ICT-2011-288048

EURECA

Enabling information re-Use by linking clinical REsearch and CAre

IP
Contract Nr: 288048

Deliverable: 9.4
Solution providing uniform access to sources

Due date of deliverable: (31-05-2014)
Actual submission date: (14-08-2014)

Start date of Project: 01 February 2012 Duration: 42 months

Responsible WP: WP 9

Revision: final

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Service	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (excluding the Commission Services)	

0 DOCUMENT INFO

0.1 Author

Author	Company	E-mail
Kerstin Rohm	FhG-IBMT	Kerstin.rohm@ibmt.fraunhofer.de
Gabriele Weiler	FhG-IBMT	Gabriele.weiler@ibmt.fraunhofer.de
Monique Hendriks	Philips	Monique.hendriks@philips.com
Jasper van Leeuwen	Philips	jasper.van.leeuwen@philips.com
Haridimos Kondylakis	FORTH	kondylak@ics.forth.gr
Kristof De Schepper	CUSTODIX	kristof.deschepper@custodix.com
Annette ten Teije	VUA	annette@cs.vu.nl
Dietlind Zühlke	FhG-IAIS	dietlind.zuehlke@iais.fraunhofer.de
Sergio Paraíso Medina	UPM	sparaiso@infomed.dia.fi.upm.es
Ruud Van Stiphout	UOXF	ruud.vanstiphout@oncology.ox.ac.uk
Cyril Krykwinski	IJB	cyril.krykwinski@bordet.be
Timm Kissels	FhG-IAIS	timm.kissels@iais.fraunhofer.de
Stefan Rüpping	FhG-IAIS	stefan.ruepping@iais.fraunhofer.de
Nikolaus Forgó	LUH	nikolaus.forgo@iri.uni-hannover.de
Magdalena Góralczyk	LUH	goralczyk@iri.uni-hannover.de
Alan Dahi	LUH	dahi@iri.uni-hannover.de

0.2 Documents history

Document version #	Date	Change
V0.1	01/03/2014	TOC
V0.2	05/05/2014	Version within contribution of most partners
V0.5	23/07/2014	Improved version
V0.6	06/08/2014	Improved version (for internal review)
V1.0	14/08/2014	Final version

0.3 Document data

Keywords	
Editor Address data	Name: Kerstin Rohm Partner: Fraunhofer-IBMT Address: Ensheimer Str. 48, 66386 St. Ingbert, Germany E-Mail: Kerstin.rohm@ibmt.fraunhofer.de

0.4 Distribution list

Date	Issue	E-mailer
		fp7-eureca-all@listas.fi.upm.es
		INFSO-ICT-288048@ec.europa.eu
		Benoit.abeloos@ec.europa.eu

Table of Contents

1 Introduction

Despite the big number of initiatives focusing on interoperability, in order to improve the uniform access of clinical research and care data, there are still lots of barriers. Those barriers are preventing an efficient discovery and knowledge transfer to researchers and clinicians. Some of the aforementioned barriers are the following:

- Clinical care data in EHR systems are seldom properly accessible for secondary use
- The current separation into clinical care and clinical research data
- The lack of interoperability, common standards and terminologies

The mentioned facts cause suboptimal issues like:

- The detection of serious side effect is difficult, because therapy and drugs are documented into several independent systems.
- There is a wide knowledge gap between the care that is provided in top research sites and standard care sites.
- The enrolment process of patients for clinical trials is slow and inefficient, because of redundant data entry and inconsistences.

One of the main goals of the EURECA project is to improve these situations by building up a framework that will bridge the gap between medical care and research. This is realised by the specific EURECA platform that provides a harmonized, secure access to data from both domains and enables the re-use of all available data.

This EURECA platform is built upon a multi-layered architecture, where specific components and modules are operating through interfaces. The layer that is mainly responsible for semantic data integration is called the Semantic Integration layer. It harmonises data from different sites using the EURECA Common Data Model (CDM) which is terminology linked to the EURECA Core Dataset (popular clinical terminologies like SNOMED, LOINC, etc.). [1] The uniform access of data through die EURECA platform is shown in Figure 1.

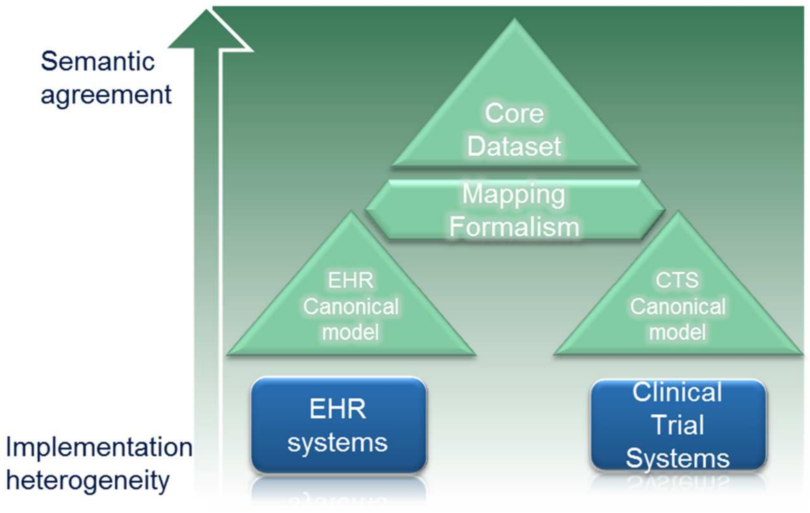


Figure 1: Uniform access of clinical data through the EURECA platform [1]

The Semantic Integration Layer is built on the Semantic Interoperability Platform that enables the EURECA end-user applications to retrieve inhomogeneous medical data from different clinical care and research data sources in a uniform, secure and flexible way.
Deliverable 4.3/9.3 [3] describes the mapping formalisms of the EURECA CDM as well as the data provision by the clinicians. In this document, we describe how the different EURECA applications retrieve these data from the EURECA CDM. We describe the Semantic Interoperability Platform for data integration as well as security aspects necessary for the data access.

1.1 Structure of the Deliverable

This deliverable is structured as follows:

Chapter 2 gives an overview of the EURECA architecture. Several layers of this architecture are involved to harmonise the data from different clinical sites. In particular, the EURECA Semantic Interoperability Platform provides methods to the EURECA end-user applications to access clinical data in a uniform way.

Chapter 3 summarises the functionalities of the EURECA Semantic Interoperability Platform. Section 3.1 describes how to map and load the original clinical data sources into the EURECA CDM. Section 3.2 describes the techniques that the EURECA end-user applications can use in order to retrieve the data from the EURECA CDM.

Chapter 4 focuses on the legal and security issues.

Chapter 5 shows how the EURECA specific end-user applications can retrieve data from the EURECA platform in a uniform way.

2 General Overview of the EURECA Architecture

The EURECA platform is a framework of tools, which can be easily interconnected in different configurations, tailored to the needs of different environments and end-users. This enables the re-use of available data and the linkage of data of the care and clinical domains. The following chapter will give a short overview of the different layers and components of the EURECA platform. More details can be found in deliverable 2.2 “Initial EURECA architecture”. [2]

2.1 Layers of EURECA Architecture

The EURECA platform is designed as a multi-layered architecture, where every component or service can be mapped to one of these layers. Figure 2 shows the different layers.

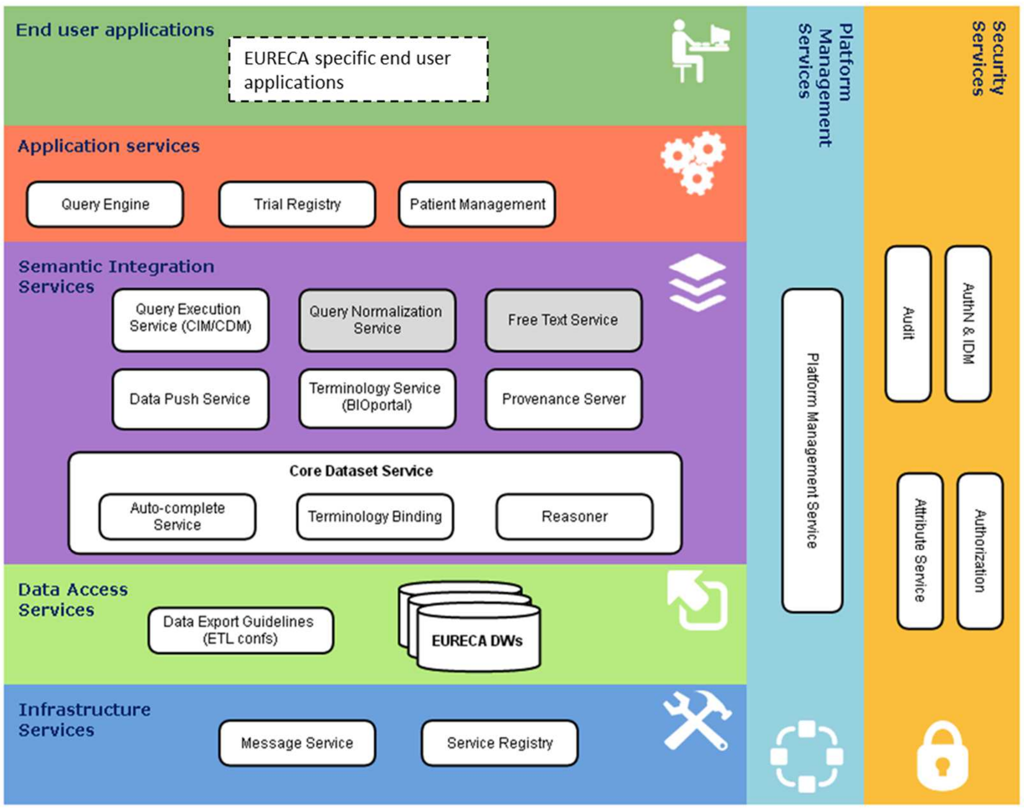


Figure 2: EURECA architecture and its layers

- End-user applications layer:
  The components of this layer represent the endpoints to the end users. The applications are offering the underlying back-end functionality in a user-friendly way. This is usually realised by using graphical user interfaces.
- Application layer:
  This layer provides the core functionality of the EURECA services as it houses the application services. These generic services provide secondary but essential

functionalities for the EURECA end-user applications. Application services are for example:

- o *Query Engine:*
  The Query Engine translates SNAQL (Snaggletooth Query language) queries to SPARQL queries and executes these queries on the Query Execution Service. SNAQL allows users to define queries in a close natural language without having a deep technical knowledge.
- o *Trial Registry:*
  The Trial Registry provides a central metadata repository. In this repository, a set of eligibility criteria of actively recruiting trials is stored. The repository relies on the Biomedical Research Integrated Domain Group (BRIDG)[1]. These criteria can be received by the EURECA end-user applications.
- o *Patient Management:*
  The Patient Management is a repository holding general basic information about patients.

