounter start time
- Encounter end time
- Encounter description
- Condition
- Condition type
- Condition start date
- Condition end date
- Prescription
- Prescription drug name
- Prescription dosage description
- Prescription start date
- Prescription end date
- Allergy
- Allergy start date
- Allergy end date
- Allergy description
- Family disease history
- Lab result
- Lab result type

- Lab result date
- Lab result objective
- Immunization
- Immunization date
- Immunization disease
- Doctor specialization

2.4 Inception phase evaluation

We divided this section in two parts, as we consider that both of them are essential to provide a complete description of the project.
First, we made a general discussion on how we handle the instruction given and problems that arised by taking as a reference some iTelos principles, verifying that they are totally or partially respected and to which extent.
In the second place we evaluated the metrics defined in the evaluation theory comparing concepts and properties retrieved from the CQs with the ones present in the onthologies we selected.

2.4.1 Discusions on firsts itelos principles

- Since the first principle is Purpose Driven Integration we should ask ourselves if the data collected are coherent with the purpose. We collected three synthetically generated datasets containing heterogeneous medical data, which allow us to address fairly well our competency queries. Thus, we evaluate our data as suitable in the scope of creating a KG able to represent the structure of a service which help the citizens to handle their own healthcare data.
- Another important aspect of iTelos is the comprehensiveness of domain of interest composition. That means the representation of purpose specific DoI by composition of domains designed using high shareable resources. Considering the goal of integration, the number of sources found is in line with the given indications, even though we know that a greater number would be better. We didn't manage to find other datasets for two reasons:
  1) those resources difficult to collect because they are rare, since we searched for synthetic data and it's quite complicate process to create realistic description for a quite general purpose of patients and their properties;
  2) in relation to the motivation number 1, even if we accepted the idea of using real patients data, real word datasets are so specific and distant from our general purpose and they are really hard to trace back to individuals, we imagine due to the privacy concerns.
  We were able to retrieve interesting datasets related somehow to healthcare, but unfortunately they didn't share almost any of the general metadata we defined, for example suicidal rates in several countries, defibrillator distributions in Spain, results of strategic plans for prevention policies, etc. that we couldn't connect with our purpose.

- iTelos Data Life Cycle (Data Normalization, Syntactic Alignment, Semantic Alignment, Entity Matching and Application Alignment) are essential provide sharebility and reusability. These will be discussed later when it comes to the production of our reusable and shareable output, for the inception phase only focused about the creation of and adequate set of CQs.

2.4.2 Metrics evaluation

We were able to extract 11 concepts -in bold- and 29 properties -in the round brackets (7 + 0 + 4 + 0 + 3 + 4 + 3 + 3 + 3 + 0 + 2 = 29).
Patient (ID, age, date of birth, address, gender, phone, email), **Hospital**, **Doctor** (address, name, contact, specialization), **Location**, **Encounter** (start time, end time, description), **Prescription** (end date, start date, drug name, dosage description), **Condition** (start date, end date, type), **Lab result** (type, date, objective), **Allergy** (start date, end date, description), **Family disease history**, **Immunization** (date, disease).

In this paragraph we performed evaluation in a non-canonical but very precise way, instead of considering concepts and properties separately, we combine these pieces of knowledge together, to get an accurate score for each concept. This score is computed taking in account to which extent the properties of each concept are present also in the onthology or in the datasets. We want to underline that we considered overlapping concepts, not terms, so we used our intuitive interpretation for the cases in which different words have the same meaning for our final purpose (Healthcare service = Hospital, Practitioner = Doctor, Medication = Prescription, ...).

The logic for the concept coverage is that its score can span from 0 to 1, where 0 means that the concept is not present in our reference (onthology or datasets) or it has no properties represented in our reference, while 1 indicates that all the properties are represented in the reference. As an example, let us imagine that "Doctor" has 3 out of 4 properties present, so Cov(Doctor) = 0.75 , "Immunnation" has 1 out of 2 properties represented, so Cov(Immunization) = 0.5. The final score for the metric is simply the mean of concepts' ones.
The final score for the metric is simply the mean of concepts' ones. The same line of reasoning is done as far as regards extensiveness, but the subject are the properties in the reference that are not represented in our CQs.

- Metrics evaluation of concepts and properties in competencies questions versus concepts and properties in Ontologies (To compute these scores we retained that the best ontholology to consider is FHIR patient https://hl7.org/FHIR/resourcelist.html):

  1. Coverage=Coverage = (1 + 1 + 1 + 0 + 0.75 + 0.75 + 1 + 1 + 0.67 + 1 + 1) / 11 = 9.17 / 11 = **0.833**
  2. Extensiveness= (0.67 + 0 + 0.9 + 0 + 0.6 + 0.8 + 0.25 + 0.11 + 0.11 + 0 + 0.06) / 11 = 3.49 / 11 = **0.318**

- Metrics evaluation of concepts and properties in competencies questions versus concepts

and properties in datasets to compute these metrics we are considering the names of the datatables we collected as concepts and the attributes they contain as properties):
From the datasets we retrieved 20 concepts and 168 properties - In the following text we report all concepts and properties obtained by unification of all datasets while excluding redundant information:
patients (19 properties), vitals (17 properties), allergies (10 properties), clinicalnotes (4 properties), documents (7 properties), familyhistory (7 properties), imagingstudies (12 properties), immunizations (9 properties), lab results (8 properties), meds (12 properties), problems (4 properties), procedures (4 properties), refills (4 properties), socialhistory (2 properties), careplans(7 properties), conditions (4 properties), devices (5 properties), encounters (11 properties), doctors (11 properties), hospitals(11 properties).