- Semantic Integration layer:
  The main goal of this layer is to homogenise the access to clinical data. The components of this layer are operating in the EURECA Semantic Interoperability Platform. The main functionalities of this layer are:

  - o To load homogenous clinical data sources into the EURECA Semantic Interoperability Platform (Data Push Service)
  - o Uniform data retrieval from the EURECA Semantic Interoperability Platform to the EURECA upper layer application components and services (Query Execution Service).

- Data Access layer:
  This layer represents the clinical data in the EURECA Data Warehouses through the EURECA CDM with semantics according to the EURECA Core Dataset.

- Security layer:
  This layer is connected to all other architectural layers. The EURECA security solution consists of re-usable modular components that respectively deal with authentication (authN), authorisation (authZ), audit and privacy enhancing (i.e. services oriented specifically at data privacy protection).

- Infrastructure layer:
  The components in this layer provide service communication and service management capabilities to other EURECA components.

- Platform Management layer:
  These components help to manage the integration of components in the EURECA platform.

This deliverable focuses on the uniform access of data sources. Several mentioned layers from the EURECA architecture are needed in order to realise these functionalities. The EURECA end-user applications access the data sources from the EURECA Data Warehouses through the Query Execution Service semantically. The legal issues for the processing of these data are respected through the services of the Security layer. In the

[1] http://www.bridgmodel.org/

following chapters, we describe in more detail the data access capabilities of the Semantic Integration layer and the Security layer. Chapter 5 describes in detail how the different EURECA applications and services utilise the EURECA infrastructure for uniform data access.

3 EURECA Semantic Interoperability Platform

The Semantic Integration Layer of the EURECA architecture (see chapter 2) provides the main functionalities for homogeneous storing and retrieving of clinical data. The interactions of the components and services of this layer is realised through the EURECA Semantic Interoperability Platform.

The clinical data in the EURECA Data Warehouses (DW) are described through a consistent data model – the EURECA Common Information Model (CIM). The CIM consists of the EURECA CDM and the EURECA Core Dataset. The CDM is a HL7 RIM based data repository. The EURECA Core Dataset enables a coding of these data by vocabularies from SNOMED CT, LOINC and HGNC.

This chapter focuses on the description of the interaction between the components of the Semantic Interoperability Platform (see Figure 3). This platform enables homogenised access to the clinical data. We will first briefly summarise the steps of the data provision for the EURECA DWs. After that, we will describe the technologies of the Semantic Interoperability Platform in order to retrieve the available data from the EURECA DWs in a uniform way.

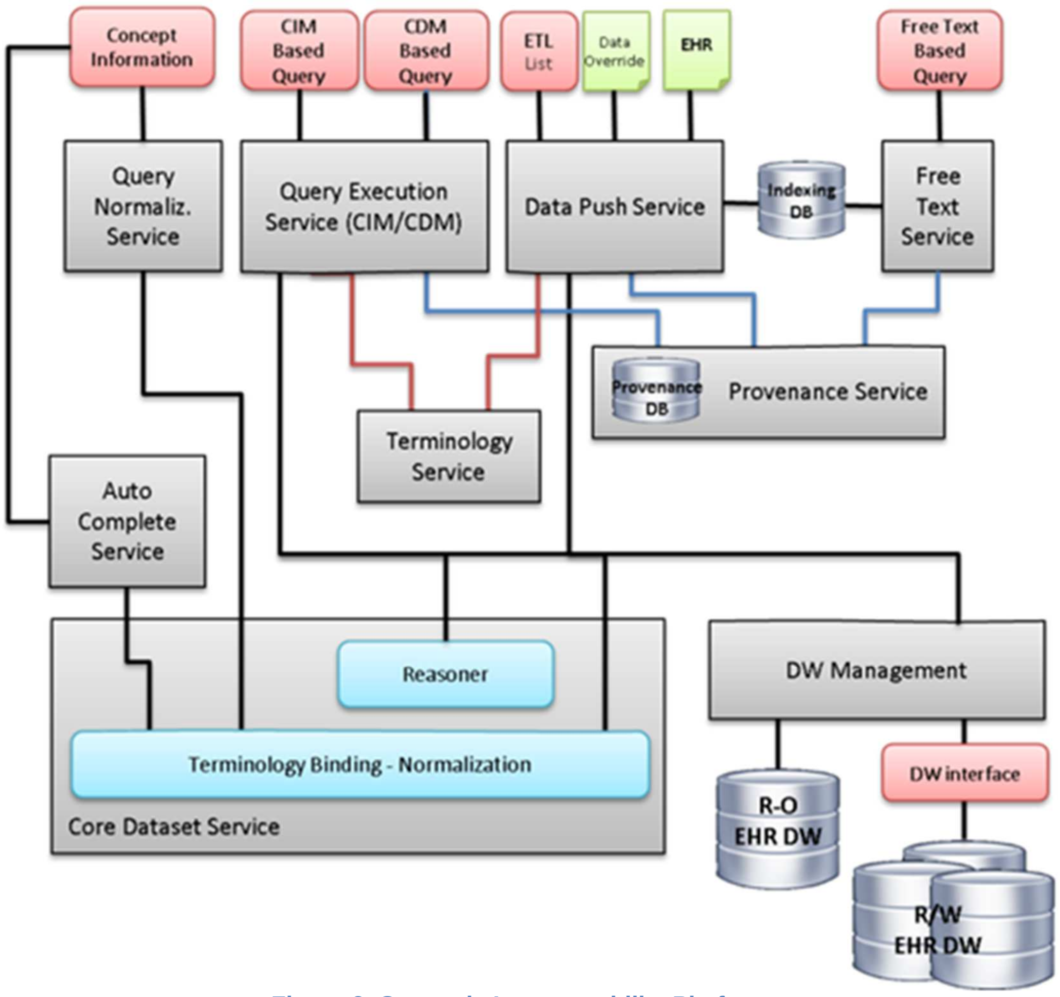


Figure 3: Semantic Interoperability Platform

3.1 Data Provision for EURECA CDM

The accessible data of the EURECA DWs was provided by the different clinical sites. The standards and technologies that are used by the clinical partners are very different. As a consequence the IT technicians at each hospital site have to process the provided data sets before they can be processed by the EURECA **Data Push Service**, which harmonises the data and pushes it into the DW.

As the EURECA DWs store data compliant to a HL7 RIM based data model, it is required to transform the original data sources into the EURECA Data Push Service format which is HL7 v3. The medical terms of these documents have to be coded by the EURECA Core Dataset (or codes from, ICD 10, ICD 9, CTCAE, MeSH, MedDRA or NCIt).

The original clinical data sources are available in structured and unstructured form and have to be handled by different methods. The processes of the data provision for the EURECA CIM can be summarised into the following steps:

1. Transform original clinical data into a EURECA Push Service compatible format
    a. Structured data:
        i. The terms of the original data sets have to be annotated by the EURECA Core Dataset (vocabularies from SNOMED CT, LOINC and HGNC). This task can be realised with the help of the web application Concept2HL7[2]. Codes from, ICD 10, ICD 9, CTCAE, MeSH, MedDRA or NCIt are also possible but an additional processing by the **Terminology Service** is needed to link them to the Core Dataset (see step 3).
        ii. The annotated clinical data have to be transformed into HL7v3 CDA message standard. This can be realised with the help of specific tools from Mirth Connect[3] or templates, which were created for the specific EURECA requirements and are built upon the IHE recommendations.
    b. Unstructured data: **Free Text Service**
        i. Natural Language Processing (NLP) extracts specific information from the original unstructured data and translates them into medical concepts. The output format is triple-based.
        ii. A mapping process maps the NLP output into HL7v3 CDA messages.
2. Send HL7v3 CDA messages via SOAP messages to EURECA Data Push Service through WebService technology over the Security layer.
3. Process HL7v3 CDA messages by EURECA Data Push Service
    a. The Extract, Transform and Load process (ETL) extracts the information from the received CDA messages and inserts this information into the EURECA CDM.
    b. The Normalisation process homogenises the received data for the EURECA CDM following the Core Dataset. As far as a concept is not covered by a Core Dataset code, it has to be handled by the **Terminology Service** first.

[2] http://kandel.dia.fi.upm.es:8078/login
[3] http://www.mirthcorp.com/products/mirth-connect

4. Store the data into the EURECA CIM of the EURECA DW.

Details about the data provision in EURECA can be found in the deliverable 4.3/9.3 “Initial proposal for the mapping formalism and mappings to EHR and CT models”. [3]

3.2 Data Access form the EURECA CDM

A EURECA end-user application can retrieve the homogeneously stored data from the EURECA CDM through the EURECA Semantic Interoperability Platform. The following chapter describes the EURECA technologies that enables retrieving of these data. Figure 4 shows a simplified data retrieving process.

The major services for retrieving the aforementioned data are the following:

- Query Execution Service:
  The end-user application can retrieve data from the EURECA CDM by CDM based SPARQL queries (details see section “3.2.2 Query Execution Service”).
- Query Normalization Service:
  This service creates templates which can be used to build CDM based SPARQL queries upon a Core Dataset concept (details see section “3.2.1 Query Normalization Service”)

These two major services are coupled with the functionalities of the following EURECA services:

- Core Dataset Service:
  The Core Dataset Service is responsible for inferring vocabulary knowledge in the different components of the platform. This is carried out by the following parts:
  - *Auto-Complete Service*: Within this service, it is possible to obtain all information about a Core Dataset concept, which are title, code and context.
  - *Reasoner*: This module is needed to get hierarchical knowledge of the Core Dataset concepts. This will be shown by the following example:
    The CDM based SPARQL part:
    FILTER (?targetSiteCode IN (**isAnySubclassOf(91532001**)))
    have to be processed by the Reasoner Module. The module returns all child concepts for the SnomedCT code 91532001 (Female mammary gland structure) and expands the original SPARQL queries by these child concepts.
  - Terminology Binding: This component is used for the mapping of the CDM and the Core Dataset concepts
- Terminology Service:
  The Terminology Service translates the terminologies ICD 10, ICD 9, CTCAE, MeSH, MedDRA and NCIt to Core Dataset terminologies. This is realised with the help of the BioPortal’s[4] functionalities.

[4] http://bioportal.bioontology.org/

Some of the EURECA end-user applications require also the interaction with services from the application layer. They are described in chapter 2.1 “Layers of EURECA Architecture”.

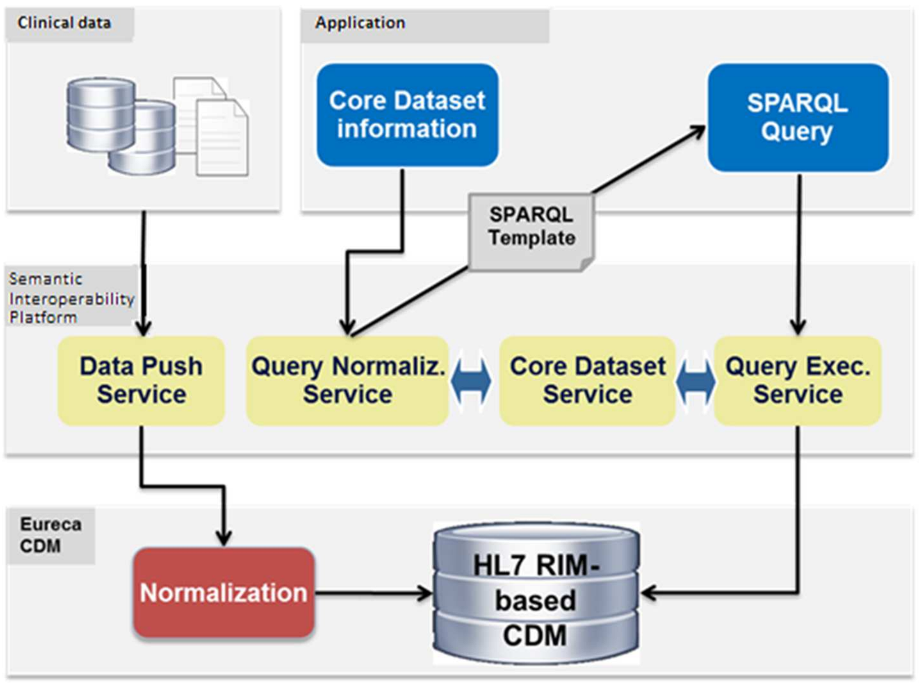


Figure 4: Interaction of main components of the EURECA Semantic Interoperability Platform

3.2.1 Query Normalization Service

EURECA CDM based SPARQL queries can be built with the help of the Query Normalization Service. Therefore, a Core Dataset concept or concepts have to be sent to the Query Normalization Service via WebService technologies[5]. It returns a list of templates within possible filters and attributes in XML format, which can be used to query specific values from the EURECA CDM.

The building of these SPARQL templates is achieved through the interaction with the Core Dataset Service. The Core Dataset Service extracts all needed information related to the given concept or concepts. This normalisation process is based on the SNOMED guidelines to homogenise post-coordinated concepts. The corresponding CDM template can be built upon this extracted information with the help of a EURECA CDM specific Query Template Library.

The following example shows the building of a CDM based SPARQL query within the Query Normalization Builder. The query should retrieve all patients that had a diagnosis of *“Neoplasm of female breast (disorder)”* (*126927001*, SNOMED CT code):

1. Call the Query Normalization Service with the SNOMED CT code *126927001*.

[5] WebService WSDL: https://euler.dia.fi.upm.es:8443/QueryBuilder_dev/services/QueryBuilderService?wsdl

2. The Query Normalization Service will return a template that enables to query the observations with the normalised form of the original concept. The whole Template is added in the appendix A.1. In this case the normalised form of the original concept is:
    a. Code: *108369006*, *Neoplasm (morphologic abnormality)*
    b. Target Site*: 91532001, Female mammary gland structure (body structure)*
3. Within this query template, it is possible to obtain a general query (query between <*sparqlQuery*> clauses, see example appendix A.1). It is possible to add more attributes or filters to this general query by the <*optionalStructure*> clauses.
4. It is possible to execute this by the Query Execution Service now.

The Query Normalization Service is described in detail in section 2.2.3 of deliverable 4.3/9.3. [3]

3.2.2 Query Execution Service

The Query Execution Service receives CDM based SPARQL queries (e.g. created with the Query Normalization Service see chapter 3.2.1), processes them and returns the result as an XML file. The transfer is carried out via WebService technologies.

For some SPARQL queries semantic reasoning is needed (details see explanation of Core Dataset Services). These queries require a processing of the Reasoner Module first, which expands the original SPARQL query. This is realized through a SESAME Server. SESAME[6] is an open source Java framework for storing, querying and reasoning with RDF and RDF Schema.

As mentioned before, the data in the EURECA DWs are described by the EURECA CDM. The CDM is built upon the HL7 RIM, which is a relational database. In order to enable SPARQL query access to these data, a mapping of the CDM described data to RDF triples is needed. This is realized by MORPH [9]. MORPH (formerly called ODEMapster) is an RDB2RDF engine, which follows the R2RML specification (http://www.w3.org/TR/r2rml/). Figure 5 visualises the processing of the Query Execution Service.

[6] http://www.openrdf.org/

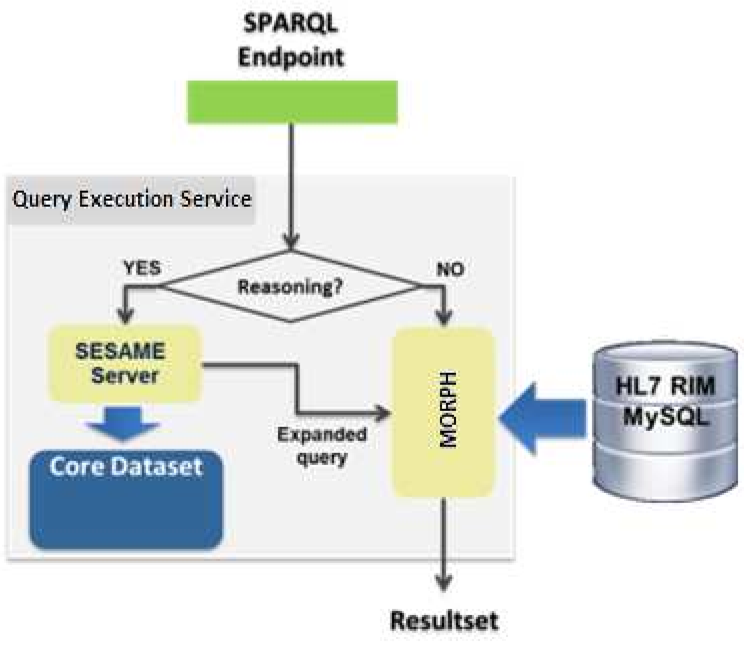


Figure 5: Processing of the Query Execution Service

The following example shows a CDM based SPARQL query that can be executed by the Query Execution Service. The query retrieves the ICD10 code and the patient id for all patients with a nosocomial infection:

```
SELECT DISTINCT ?patientId ?valueCode
       WHERE {
              hl7rim:person_id ?patientId.instPerson
              hl7rim:person_code '337915000'.sapiens (organism)
              OPTIONAL{
                     ?instPerson    hl7rim:person_birthTime ?birthTime
              }
              ?instPerson    hl7rim:person_role ?instRole2.
              ?instRole2     hl7rim:role_entityId ?patientId.
              ?instRole2     hl7rim:role_participation ?instPart2.
              ?instPart2     hl7rim:participation_entityId ?patientId.
              ?instPart2     hl7rim:participation_act ?instAct.
              ?instAct       hl7rim:act_code ?valueCode;
              hl7rim:act_id  ?id.
              OPTIONAL{
                     ?instAct hl7rim:act_creationTime ?creationTime
              }
              FILTER (?code IN (isAnySubclassOf(19168005)))
              OPTIONAL{
                     ?instAct     hl7rim:act_observationAct ?instObs.
                     ?instObs     hl7rim:observationAct_actObservationValues
                     ?instValues.?instValues          a    hl7rim:actObserva-
                     tionValues
                     OPTIONAL{
                            ?instValues     hl7rim:actObservationValues_code
                     ?valueCode
                     }
              }
       }
```

- When executing this query, the Reasoner expands the SPARQL part "isAnySubclassOf (19168005)" by all concepts of the Core Dataset that are subclasses of the 19168005 (SNOMED CT concept for nosocomial infection).
- The MORPH Server translates the Result Set of the CDM into RDF
- The Query Execution Service returns the following result in XML format:

```
<sparql>
    <head>
        <variable name="patientId"/>
        <variable name="valueCode"/>
    </head>
    <results>
        <result>
            <binding name="patientId">
                <literal>128f463277b55c4db84ec9c826bd6ff0</literal>
            </binding>
            <binding name="valueCode">
                <literal>A49.0</literal>
            </binding>
        </result>
   </results>
</sparql>
```

4 Security Services

The EURECA datasets contain sensible patient information that is subjected to the strict legal requirements of the EURECA legal framework. The legal framework provides a set of legal and ethical (contracts, consent) and security (access controls) preconditions which have to be fulfilled before access to data is granted. Further information can be found in deliverable 7.1 “Initial EURECA Legal and Ethical Requirements”. [7]

In order to comply with these legal requirements, a technical security solution has been developed and deployed. Several security tools were integrated in the EURECA semantic interoperability solution as described in the following sections (see also deliverable 7.3 “EURECA Security and Privacy Services” [8]).