1. Coverage=(1 + 1 + 0 + 0 + 1 + 1 + 1 + 1 + 1 + 1 + 1) / 11 =**0.82**

   On the other hand, sparsity was calculated using the method explained in class
2. Sparsity(concepts)= 1 - 11/20 = **0.45**
   Sparsity(properties)= 1 - 29/168 = **0.81**

3 Informal Modeling

In this phase of the iTelos methodology we will work both on knowledge and data levels. As a part of the knowledge (schema) level work-process, we will finalize the competency query formalization initiated in the Inception phase and create a ER graph based on those results. As for the data level, by utilizing the ER graph and the collected datasets, we will perform dataset filtering in order to select information relevant to the scope of our problem.

Due to a large number of overlapping properties between the Doctor and the Patient eTypes, we decided to introduce a new Person eType. This is a Common eType which both Doctor and Patient eTypes are extending (ISA relationship).
Additionally, the same was done for eType Allergy after seeing it's overlap with the eType Condition. Allergy eType was set to extend the eType Condition.

3.1 Purpose and CQ formalization:

Here, we will present the results of classifying and attributing the presented competency question. Every concept used in a CQ will be classified into core, common or contextual as well as being an object, a function or an action. Each the result of this classification will be used to define proper ETypes and their properties.

#	Concept	Object	Function	Action
1.1	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of hospitals	
	Context.	Prescription		Encounter
1.2	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of hospitals	
	Context.	Condition		Encounter
2.1	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of hospitals	
	Context.	Condition		Encounter
2.2	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of hospitals	
	Context.	Condition	Allergy	Encounter
2.3	**Core**		Patient	
	Common	Person		
	Context.	Family disease history		
2.4	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of hospitals	
	Context.	Prescription		Encounter
2.5	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of hospitals	
	Context.	Lab result		Encounter
2.6	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of hospitals	
	Context.			Encounter
3.1	**Core**		Patient	
	Common	Person	Location of patient	
	Context.			
3.2	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of hospitals	
	Context.	Condition	Allergy	Encounter
3.3	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of hospitals	
	Context.	Prescription		Encounter
3.4	**Core**	Hospital	Patient, Doctor	
	Common		Locations of hospitals	
	Context.	Condition		Encounter
4.1	**Core**		Patient	
	Common	Person	Locations of patients	
	Context.			
4.2	**Core**	Hospital	Patient	
	Common	Person	Locations of patients	
	Context.	Condition		

#	Concept	Object	Function	Action
4.3	**Core**	Hospital	Patient, Doctor	
	Common	Person	Locations of patients	
	Context.	Condition	Allergy	
4.4	**Core**	Hospital	Patient	
	Common	Person	Locations of patients	
	Context.	Family disease history		
5.1	**Core**		Patient	
	Common	Person	Location of patient	
	Context.			
5.2	**Core**	Hospital	Patient	
	Common	Person	Location of hospital	
	Context.	Immunization	COVID Immunization	
5.3	**Core**	Hospital	Patient	
	Common	Person	Location of hospital	
	Context.	Lab result	Results of PCR test	
6.1	**Core**	Hospital		
	Common		Locations of hospitals	
	Context.	Condition	COVID diagnosis	
6.2	**Core**	Hospital	Patient	
	Common	Person		
	Context.	Condition, Prescription	COVID diagnosis, Ivermectin therapy	
6.3	**Core**	Hospital	Doctor	
	Common	Person	Locations of hospitals	
	Context.		Doctor pulmologist	
6.4	**Core**		Patient in age category	Age category
	Common	Person		
	Context.	Lab result, Condition	Lab result blood, Mild COVID infection	Infection period
7.1	**Core**		Doctor, Doctor address	
	Common	Person	Locations of doctors	
	Context.		Doctor orthopedic	

Using the classification above we define ETypes and their properties. These entities, their relationships and properties are selected to be in line with the initial datasets collected and our project's purpose. However, as this is an informal modeling phase, they will be subject to modification in downstream integration processes.

- **Person:** a common entity type describing a person.

  Relations:

  - The eTypes Person and Doctor extend it

  - Has exactly one permenent location.

  Attributes:

  - *firstName*: [string]
  - *lastName*: [string]
  - *dateOfBirth*: [date]
  - *gender*: [boolean]

- **Patient:** a core entity type describing a patient.

  Relations:

  - Extends the eType Person and inherits all of its properties.
  - Has 0 or multiple medical prescriptions (current or previous).
  - Has 0 or multiple conditions.
  - Has 0 or multiple laboratory results.
  - Has 0 or multiple immunizations.
  - Has 0 or multiple medical encounters

  Attributes:

  - *patientID*: a unique identification of a patient [string]
  - *familyHistory*: [List(Tuple(Enum,String))] this list contains tuples where the first in the pair is an enumeration representing which relative (e.g. 1 - mother, 2 - father, 5 - aunt) and string defines the medical condition of that relative.

- **Doctor:** a core entity type describing a doctor - an employee of a hospital.

  Relations:

  - Extends the eType Person and inherits all of its properties.
  - Is affiliated with exactly one hospital.
  - Has performed 0 or multiple medical encounters. During each of these 0 or multiple medical treatments can be perscribed, 0 or multiple conditions can be diagnosed, 0 or multiple lab tests can be ordered and 0 or multiple immunizations can be ordered.

  Attributes:

  - *doctorID*: [string]
  - *specialization*: [string]

- **Encounter:** a contextual entity type describing a medical check-up that includes a patient, a doctor and a hospital.