4.1 De-identification of Patient Data

The patient data in the EURECA project can be covered by three legal domains (see also deliverable 7.3 [8]):

- Care domain:
  Data processing operations are carried out for treatment purposes, which directly benefit the patient.
- Research domain:
  Data of different provenience (care, but also other research, or completely new data) is being used for scientific research purposes.
- Clinical trial preparation domain:
  Consists of scenarios, which reach beyond care, but do not fall under the research category. Those include supporting researchers and physicians in developing a clinical trial, as well as finding suitable patients for it.

Patient data of scenarios in the care legal domain (e.g. Patient Screening Service), do not need any pseudonymisation or an anonymisation processing. However, in some scenarios anonymisation of the patient information is required in the exploitation phase (those that are in the research domain and a subset of the clinical trial preparation domain). For these scenarios the data that is imported in the Semantic Interoperability Platform needs to be anonymised (mainly de-identified and pseudonymised) by the services of the EURECA privacy framework. An overview of the data flow during this de-identification process is given in Figure 6.

Beside other, the access of pseudonymised or anonymised patient data is realised through PIMS (Personal Information Management System) and CATS. PIMS is a versatile platform for patient record linkage and management of patient identifiers, pseudonyms and biological material IDs originating from different data sources. A configurable matching engine allows linking records representing the same real world patient, hereby compensating for incomplete or differently structured information. During the pseudonymisation, original ids will be linked with pseudonyms in PIMS. CATS, is the processor of data and will do the pseudonymisation transformation on the data. PIMS will be used to keep the links between the original id’s and the pseudonyms.

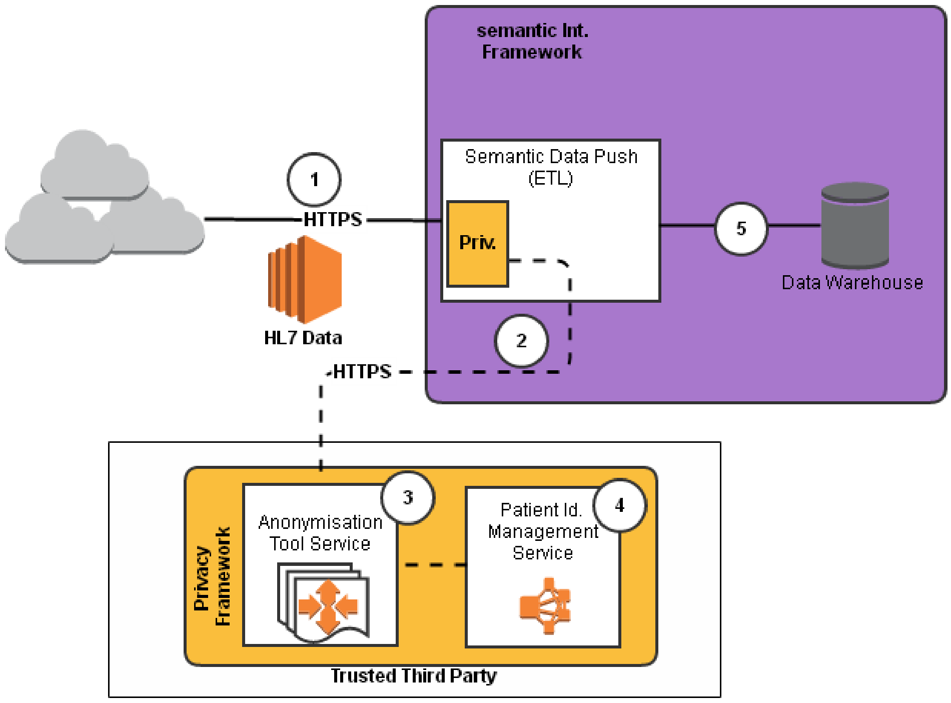


Figure 6: Privacy framework integration

The de-identified and pseudonymised process can be summarised by the following steps:

1. Patient data encapsulated in HL7 messages (see deliverable 4.3/9.3 [3]) are sent from the data provider to the Data Push Service of the Semantic Interoperability Platform (see chapter 3.1). This data has already had a first round of de-identification (out-of-scope for the Semantic Interoperability Platform), done at the data provider site (see deliverable 7.3 [8]).
2. The Data Push Service will ETL the data to the EURECA CDM. In this tool, a privacy component is installed to intercept the data and send it to an anonymisation tool service like CATS (Custodix Anonymisation Tool Service).
3. In the Anonymisation Tool Service, the input data is transformed to anonymised output data using privacy profiles that contain a predefined set of transformation rules. De-identification rules will clear patient identifying information from the input. Pseudonymisation rules will send patient identifying information to the patient identity management service in order to get a pseudonym for each patient in the input data.
4. The Patient Identification Management Tool will generate for each patient a unique pseudonym using the provided identifying information. The link between the identifying patient data and the pseudonym is managed by this service and will be controlled by a Trusted Third Party (TTP) (see deliverable 7.1 [7]). Only the TTP can access and view the links.
5. After the Anonymisation Tool Service outputs the data, the anonymised data is further processed with ETL and stored in the CDM. Services that access the Semantic Interoperability Platform only work with anonymised data, conform to the EURECA privacy framework.

4.2 Secure Data Access by the Security Services

The EURECA datasets contain sensible (possibly anonymised) patient information that are subjected to the strict legal requirements of the EURECA legal framework. In order to comply with these legal requirements, a technical security solution has been developed that enables authentication authorisation and auditing about the queried data sets from the EURECA CDM.

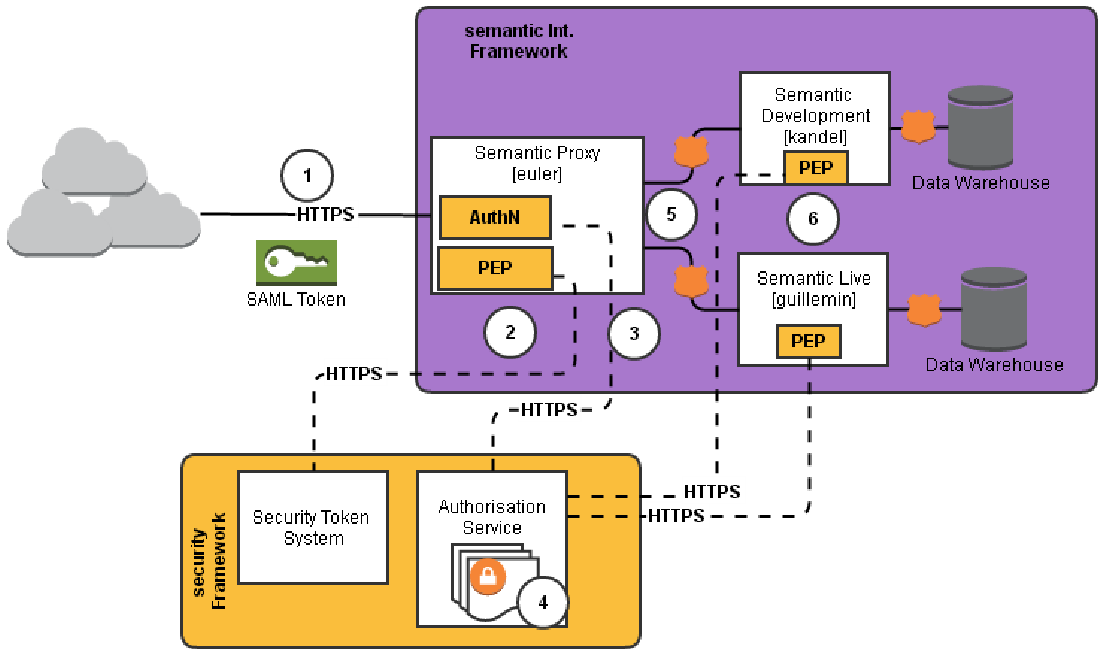


Figure 7: Security Framework Integration

The process of the secure data receiving can be summarised by the following steps (each of the described action here is sent to the central auditing service) (see Figure 7):

1. Each incoming data query request (sent over HTTPS) should contain a signed SAML token issued by a EURECA identity provider. It contains attribute information (name, organisation, and id) about the requester.
2. In the Semantic Proxy, the SAML token will be validated (signature and timestamp checking). If the SAML token is found invalid, access will be rejected to the request.
3. Next, the Policy Enforcement Point of the Semantic Proxy will generate an access request including the attribute information about the requester. This request is sent to the central EURECA Authorisation service (over HTTPS).
4. The Authorisation Service will retrieve the request coming from the Semantic Proxy and will generate an access decision, based on defined policies. These policies are compliant with the contracts[7] signed by the EURECA partners. The Authorisation Service will return the access decision back to the Semantic Proxy.

[7] Hence through the Authorisation Service the system assures that the Institution has signed the End-user Agreement and that the individual signed an Annex C to this contract (which has a function of a Non-disclosure-agreement).

All EURECA partners interested in accessing the data signed the end-user agreement.

5. If access was granted, the query request will be forwarded to the internal semantic services (that are only accessible by the Semantic Proxy). If access was denied, access will be rejected to the request.
6. The Semantic Services contain an extra Policy Enforcement Point, which will handle fine-grained access control (access control on the internals of the datasets). The flow is similar as in step 3 and 4.

The contracts between the End-users and CDP are included in the Annex to D7.1.

5 Uniform Data Access in EURECA Services

The previous chapters described the technologies of the EURECA platform that enables access to the uniform described data sources from the EURECA CDM in a secure way. This chapter focuses on the specific EURECA services, which are relying on these uniform data access capabilities of the EURECA platform. We will especially focusing on how these services access and integrate data.

5.1 Contextualisation (Physician facing)

Through the Contextualisation Service the user can get additional information for a specific patient. Therefore, available patient data coming from a local EURECA DW can be combined with public available data sources from the internet. The Semantic Interoperability Platform provides the needed technologies in order to combine all these data.

The service integrates the following data sources:

Pharmacological:
- Linked Open Drug Data
- Optional: OpenFDA (Publication expected in summer 2014)
- Optional: CTCAE

Guidelines:
- Optional: Dutch Breast cancer guideline (Each entry will be annotated)

Tumour classifications:
- TNM
- Optional: WHO (Brain cancer)
- Optional: Gleason grading system (prostate cancer)

Literature:
- PubMed (Initially we will focus on case descriptions of rare co-morbidities)

An example of a contextualised query might be the generalisation of medication (brand) names the patient is receiving, which is relevant when investigating whether or not a patient is eligible for clinical trials. These medications will be generalised either by physiological function or by active molecular compound. Additionally, symptoms experienced by the patient can be linked to known side effects of his/her medication, and otherwise be scored as potential unknown side effects.

The processing of the Contextualisation Service will be summarised in the following lines. The user selects one of the displayed pre-defined queries into the Physician Facing Contextualisation Tool for a specific patient the doctor is treating at that moment. This pre-defined query is processed by the Query Engine of the Application Layer and will be sent to the Query Execution Service.

The Query Execution Service, combines the contextualised query with the patient data from the EURECA CDM with Public available data from the internet with the help of the Terminology Service and reasons over them with the Reasoner of the Core Dataset Service. In order to ensure secure retrieving of patient data from the EURECA CDM, this process will be carried out within the functionalities of the Security Layer.
Results of the contextualisation are presented to the physician in the Physician Facing Contextualisation Tool.

5.2 Patient Screening

The main goal of the Patient Screening Service is to assist physicians in checking the eligibility of patients for a given set of trials. Therefore, specific patient data like age, gender, diagnosis, histology, stage of disease, tumour volume, primary diagnosis or relapse comorbidity have to be compared with eligibility criteria of trial protocols coming from the Trial Registry.

The end-user selects in this scenario a patient of interest and a preferred set of trials in a user interface. A criteria matcher will start matching the eligibility criteria of the selected trials with the data of the selected patient. The patient data are retrieved through the Query Execution Service from the local EURECA DW. The eligibility criteria of the selected trials are downloaded from the Trial Registry. Each returned criterion of the Trial Registry is formalised as a Groovy script with a SPARQL template included. This template will be filled by the data of the selected patient and sent to the Query Execution Service of the Semantic Interoperability Platform. Based on the returned SPARQL result, the criteria matcher generates a decision using the logic of the Groovy script (match/non-match/undetermined).

For the security issues it has to be mentioned, that the displayed patients list only contains patients that are located on the site where the screening is done. The shown patients are also depending on the rights and roles of the logged in end-user.

The following example shows the data flow of the Patient Screening Service. The criterion “A female patient younger than 70 years” is matched on a selected patient.

- The criteria matcher receives the following eligibility criterion (formalised as a Groovy script) from the Trial Registry:

```
import com.custodix.integrate.criteriamatcher.model.*;
import com.custodix.integrate.criteriamatcher.wsclients.SemanticLayerWSClient;
import com.custodix.integrate.criteriamatcher.model.MatchResult;
import groovy.time.TimeCategory;

// CRITERION: Female patient of at most 70 years. (<= 70)
def query="""\
SELECT DISTINCT ?id ?birthTime ?gender
WHERE {
        ?instLiSu a hl7rim:livingSubject; hl7rim:livingSubject_id ?id.
        OPTIONAL {
                ?instLiSu hl7rim:livingSubject_birthTime ?birthTime;
                hl7rim:livingSubject_administrativeGenderCode ?gender.
        }
        FILTER (?id = "$patientID")
}""";

Map<String,String> values = semanticLayerWSClientImpl.executeQuery(target,
query);
QueryResult[] result = new QueryResult[1];
result[0] = new QueryResult();

if(values == null){
        result[0].setResult(MatchResult.UNDETERMINED);
} else {
        String gender = values.get("gender");
        String birthTime = values.get("birthTime");
        Date birthDateTime = new Date().parse("yyyy-M-d H:m:s",birthTime);
        String id = values.get("id");
        Date startDate = null;
        use (TimeCategory) {
        startDate = new Date() - 71.years;
}
if ( gender.equals("248152002") && ( birthDateTime > startDate) ){
        result[0].setResult(MatchResult.MATCH);
} else {
        result[0].setResult(MatchResult.NONMATCH);
};
Evidence evidence = new Evidence();
evidence.setEvidenceId(id);
result[0].setEvidence(evidence);
};
return result;
```

- The criteria matcher fills in the SPARQL query template included in the Groovy script based using the data of the selected patient:

```
SELECT DISTINCT ?id ?birthTime ?gender
WHERE {
    ?instLiSu a hl7rim:livingSubject;hl7rim:livingSubject_id ?id.
  OPTIONAL {
    ?instLiSu hl7rim:livingSubject_birthTime ?birthTime;
              hl7rim:livingSubject_administrativeGenderCode ?gender.
  }
  FILTER (?id = " fictitious1")
```

- This SPARQL query is send to the Query Execution Service via a SOAP request.
- The criteria matcher service receives the following result from the Query Execution Service:

```
<sparql>
    <head>
        <variable name="id"/>
        <variable name="birthTime"/>
        <variable name="gender"/>
    </head>
    <results>
        <result>
            <binding name="patientId">
                <literal>fictitious1</literal>
            </binding>
            <binding name="birthTime">
                <literal>1965-03-01T00:00:00.0</literal>
            </binding>
            <binding name="gender">
                <literal>248152002</literal>
            </binding>
        </result>
   </results>
</sparql>
```

- The criteria matcher generates a decision (match/non-match/undetermined) based on the logic in the Groovy script. The result is displayed to the end-user via the user interface.

5.3 Trial Recruitment

In the local trial recruitment scenario, a hospital site wants to recruit patients for a new clinical study. For this, a recruiter checks the patients of their site DWs with the eligibility criteria of the clinical study.

The hospital sites will query their own datasets from their local DWs in order to see how much patients can be recruited for the eligibility criteria of the given study. Again as in the Patient Screening Service, eligibility criteria of the Trial Registry should be matched with the patient data coming from the EURECA DW.

The next lines will summarise the workflow of the Trial Recruitment Service. The end-user selects a preferred set of eligibility criteria of a trial via a graphical user interface (see Figure 8). The available criteria and trials are downloaded from the Trial Registry. In the Trial Registry the eligibility criteria are formalised as SNAQL scripts (next to some additional logic). The selected eligibility criteria (which are SNAQL scripts) are sent to the Query Engine.

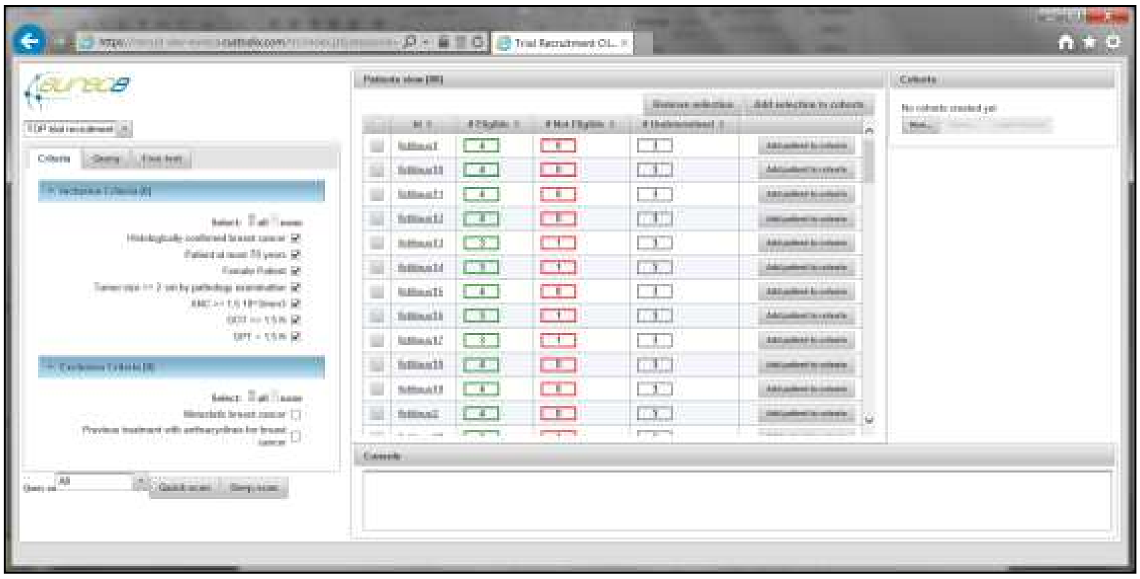


Figure 8: Trial Recruitment GUI

The Query Engine will translate the selected criteria to SPARQL queries by using the Query Normalization Service (which contains SPARQL templates). These SPARQL queries are executed on the Query Execution Service of the Semantic Interoperability Platform. A list of suitable patients are returned (usually pseudonymised) by the Query Execution Service. The security aspects, as described in chapter 4.2, have been respected.

The following example shows the data flow of the recruitment of female patients in a specific trial.