  Relations:

- It involves 1 or none doctors. The possibility not to connect encounters to doctors is due to the fact that only Synthea dataset contains information which doctor conducted which encounter.
- It involves 1 patient.
- 0 or multiple medical conditions or allergies of a single patient may be diagnosed during the encounter.
- 0 or multiple treatments can be prescribed to a single patient during an encounter.
- 0 or multiple lab tests or immunizations can be ordered to a single patient during an encounter.

Attributes:

- *startOfEncounter*: [dateTime]
- *endOfEncounter*: [dateTime]
- *descriptionEncounter*: [string]

• **Condition:** a contextual entity type describing a condition diagnosed on a patient.

Relations:

- It Belongs to a single patient
- It can be related to an encounter, but doesn't have to be. This is due to the fact that the FIHR dataset does not contain the Encounter entity.

Attributes:

- *descriptionCondition*: [string]
- *startDate* : [date]
- *curedDate* : [date] - date on which the condition is cured. Can be empty if the condition is ongoing.

• **Allergy:** a contextual entity type describing an allergy diagnosed.

Relations:

- Extends the Condition eType (ISA relationship) and inherits all of it's properties.

Attributes:

- *descriptionAllergy*: [string]

• **Prescription:** a contextual entity type describing a medical treatment being prescribed for a patient.

Relations:

- It Belongs to a single patient.

  - It can be related to an encounter, but doesn't have to be. This is due to the fact that the FIHR dataset does not contain the Encounter entity.

  Attributes:

  - *startDate*: [date]
  - *stopDate*: [date]
  - *reason*: [string] - why is this medication taken.
  - *treatmentDescription*: [string] - which drug and what is the proper dosage and time intervals.

- **Immunization:** a contextual entity type describing immunization of a single patient for a single disease.

  Relations:

  - It Belongs to a single patient.
  - It can be related to an encounter, but doesn't have to be. This is due to the fact that the FIHR dataset does not contain the Encounter entity.

  Attributes:

  - *dateOfVaccination*: [date]
  - *diseaseVaccine*: [string]

- **LabResult:** a contextual entity type describing a result from a single laboratory testing for a single patient.

  Relations:

  - It Belongs to a single patient.
  - It can be related to an encounter, but doesn't have to be. This is due to the fact that the FIHR dataset does not contain the Encounter entity.

  Attributes:

  - *dateOfTest*: [date]
  - *testResult*: [string] - the test result itself.
  - *testingObjective*: [string] - what is being mesured.

- **Hospital:** a core type entity describing a single hospital.

  Relations:

  - It is related to a single location.
  - It has at least one or multiple doctors employed.

  Attributes:

  - *hospitalID*: [string]
  - *name*: [string]
  - *contact*: [string]
  - *capacity*: [integer]

- **Location:** a common type entity describing a location.

  Relations:

  - Related to all eTypes that have a location attribute: permanent location of a person or location of a hospital.

  Attributes:

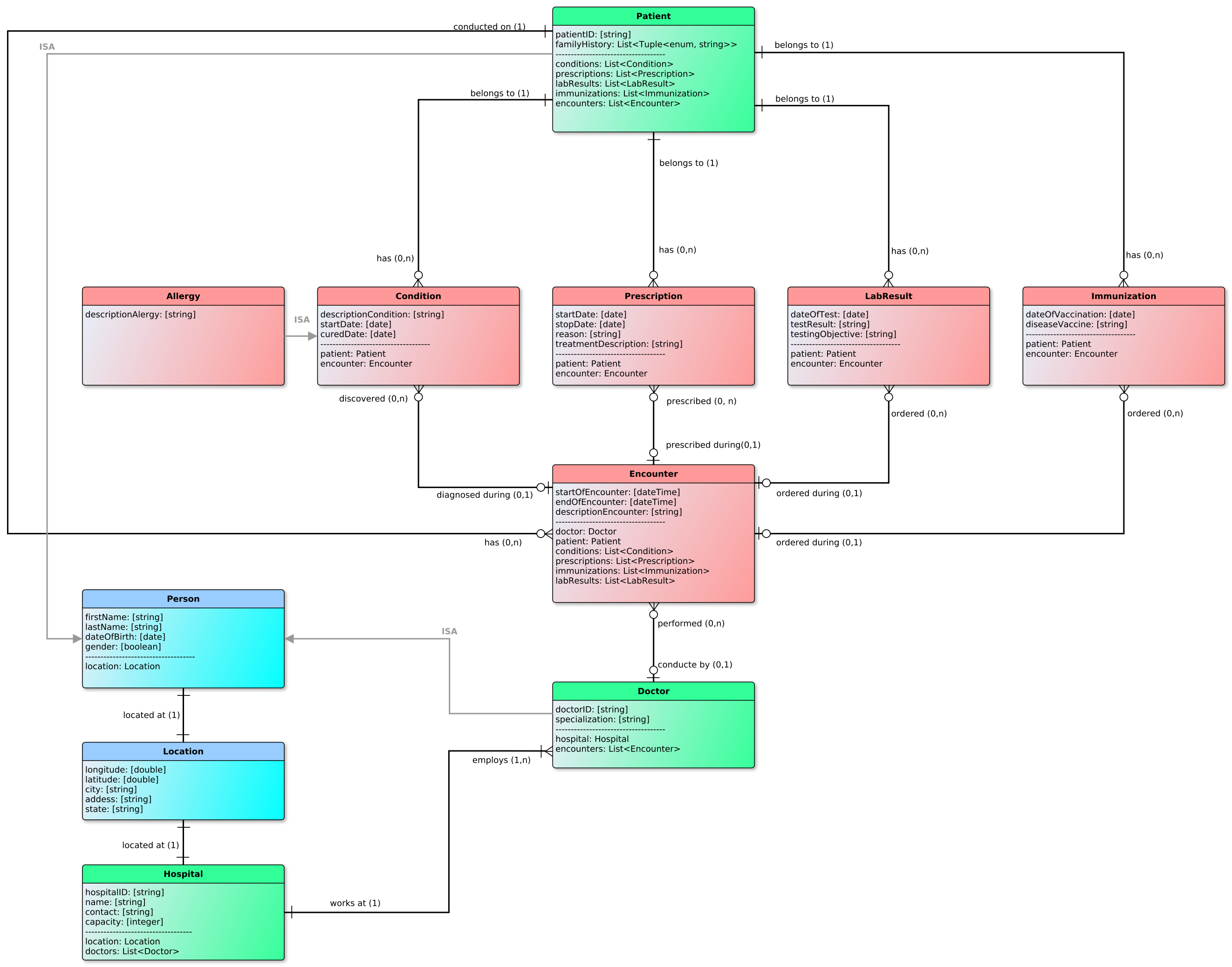
  - *longitude*: [double]
  - *latitude*: [double]
  - *city*: [string]
  - *address*: [string]
  - *state*: [string]