- The user selects the “female patients” criterion for a specific trial. The Query Engine builds the following SPARQL query based on this request:

```
SELECT DISTINCT ?patientId ?birthTime
```

```
        WHERE{
                        ?instEntity hl7rim:entity_id ?patientId;
                        hl7rim:entity_determinerCode ?patientDeterminerCode;
                        hl7rim:entity_livingSubject ?instLiSu.?instLiSu
                        hl7rim:livingSubject_birthTime ?birthTime.
                OPTIONAL{
                        ?instLiSu
                        hl7rim:livingSubject_administrativeGenderCode ?gender.
                }
                OPTIONAL{
                        ?insLiSu hl7rim:livingSubject_birthTimeTolerance.
                ?birthTimeTolerance
                }
                FILTER (?gender = "248153007")
        }
```

- This SPARQL query is sent to the Query Execution Service via a SOAP request
- The Query Execution Service returns the following result:

```
<sparql>
    <head>
        <variable name="patientId"/>
        <variable name="birthTime"/>
    </head>
    <results>
        <result>
            <binding name="patientId">
                <literal>fictitious1</literal>
            </binding>
            <binding name="birthTime">
                <literal>1965-03-01T00:00:00.0</literal>
            </binding>
        </result>
        <result>
            <binding name="patientId">
                <literal>fictitious10</literal>
            </binding>
            <binding name="birthTime">
                <literal>1966-02-02T00:00:00.0</literal>
            </binding>
        </result>
   </results>
</sparql>
```

- The result is displayed to the user via the GUI

5.4 Protocol feasibility

Yakobo, the protocol feasibility application, assesses the feasibility of a clinical protocol by estimating the possible enrolment rate given by (historical) patient data. The patient data resides in the EURECA DWs. The functionalities of Yakobo and its data retrieving through components of the EURECA architecture is described in the following lines.
The Query Engine provides a higher-level query language designed to select patient cohorts. Yakobo formulates each criterion in SNAQL and obtains the patient cohort (containing amongst others the list of patient identifiers and the id of the evidence used in the criterion evaluation). Yakobo aids the user in creation of the SNAQL statements by offering a collection of widgets (which are in effect SNAQL templates). Yakobo uses the Auto-Complete Service of the Core Dataset Service to populate parts of these

templates. Finally, protocol feasibility definitions are persisted in the Trial Registry and can be accessed from there.

The Query Engine translates SNAQL into SPARQL (internally, it uses the Query Normalisation Service to obtain SPARQL templates to aid in the translation process). The SPARQL queries are used to obtain patient data from the Query Execution Service of the Sematic Interoperability Platform. The security aspects, as described in chapter 4.2, have been respected. Because of the mentioned functionalities, it is possible to get the following information:

- For each criterion in a clinical protocol:
    - A cohort of patients that match the criterion
    - A cohort of patients that do not match the criterion
    - A cohort of patients of who it is not known whether they match or do not match the criterion

- Given a patient id and a criterion, any evidence for the criterion evaluation

The following example illustrates the processing of the Yakobo application. Figure 9 shows a widget to select the patients in a particular age range, in this case patients at least 50 years old. The corresponding (generated) SNAQL would be result = *patient agegteq.50.years*.

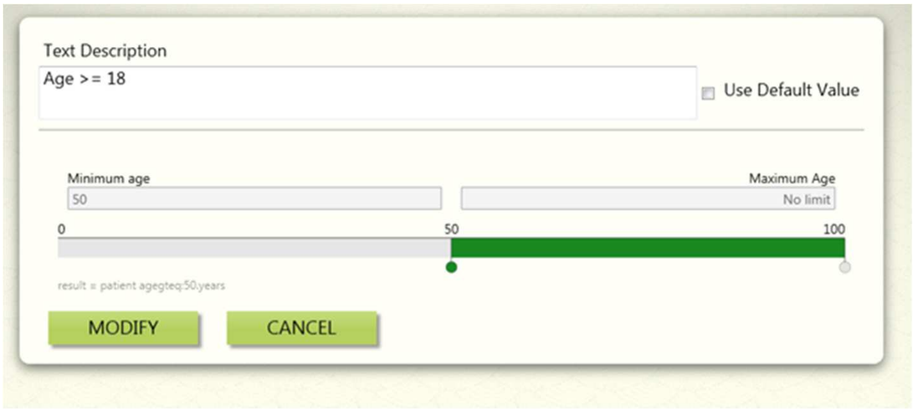

Text Description
Age >= 18
☐ Use Default Value
Minimum age: 50
Maximum Age: No limit
result = patient agegteq:50.years

Figure 9: widget of Yakobo Graphical User Interface

The Query Engine generates the following SPARQL query that can be executed by the Query Execution Service.

```
SELECT DISTINCT ?patientId ?birthTime
        WHERE{
                ?instEntity hl7rim:entity_id ?patientId;
                hl7rim:entity_determinerCode ?patientDeterminerCode;
                hl7rim:entity_livingSubject ?instLiSu.?instLiSu
                hl7rim:livingSubject_birthTime ?birthTime.
                OPTIONAL{
                        ?instLiSu
                        hl7rim:livingSubject_administrativeGenderCode ?gender.
                }
                OPTIONAL{
                        ?insLiSu hl7rim:livingSubject_birthTimeTolerance
                ?birthTimeTolerance
                }
                FILTER (?birthTime <= “1964-08-01T00:00:00.0” )
        }
```

The return values of this query are handled by the application and the result can be displayed in the GUI.

5.5 Update of Guidelines

The Update of Guideline application provides various selection for guideline designer and other researchers in order to find relevant research findings based on the corresponding guideline content (alternatively called guideline items or guideline conclusions). The update of guideline application is integrated in the web application SemanticCT[8].

The service integrates the following data sources:

Guidelines:
- Evidence-based Clinical Guidelines, e.g., Dutch Breast cancer guidelines

Ontologies/Terminologies:
- SNOMED CT
- UMLS
- MESH
- Optional: ICD10
- Optional: HL7 RIM

Literature:
- PubMed (Optional: Abstract with semantic annotations)

The interface will provide a selection of options in order to select the formalised evidence-based clinical guidelines with different topics. For each selected topic, the application will show a list of the guideline items with their evidence levels, refereed literature and their evidence classes.

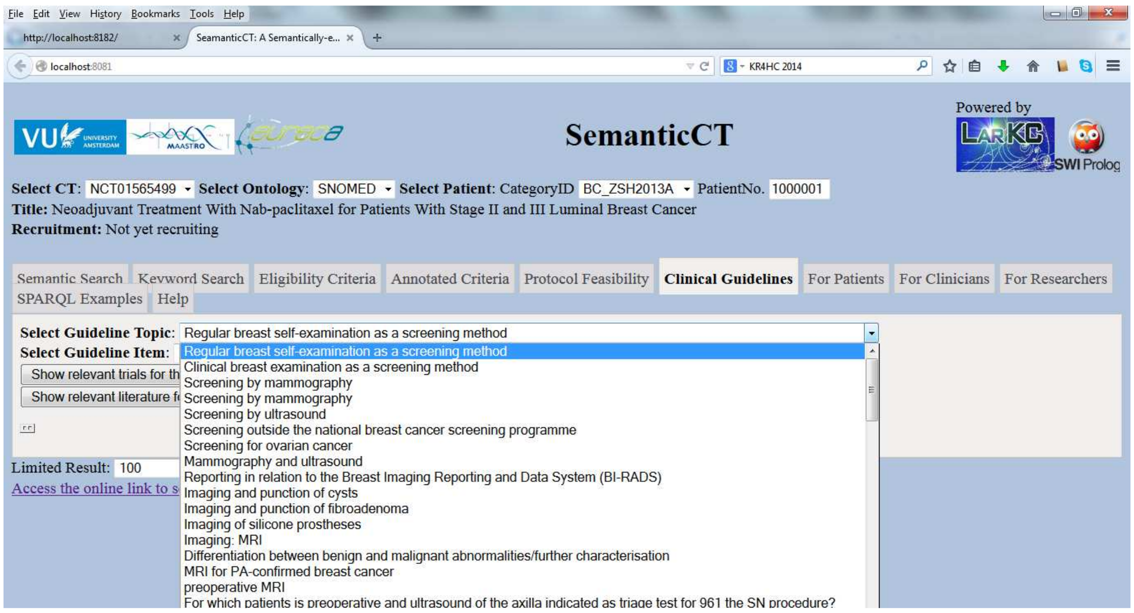
The user can select a guideline item and then different functionalities for checking the relevant literature. Those functionalities include detecting relevant research findings with different options on temporal aspects, such as "latest" new finding or finding in a specific year, and others. Based on the selected item and functionalities, the Update of Guideline application builds a SPARQL query. That processing include a workflow to make an analysis on the request, and obtain their annotated concepts such as UMLS/MESH terms, add additional parameters, such as date or year, then post the request to a PubMed Web server to obtain the information of relevant literature.

It is also possible to find out which trial is relevant for the selected guideline item. This relevance can be checked within the trial eligibility criteria from the Trial Registry.