3.2 ER model

By relying of decisions made in the previous chapter, we constructed an entity relation (ER) diagram. Following the notation previously used, core eTypes are green, common ones in blue and contextual ones in red.

This diagram was defined by looking at the available data, the project's purpose and the competency questions. However, this procedure made us rethink about all the possible competency questions our solutions can provide an answer to, therefore we decided expand. Doing this allowed us to incorporate a larger amount of diverse data while still maintaining sparsity at a fairly low level.

We introduced the Person eType in order to avoid redundancy of defining basic information about our personas (Doctor and Patient). Both of these eTypes extend the eType Person with additional properties unique to them. Moreover, we changed the eType Allergy to be an extension of the eType Condition. Initially, there were two distinct eTypes due to the fact that they were categorized differently in our initial datasets. However, after observing that these entities share all attributes we connected them via IS-A relationship.



3.3 Data layer and dataset filtering

Using the results obtained in the previous chapters we sought to find a good trade-off between retaining as much as possible data from our initial dataset and not having many missing values. Several factors made this process harder. Firstly, datasets, vary greatly in the number of attributes describing each entity (EMRBOTS vs Synthea and FIHR Smart). Secondly, the number of entities themselves across datasets varies by orders of magnitude (FIHR Smart vs EMRBOTS and Synthea). For this reason, as we mentioned in the previous chapter, we decided to take a subset of 100 patients (and their related properties) from each dataset. Thirdly, due to language level obstacles, we needed time realizing which entities and attributes across the datasets had the same semantics.

Additionally, we conducted the data collection phase again. Using the Synthea command-line tool, we generated 100 synthetic patients with all related entities. We also downloaded the

reduced EMRBOTS dataset consisting of only 100 patients and their related entities. The FIHR dataset has 67 pateints and their entities.
The following chapter describes which properties where selected from initial datasets and the number of instances of each entity:

1. **EMRBOTS**: this dataset is not overly rich in attributes and it is the main cause of sparsity in our project. Almost all data properties of these entities were retained:

   AdmissionsCorePopulatedTable.csv: 372 instances

   - patient id
   - admission id
   - start date
   - end date

   AdmissionsDiagnosesCorePopulatedTable.csv: 372 instances

   - patient idthe datasets had the same semantics. The following chapter describes which properties and
   - admission id
   - diagnosis

   LabsCorePopulatedTable.csv: 3499 instances

   - patient id
   - admission id
   - lab analysis name
   - unit of measurement
   - lab result
   - date of test

   Here, we joined lab result and unit of measure fields into a single property.

   PatientCorePopulatedTable.csv: 100 instances

   - patient id
   - gender
   - date of birth
   - lab unit of measurement
   - lab result
   - date of test

2. **FIHR**: This dataset contains 7 data tables all of which are used. Low number of instances in this dataset lead us to subset the other two datasets.

   Patients.csv: 67 instances

   - patient id
   - start of allergy
   - end of allergy
   - allergy name
   - condition caused by allergy

   Allergies.csv: 18 instances

   - patient id
   - start of allergy
   - end of allergy
   - allergy name
   - condition caused by allergy

   Conditions.csv: 561 instances

   - patient id
   - start of conditon
   - end of condition
   - condition description

   FamilyHistory.csv: 16 instances

   - patient id
   - relative
   - condition description

   Immunizations.csv: 28 instances

   - patient id
   - date
   - disease against vaccination

   Labs.csv: 3513 instances

   - patient id
   - date
   - test name
   - test result
   - unit of measure

Here, we joined test result and unit of measure fields into a single property.
Meds.csv: 363 instances

- patient id
- starting date
- ending date
- description of the treatment

3. **SYNTHEA**: We consider this dataset the most detailed one. We included a number of entities, object and data properties contained only in this set because we believe that in a real-world scenario these information would be useful or even essential for a integrated health-care system.

   Patients.csv: 100 instances

   - patient id
   - birth date
   - first name
   - last name
   - gender
   - address
   - city
   - state
   - longitude
   - latitude

   The last five data properties are assigned to a Location eType. Every person has a single Location an object property.
   Allergies.csv: 62 instances

   - patient id
   - start of allergy
   - end of allergy
   - allergy name
   - condition caused by allergy
   - encounter id during which the diagnosis was made

   Conditions.csv: 1587 instances

   - patient id
   - start of conditon
   - end of condition
   - condition description

- encounter id during which the diagnosis was made

Doctors.csv: 368 instances

- doctor id
- hospital id in which he is employed
- name
- last name
- gender
- specialization
- address
- city
- state
- longitude
- latitude

The last five data properties are assigned to a Location eType. Every doctor has a single Location as an object property.
Encounters.csv: 4755 instances

- encounter id
- patient id
- doctor id
- description
- start date/time
- end date/time

Hospitals.csv: 151 instances

- hospital id
- name
- contact
- capacity of hospital
- address
- city
- state
- longitude
- latitude

As with the patient/doctor eType, last five properties are used for the Location eType.
Immunizations.csv: 1074 instances

- patient id
- encounter id
- date
- disease against vaccination

Medication.csv: 5146 instances

- patient id
- encounter id when the medication was prescribed
- description of the treatment
- reason of prescribing this drug
- starting date
- ending date

By extracting these information we have all the necessary data to perfectly match what was defined in our ER diagram.
Additionally, we included new metadata describing the key features of these filtered dataset using the SHAPEness software. This ttl file can be found on our Github project repository.

3.4 Evaluation of the informal modelling phase

In our ER model there are 11 concepts and 64 properties - in the round brackets. **Patient** (ID, family history, conditions, prescriptions, lab results, immunizations, encounters), **Encounter** (start time, end time, description, dcotor, patient, conditions, prescriptions, immunizations, lab results), **Prescription** (start date, stop date, reason, description, patient, encounter), **Condition** (start date, cured date, description, patient, encounter), **Lab result** (date, test result, objective, patient, encounter), **Allergy** (description), **Immunization** (date, disease, patient, encounter), **Hospital** (ID, name, contact, capacity, location, doctors), **Person** (first name, last name, date of birth, gender, location), **Doctor** (ID, specialization, hospital, encounters), **Location** (latitude, longitude, city, address, state, hospitals, persons).
The metrics were evaluated as done in class:

- Metrics evaluation on schema level of concepts and properties in ER model versus concepts and properties in competencies questions (we already retrieved these info in 2.4.2):
  1. We evaluate how much the proposed informal ER model covers CQs:

     Coverage(eTypes) = 8/9 = **0.88**
     Coverage(properties) = 27/29 = **0.9**
     We therfore determine that CQs have been covered well by the ER model.
  2. We evaluate how much the proposed informal ER model properly extends CQs:

     Extensiveness(eTypes) = 1/13 = **0.077**
     Extensiveness(properties) = 23/66 = 0.35

There is an much greater extension of prperties compared to eTypes, which is close to the ideal value.

- Metrics evaluation on data lavel of eType and properties in ER model versus eType and properties in datasets. In our datasets there are 9 concepts and 67 properties:

  allergies (start, stop, patient, encounter, allergy_description, condition, allergy), **conditions** (start, stop, patient, encounter, description), **doctors** (doctorid, organization, name, lastname, gender, specialization, address, city, state, lat, lon), **encounters** (id, start, stop, patient, doctor, description), **hospitals** (id, name, address, city, state, lat, lon, contact, capacity), **immunizations** (date, patient, encounter, disease), **labresults** (patient_id, encounter_id, test_date, test_name, result), **medications** (start, stop, patient, encounter, description, reason, description), **patients** (id, birthdate, first, last, gender, address, city, state, lat, lon)

  1. If one considers that some properties in the data will become eTypes in the following stages of integration, all of concepts and properties present in the ER are also present in the data. We therefore report both eType and property coverage of ≈ 1.
  2. Again, if we consider that some properties will become eTypes in the next stages of integration, we report both eType and property sparsity of ≈ 0.

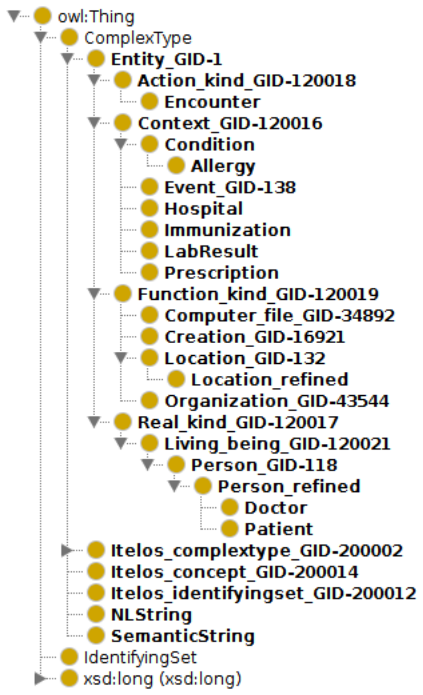
4 Formal Modeling

In this section, we describe the creation of the ontology in Protege, language alignment, preparation of the data for integration and lastly the creation of the metadata in the DCAT standard.

4.1 ETG generation

We started by using the predefined ontology adapted for the iTelos methodology. This ontology template was then was imported into Protégé, an open-source ontology editor and framework developed by the Stanford university.

By creating all the entity types, their object and data properties inside the given template following structure was built:


```
owl:Thing
  ComplexType
    Entity_GID-1
      Action_kind_GID-120018
        Encounter
      Context_GID-120016
        Condition
          Allergy
        Event_GID-138
        Hospital
        Immunization
        LabResult
        Prescription
      Function_kind_GID-120019
        Computer_file_GID-34892
        Creation_GID-16921
        Location_GID-132
          Location_refined
        Organization_GID-43544
      Real_kind_GID-120017
        Living_being_GID-120021
          Person_GID-118
            Person_refined
              Doctor
              Patient
    Itelos_complextype_GID-200002
    Itelos_concept_GID-200014
    Itelos_identifyingset_GID-200012
    NLString
    SemanticString
  IdentifyingSet
  xsd:long (xsd:long)
```