The system will call the finding analysis component to make an analysis on the relevant finding information, and make a ranking on the results when it is necessary. The return

[8] The Semantic CT system is built on the LarKC platform, a platform for scalable semantic data processing (http://www.larkc.eu)

values of the selected results are displayed to the user. A screen shot of the guideline update component for the topic selection is shown in Figure 10:



Select CT: NCT01565499 **Select Ontology**: SNOMED **Select Patient**: CategoryID BC_ZSH2013A PatientNo. 1000001
Title: Neoadjuvant Treatment With Nab-paclitaxel for Patients With Stage II and III Luminal Breast Cancer
Recruitment: Not yet recruiting

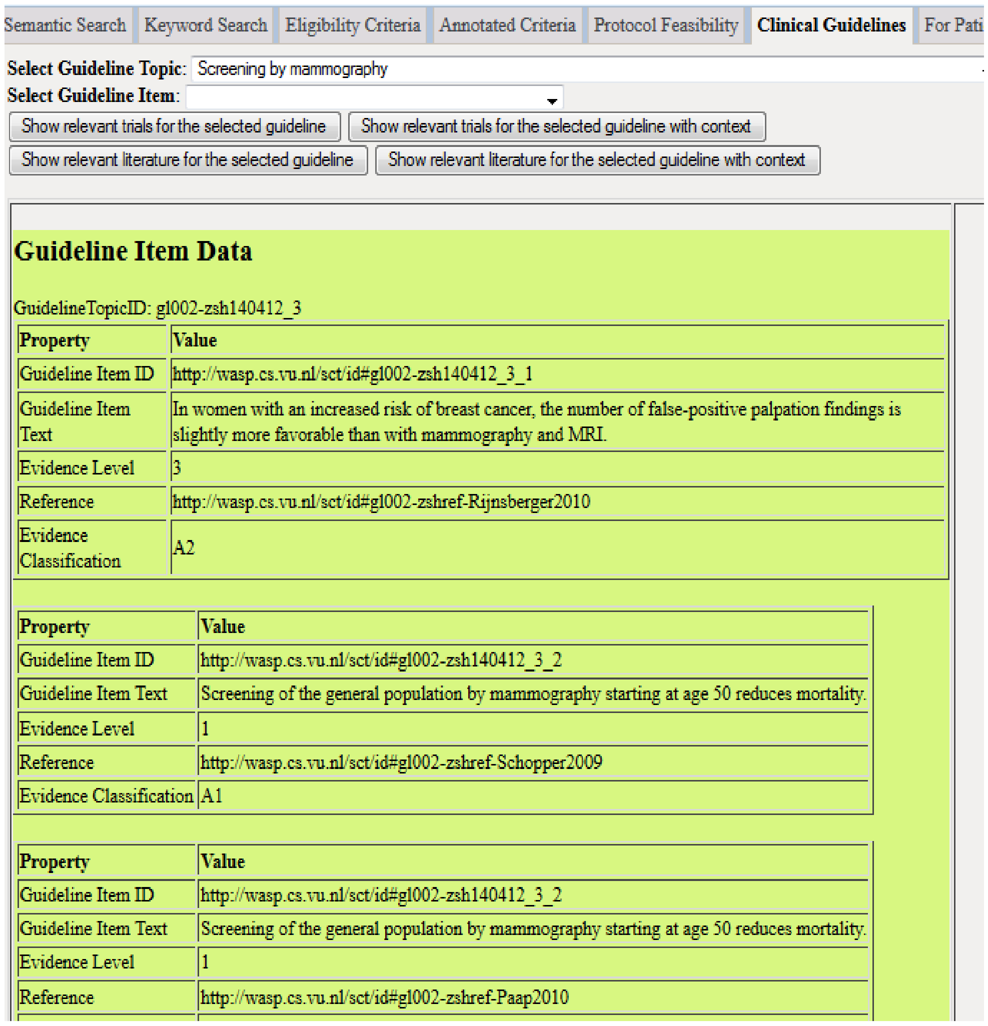
Select Guideline Topic: Regular breast self-examination as a screening method
Select Guideline Item:

Limited Result: 100

Regular breast self-examination as a screening method
Clinical breast examination as a screening method
Screening by mammography
Screening by mammography
Screening by ultrasound
Screening outside the national breast cancer screening programme
Screening for ovarian cancer
Mammography and ultrasound
Reporting in relation to the Breast Imaging Reporting and Data System (BI-RADS)
Imaging and punction of cysts
Imaging and punction of fibroadenoma
Imaging of silicone prostheses
Imaging: MRI
Differentiation between benign and malignant abnormalities/further characterisation
MRI for PA-confirmed breast cancer
preoperative MRI
For which patients is preoperative and ultrasound of the axilla indicated as triage test for 961 the SN procedure?

Figure 10: Screenshot of a guideline update component

A result for a selected topic is shown in Figure 11:


Semantic Search | Keyword Search | Eligibility Criteria | Annotated Criteria | Protocol Feasibility | Clinical Guidelines | For Pati
Show relevant trials for the selected guideline
Show relevant trials for the selected guideline with context
Show relevant literature for the selected guideline
Show relevant literature for the selected guideline with context

Select Guideline Topic: Screening by mammography
Select Guideline Item:

Guideline Item Data

GuidelineTopicID: gl002-zsh140412_3

Property	Value
Guideline Item ID	http://wasp.cs.vu.nl/sct/id#gl002-zsh140412_3_1
Guideline Item Text	In women with an increased risk of breast cancer, the number of false-positive palpation findings is slightly more favorable than with mammography and MRI.
Evidence Level	3
Reference	http://wasp.cs.vu.nl/sct/id#gl002-zshref-Rijnsberger2010
Evidence Classification	A2

Property	Value
Guideline Item ID	http://wasp.cs.vu.nl/sct/id#gl002-zsh140412_3_2
Guideline Item Text	Screening of the general population by mammography starting at age 50 reduces mortality.
Evidence Level	1
Reference	http://wasp.cs.vu.nl/sct/id#gl002-zshref-Schopper2009
Evidence Classification	A1

Property	Value
Guideline Item ID	http://wasp.cs.vu.nl/sct/id#gl002-zsh140412_3_2
Guideline Item Text	Screening of the general population by mammography starting at age 50 reduces mortality.
Evidence Level	1
Reference	http://wasp.cs.vu.nl/sct/id#gl002-zshref-Paap2010

Figure 11: Result for a selected topic

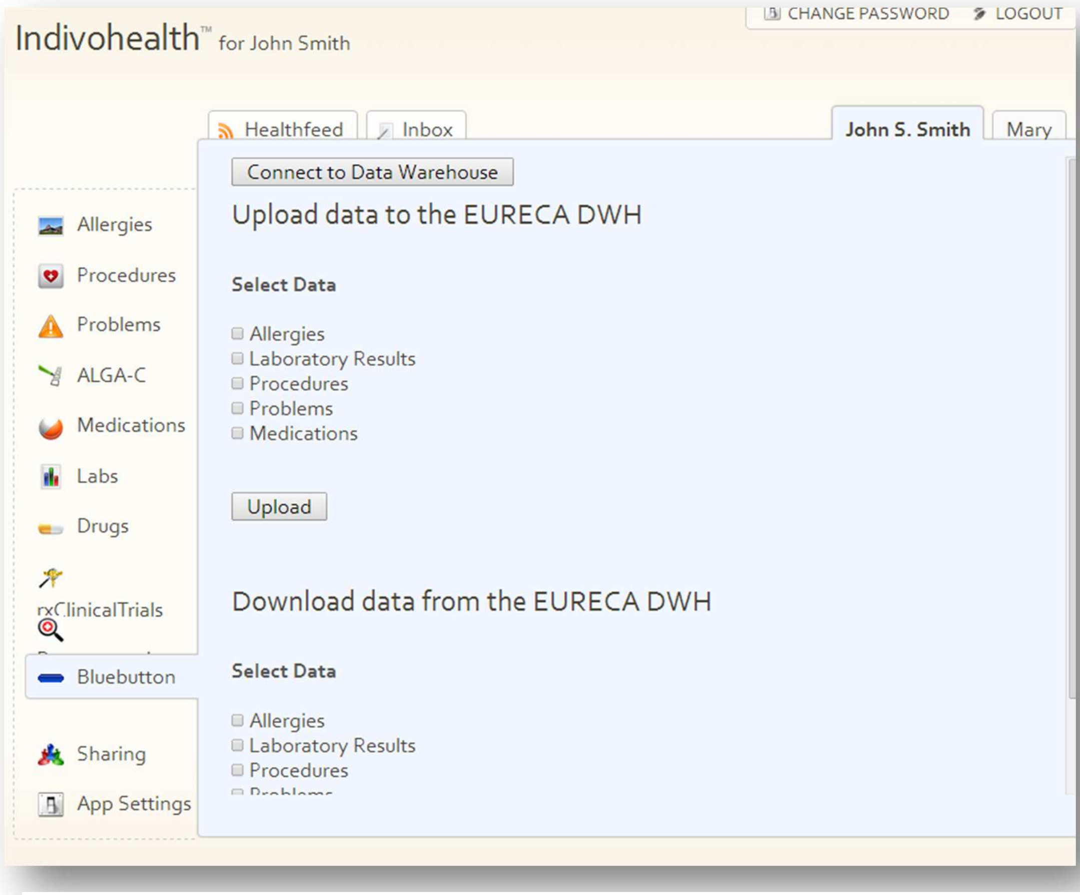
5.6 Patient Diary & Long- Term Follow-up

This scenario has to do with importing and exporting data from/to a Personal Health Record (PHR) system into/from the EURECA CDM. For this service a new application has been developed within PHR system Indivo-X[9] named “BlueButton”. Figure 12 shows the implemented import and export functionality of this application.

To be able to import and export data the specific patient in the PHR should be related to the anonymised patient stored in the EURECA CDM. The Personal Information Management System (PIMS) of the Security layer is used for the processing of anonymised patient data (see chapter 4.1). Using PIMS, we provide as input the PHR patient identifier and we get back the central identifier of the patient from the EURECA CDM.

Using this central identifier, we can proceed to import and export the patient data. Below we will describe in detail the import and export functionalities.

[9] http://indivohealth.org/



Upload data to the EUREKA DWH

Select Data

- Allergies
- Laboratory Results
- Procedures
- Problems
- Medications

Upload

Download data from the EURECA DWH

Select Data

- Allergies
- Laboratory Results
- Procedures

Figure 12: The "Bluebutton" user interface in Indivo-X

5.6.1 Download Data from a EURECA DW

To import data to the PHR from a EURECA DW we should first identify the corresponding information in the EURECA CDM. More specifically the appropriate SPARQL queries should be created for retrieving the corresponding information for the allergies, the laboratory results, the procedures the problems and the medications for a specific patient. One of these queries for retrieving the problems for a given patient and importing them into the PHR is shown in the following SPARQL query:

```
SELECT DISTINCT ?id ?code ?title ?birthTime ?effectiveTime
       WHERE {
              ?instPerson hl7rim:person_id '380b3a6f-75a4-410c-a5ed-c48141952efd'.
              ?instPerson hl7rim:person_code '337915000'.
              ?instPerson hl7rim:person_role ?instRole2. ?instRole2
              hl7rim:role_entityId  '380b3a6f-75a4-410c-a5ed-c48141952efd'.
              ?instRole2 hl7rim:role_participation ?instPart2.
              ?instPart2 hl7rim:participation_entityId
              '380b3a6f-75a4-410c-a5ed-c48141952efd'.
              ?instPart2 hl7rim:participation_act  ?instAct. ?instAct
              hl7rim:act_code ?code;
               hl7rim:act_id ?id. ?instAct
              hl7rim:act_observation ?instObs.
       OPTIONAL {
              ?instAct hl7rim:act_effectiveTime ?effectiveTime
       }
```

```
        OPTIONAL {
                ?instAct hl7rim:act_title ?title
        }
        OPTIONAL{
                ?instPerson hl7rim:person_birthTime  ?birthTime
        }
        OPTIONAL {
                ?instObs hl7rim:observation_value ?value
        }
        FILTER (!BOUND(?value))
}
```

As shown in Figure 12, the user can select the type of information that he is interested in while searching in the EURECA DW for importing or he can search for all relevant types of information. The form field is related with a SPARQL query which is sent to the Query Execution Service. The results of the query are parsed and inserted into the PHR database.