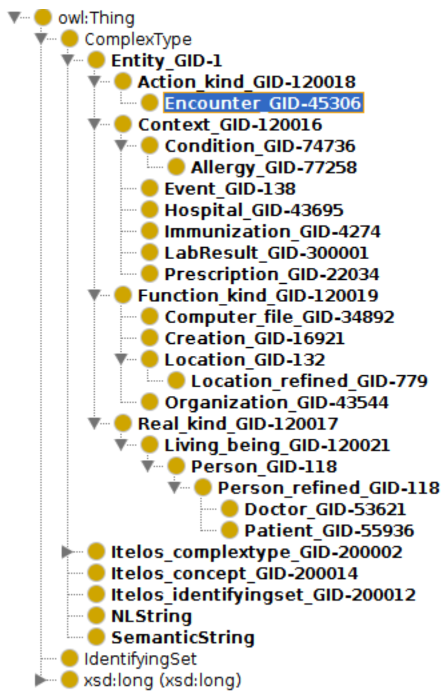
Only minor differences have been made with comparison to what was the output of the informal modelling phase:

- The eTypes **Person_refined** and **Location_refined** have been added to provide extensions to the **Person_GID-118**, **Location_GID-132** eTypes already existing in the template. These were added as subclasses.
- Due to the limited number of data types allowed to be used in the template (defined by the course), the data property **familyHistory** of **Patient** was changed to String type. All of the other data and object properties have been preserved compared to the output of informal modelling phase.
- It should be noted that both reverse relationship have not been placed nor the cardinalities. In all cases where exists an one-to-many relationship, only the side where Domain(Many)-to-Range(one) is modeled.

ETypes that have an ISA property have been modeled as a subclass of the more abstract EType.

Next, we uploaded the base-schema to KOS in order to integrate it with the framework ontology provided by UKC (a multilingual lexico-semantic resource - Universal Knowledge Core). KOS is a knowdive group tool that allows to explore the knowledge contained in the UKC and modify it. KOS is used for language alignment and the user is asked to solve the unresolved issues. If a existing concept is not chosen, the user defines a new concept. Here, during the language alignment process, a number of eTypes and especially data properties was not found in the UKC - therefore they had to be added. In many cases where the concept had to be added, a more abstract concept could have been selected as a synonym. However, to preserve exactness and expend the UKC resource, new concepts were placed.

This procedure resulted in a fully formal ETG, with GIDs placed on each concept. The hierarchy of the eTypes can be seen on the picture below. It can be seen that the same GID has been given to **Person** and **Person_refined**, however not to **Location** and **Location_refined**. It is unclear how this will affect downstream integration.



```
owl:Thing
  ComplexType
    Entity_GID-1
      Action_kind_GID-120018
        Encounter_GID-45306
      Context_GID-120016
        Condition_GID-74736
          Allergy_GID-77258
        Event_GID-138
        Hospital_GID-43695
        Immunization_GID-4274
        LabResult_GID-300001
        Prescription_GID-22034
      Function_kind_GID-120019
        Computer_file_GID-34892
        Creation_GID-16921
        Location_GID-132
          Location_refined_GID-779
        Organization_GID-43544
      Real_kind_GID-120017
        Living_being_GID-120021
          Person_GID-118
            Person_refined_GID-118
              Doctor_GID-53621
              Patient_GID-55936
    Itelos_complextype_GID-200002
    Itelos_concept_GID-200014
    Itelos_identifyingset_GID-200012
    NLString
    SemanticString
  IdentifyingSet
  xsd:long (xsd:long)
```

Since the obtained graph structure is too complex to be visualized, to further examine the structure and the properties, both ETGs have been published on our GitHub repository.

4.2 Data management (syntactic heterogeneity)

The three datasets were joined using LibreOffice and command-line tools. All of the column names were renamed in such way to represent the same things. Each instance of each eType has been appended with a string value denoting which dataset it is originating from. For all eTypes all object and data properties were present, exepct for **familyHistory** of **Patient** eType. Due to a very low number of instances, it was decided for it to be removed from downstream integration since it will cause sparsity. After these steps, we have nine separete csv tables each representing an eType with its properties. Etypes Location and Person are indirectly contained inside others that have an object property relationship with them.

We then resolved all data type and format misalignments amont the datasets. All of our simulated data was in English, therfore no language misalignments existed. Data formats for time-date fields have been left as they are because they will be synchronized in the following step while using Karmalinker.

4.3 Formal Modeling evaluation

Evaluation of our schema relies on the calculation of several metrics. It is important to note that since our reference schema is extremely large and our addition very small these metrics are quite skewed. Furthermore, most of our ontology was based on the existing FHIR schema. The following were evaluated:

- If the formal ETG and its Etypes are properly defined by their properties, using metric Cue(ETG) and Cue(Etype). It provides information about sharebility and unity, as well as property richness.
- If the proposed ETG is different from the reference ontologies, using metric Sparsity
- If the ETG is well-designed, and information in the ETG is correct. By sampling from ETG and then checking manually.

In total, our ETG contains 11 different eTypes and 45 properties in total. This number is lower than the number of properties of the ER model because we are not counting reverse relations - these are not modeled in Protege. Firstly, cue validity was calculated for each eType, then the total was calculated as the sum Cue(EType):

- Patient: 4
- Condition: 1.37
- Allergy: 2.37
- Prescription: 2.87
- LabResult: 3.37
- Immunization: 2.37
- Encounter: 4
- Doctor: 3.33
- Person: 4.5
- Location: 7
- Hospital: 4.83

Giving rise to the Cue(ETG) = 40.01. Contrary to what was said in class, we observe that the common eTypes have the highest cue validity values, while almost all contextual lower.

Next, the sparsity metric was evaluated by comparing our ETG to the chosen reference ontology - FHIR. The FHIR ontology contains 855 classes in total, giving rise to sparsity value of 0.98. Information about the total number of data properties in the FHIR ontology was not present and for this reason the property sparsity was not calculated. We expect it to be close to 1 because almost all concepts and properties in our ontology have been covered by FHIR.

Finally, the ETG was manually checked several times for non-logical connections and to make sure that integration with the UKC ontology went smoothly.

5 Data Integration

In this chapter we describe using Karmalinker, a web application used for data manipulation and integration. Additionally, we describe solving semantic heterogenity problems.

5.1 Data management

All nine csv files describing our eTypes and their properties have been uploaded to Karmalinker. As the ontology, the ETG given by KOS was used. Prior to that, it was manually checked in Protege to identify potential mistakes. Prior to entity matching, some leftover syntactic issues had to be solved:

- All date formats in all files have been converted to ISO format. Now all properties of all eTypes have the same data types across datasets.
- IDs were assigned to the following eTypes: Allergy, Condition, Prescription, LabResult and Immunization. This was simply done by assigning unique integer values to each instance of the mentioned eTypes.

These modification made possible to preform entity aligning. No errors or difficulties were came across during this procedure. RDF files were exported from Karmalinker and uploaded to our GitHub repository. These will be used in the next phase to exploit our knowledge graph. Additionally, R2RML models were also retrieved and uploaded to the repository.

Since this project deals with simulated data, no entity matching had to be done. In a real world scenario, a carefully developed methodology would have to be applied, because we expect a single patient to have medical data in multiple facilities. This data often contains redundancies and conflicting data.

5.2 Data integration phase evaluation

In the evaluation of this phase we perform on the data level and we aim to measure:

- Can the competency queries from the initial phases be answered and how complete are those answers. This will be covered in the next chapter where we utilize the knowledge graph and query it to answer a couple of CQs.
- Evaluate weather our dataset is sufficiently used. We use this by computing the sparsity metric between our dataset schema and the final etg. Due to the fact that all eTypes from the ETG are present in the dataset schema, and all attributes apart from **familyHistory** we obtain an sparsity metric of 0.022. This value close to zero verifies that integration has been done successfully.

Next, we turn to non-quantitative metric and inspect both schema and data levels of the EG:

- Consistency dimension:

- No cycles in class inheritance have been implemented.
    - No polysemous elements have been used. This was one of the mistakes made in the Formal modeling phase thus making us redo it.
    - Each property has only one domain and one range.
    - There are no semantically identical classes. In cases where there are a lot of similarities between them, an ISA relationship is used (Condition - Allergy).

- Accuracy dimension:
    - The mentioned ISA relationships all have been modeled using ’subClassOf’ relationship in Protege - thus avoiding the incorrect relationship problem.
    - For each of eType there is at least one instance present, meaning our hierarchy is not over-specified. This is an exception for the class **Person_refined_GID-118** which is an abstract class (not instatiable), but other classes extend it.

- Completeness dimension:
    - We belive non of our elements are isolated, due to the fact that all of the classes and properties have connections to the rest of the ontology.
    - We made sure that all of our properties have a domain present.

6 Outcome exploitation

After integrating our data successfully, the knowledge graph is ready to be used. In the following section we make one final report of of the eTypes, their properties and number of instances in each. Only the classes were written with the GID appended by KOS, properties were written without it for the sake of keeping the notation simpler. Moreover, when writing the object properties, we write only the ranges. Abstract classes, such as Person_refined_GID-118, have no instances related to them.

- **Person_refined_GID-118:** Extends the Person_GID-118 eType

  Object properties:
    - *has_location*: [Location_refined_GID-779]

  Data properties:
    - *has_firstName*: [string]
    - *has_lastName*: [string]
    - *has_dateOfBirth*: [datetime]
    - *has_gender*: [boolean]

- **Patient_GID-55936:** Extends Person_refined_GID-118 eType

  Instance count: 267
  Data properties:

  - *has_patientID*: [string]

- **Doctor_GID-53621:** Extends Person_refined_GID-118 eType

  Instance count: 386

  Object properties:

  - *has_hospital*: [Hospital_GID-43695]

  Data properties:

  - *has_doctorID*: [string]
  - *has_specialization*: [string]

- **Location_refined_GID-779** Extends Location_GID-132 eType, from which it inherits longitude and latitude properties.

  Instance count: 307
  Data properties:

  - *has_city*: [string]
  - *has_state*: [string]
  - *has_address*: [string]

- **Prescription_GID-22034:**

  Instance count: 5509
  Object properties:

  - *has_patient_of_prescription*: [Patient_GID-55936]
  - *has_been_prescribed_during*: [Encounter_GID-45306]

  Data properties:

  - *has_prescription_start*: [datetime]
  - *has_prescription_end*: [datetime]
  - *has_prescription_reason*: [string]
  - *has_prescription_description*: [string]
  - *has_prescriptionID*: [string]

- **LabResult_GID-300001:**

  Instance count: 7012
  Object properties:

  - *has_patient_of_labResult*: [Patient_GID-55936]
  - *has_been_tested_during*: [Encounter_GID-45306]

  Data properties:

  - *has_testing_date*: [datetime]
  - *has_result*: [string]
  - *has_objective*: [string]
  - *has_labResultID*: [string]

- **Immunization_GID-4274:**

  Instance count: 1102
  Object properties:

  - *has_patient_of_immunization*: [Patient_GID-55936]
  - *has_been_ordered_during*: [Encounter_GID-45306]

  Data properties:

  - *has_vaccination_date*: [datetime]
  - *has_disease_vaccination*: [string]
  - *has_immunizationID*: [string]

- **Hospital_GID-43695:**

  Instance count: 151
  Object properties:

  - *has_hospital_location*: [Location_refined_GID-779]

  Data properties:

  - *has_hospital_contact*: [string]
  - *has_capacity*: [int]
  - *has_hospitalID*: [string]
  - *has_hospital_name*: [string]

- **Condition_GID-74736:**

  Instance count: 2520
  Object properties:

  - *has_patient_of_condition*: [Patient_GID-55936]
  - *has_been_identified_during*: [Encounter_GID-45306]

  Data properties:

  - *has_condition_start*: [datetime]
  - *has_condition_end*: [datetime]
  - *has_condition_description*: [string]
  - *has_conditionID*: [string]

- **Allergy_GID-77258:** Extends Condition_GID-74736 class and inherits all of its data and object properties.

  Instance count: 80
  Data properties:

  - *has_allergy_description*: [string]
  - *has_allergyID*: [string]

- **Encounter_GID-45306:**

  Instance count: 5127
  Object properties:

  - *has_patient*: [Patient_GID-55936]
  - *has_doctor*: [Doctor_GID-53621]

  Data properties:

  - *has_encounter_start*: [datetime]
  - *has_encounter_end*: [datetime]
  - *has_encounter_description*: [string]
  - *has_encounterID*: [string]

In total, we have 11 eTypes, described by 49 properties and having their 22377 instances.

6.1 KG exploitation and GraphDB

Next, we sought to utilize this resource in NeoJ4's Graphdb by answering some questions similiar to our initial CQs. The CQs have been formulated without looking closely at the data, so the names and locations of entities in the database are different, however the concpetual idea and the purpose are the same. In the following section we provide thee SPARQL queries for three different hypothetical scenarios. None of these queries is simply a lookup but requires a graph traversal.

- List conditions, names and last names of each patient living in Boston. This query resembles something Francesco from CQs might ask (4.2).

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix kdi:<http://knowdive.disi.unitn.it/etype#>
select ?person ?first_name ?last_name ?description ?condition where
{
    ?location kdi:has_city_GID-44481_Type-779 "Boston" .
    ?person kdi:has_location_GID-779_Type-118 ?location .
    ?person rdf:type kdi:Patient_GID-55936 .
    ?condition kdi:has_patient_of_condition_GID-55936_Type-74736 ?person .

    ?person kdi:has_firstname_GID-34006_Type-118 ?first_name .
    ?person kdi:has_lastname_GID-34003_Type-118 ?last_name .
    ?condition kdi:has_condition_description_GID-300008_Type-74736 ?description
}
```

This returns 85 results which can be found on our GitHub repository under name query1.csv.

- Show all treatments alongside the patient ID and birth date of patients with COVID-19 that are older than 44 years. This resembles something Maria from CQs might ask.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX kdi:<http://knowdive.disi.unitn.it/etype#>
select ?description ?patient ?birth_date ?age ?treatment where
{
    ?condition kdi:has_condition_description_GID-300008_Type-74736 ?description .
    filter(lcase(str(?description)) = "covid-19")
    ?condition kdi:has_patient_of_condition_GID-55936_Type-74736 ?patient .
    ?patient kdi:has_birth_date_GID-81257_Type-118 ?birth_date .
    bind( year(NOW())-year(?birth_date) as ?age )
    filter(?age >= 45)
    ?prescription kdi:has_patient_of_prescription_GID-55936_Type-22034 ?patient .
    ?prescription kdi:has_treatment_description_GID-300004_Type-22034 ?treatment
}
```

This returns 371 results which can be found on our GitHub repository under name query2.csv.

- List number of patients with COVID-19 across hospitals. This query is a good representation of our solution connecting data from three different tables.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix kdi:<http://knowdive.disi.unitn.it/etype#>
select DISTINCT ?hospital ?hospital_name (COUNT(?hospital_name) as ?total)  where
{
    ?condition kdi:has_condition_description_GID-300008_Type-74736 ?description .
    filter((lcase(str(?description)) = "covid-19"))

    ?condition kdi:has_been_identified_during_GID-45306_Type-74736 ?encounter.
    ?encounter kdi:has_doctor_GID-53621_Type-45306 ?doctor .
    ?doctor kdi:has_hospital_GID-43695_Type-53621 ?hospital .
    ?hospital kdi:has_hospital_name_GID-2_Type-43695 ?hospital_name .
} group by ?hospital ?hospital_name
```

This returns 9 results which can be found on our GitHub repository under name query3.csv.

6.2 Conclusions

This project describes our reasoning behind the complete process of integration which has a defined purpose and context to be used in. Since we are not domain experts, we were not able to define the purpose and use-cases ideally from the start. Our purpose became clearer while doing the project. The mistake in this case was over-specifying the CQs compared to what our data had to offer. Apart form a purpose, a closer inspection into the available datasets is needed before writing CQs. This caused our final solution to be matching the purpose completely but not the our initial idea.

The number and content of the collected datasets was quite poor. We could only obtain simulated data of limited content. It is also important to keep in mind that the solution was not build with the purpose of efficient exploitation - a different database schema should have been employed then. Apart from this issue, working with real data would be much harder integration task for several reasons:

- Datasets would include much more information about each patient. Likewise, the data is more sparse.
- Entity matching would definitely be a problem. An careful assessment would have to be made for each case how to resolve conflicting data.
- Anonymization of individual would have to be necessary. A platform that provides security in a sense of assigning access levels to each user would need to be used.

On top of common EHRs, bio-bank data is often added. Data reporting genomic variants can be joined with patient data. An excellent example of this is the country-wise integration of EHR data in Estonia. Genomic data plays a big role here since Estonia has one of the largest bio-banks in Europe. An expansion of this project could be random sampling of available genomic samples and placing them in the database.