5.6.2 Upload Data to a EURECA DW

In order to push data to a EURECA DW similarly to exporting the patient, the user selects the types of data he/she wishes to import to a EURECA DW. Indivo-X exports data as JSON, XML or RDF/S. However, the Data Push Service accepts messages as HL7v3 messages. Therefore, another transformation service has been implementing transforming JSON Indivo-X messages into HL7v3 messages. Those messages are then being sent to the Data Push Service and are stored in a EURECA DW.

5.7 Microbiology Safety Service

The Microbiology Safety Service (MSS) is a service that a clinician can use to get fast knowledge and analyses about antibiotic treatments, specific infectious agents, their resistance profile and possible serious side effects. This functionality can be realised, because different clinical data sources can be merged and stored in a uniform way through the features of the EURECA platform. A detailed description about the MSS can be found in deliverable 6.3 “Initial prototype of EURECA Safety Services based on EHR data”. [6]

The Data Push Service pushes the data sets from the different hospital systems into the (local) EURECA DW automatically where they are stored in a uniform way through the EURECA CDM (details deliverable 4.3/9.3 [3]). As this deliverable focuses on the access of data sources from the EURECA environment, the following sections focus on the description about how the MSS uses EURECA services in order to import specific data sets semantically. The data that is imported is defined by creating Case Records Forms (CRFs). Based on these forms the data is updated from the local DW. In the following we describe these two steps in more detail.

5.7.1 Defining data sets

The Microbiology Module realises the user interaction to the MSS via a graphical user interface. The Microbiology Module is integrated in the web application ObTiMA[10]. From here, the clinician defines the required microbiological parameters into specific CRFs.

[10] ObTiMA: Ontology-based Trial Management Application; www.obtima.org

Through the functionalities of ObTiMA the clinician can define the content, navigation, and layout of those CRFs by several predefined question types. As an example in Figure 13, a CRF for antibiotic treatment that has been created in ObTiMA for the Microbiology Module is shown.

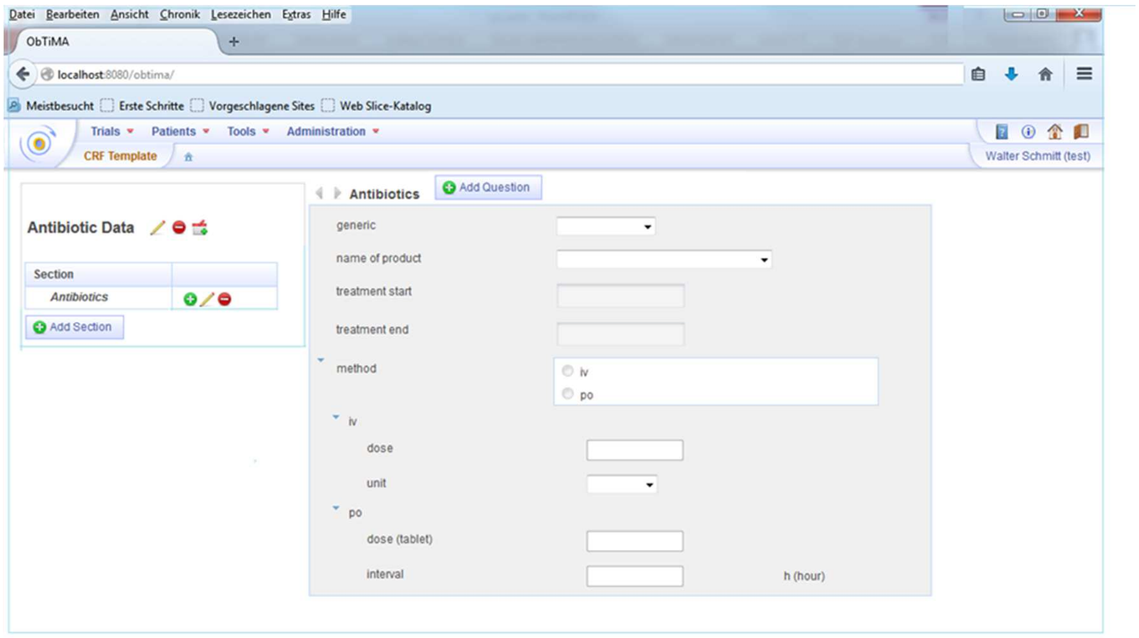


Figure 13: Creation of the Antibiotic CRF for the Microbiology Module

The defined parameters of the CRFs of the Microbiology Module are automatically pre-filled through the EURECA services (see section 5.7.2 "Importing data from local EURECA DW"). This requires the composition of specific EURECA CDM based SPARQL queries. These queries can be built with the help of the EURECA Normalization Service (see chapter 3.2.1).

The following example shows a SPARQL query that requests the parameters of the Antibiotic Treatment CRF for all available patients from the (local) EURECA DW.

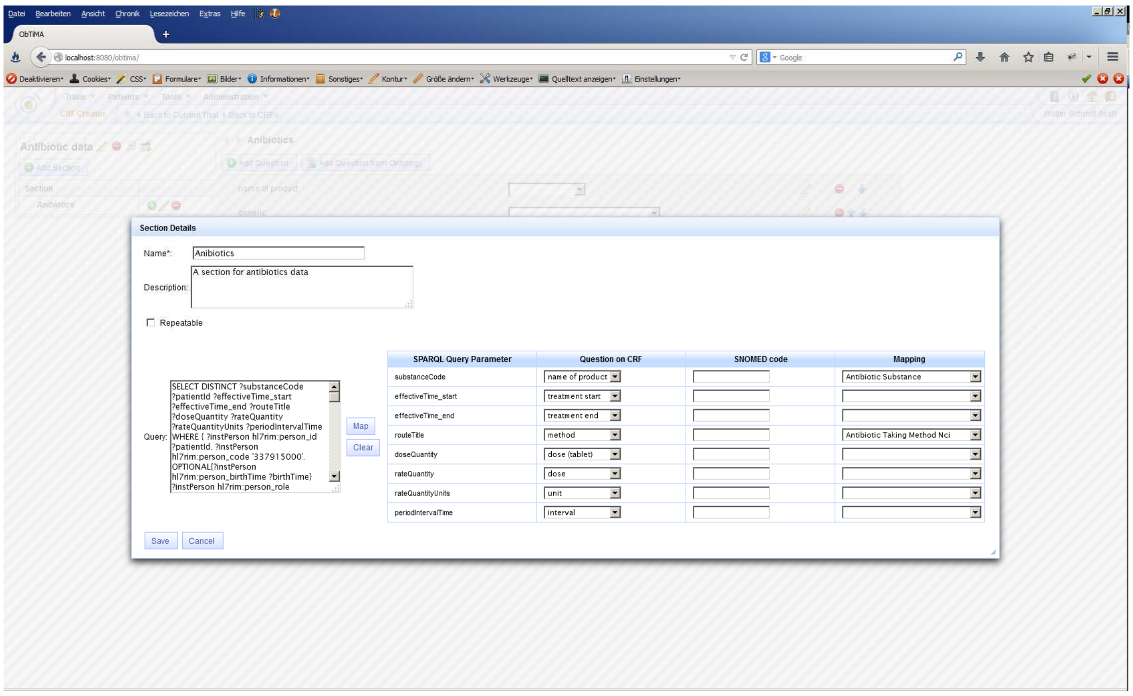
```
SELECT DISTINCT ?substanceCode ?patientId ?effectiveTime_start ?effectiveTime_end
       ?routeTitle ?doseQuantity ?rateQuantity ?rateQuantityUnits ?periodIntervalTime
       ?periodIntervalUnits
       WHERE {
              ?instPerson hl7rim:person_id ?patientId.
              ?instPerson hl7rim:person_code '337915000'.
              OPTIONAL{
                     ?instPerson hl7rim:person_birthTime ?birthTime
              }
              ?instPerson hl7rim:person_role ?instRole2. ?instRole2; hl7rim:role_en-
              tityId ?patientId.
              ?instRole2 hl7rim:role_participation ?instPart2.
              ?instPart2 hl7rim:participation_entityId ?patientId.
              ?instPart2 hl7rim:participation_act ?instAct.
              ?instAct hl7rim:act_classCode 'SBADM';
              hl7rim:act_code ?code;
              hl7rim:act_text ?text;
              hl7rim:act_codeOrig ?substanceCode;
              hl7rim:act_title ?name;
              hl7rim:act_id ?id.
```

```
                OPTIONAL{
                        ?instAct hl7rim:act_effectiveTime_start ?effectiveTime_start.
                }
                OPTIONAL{
                        ?instAct hl7rim:act_effectiveTime_end ?effectiveTime_end.
                }
                OPTIONAL{
                        ?instAct hl7rim:act_creationTime ?creationTime.
                }
                OPTIONAL{
                        ?instAct hl7rim:act_substanceAdministrationT ?instSBADM.
                        ?instSBADM a hl7rim:substanceAdministrationT
                        OPTIONAL{
                                ?instSBADM hl7rim:substanceAdministrationT_routeCode
                                ?routeCode
                        }
                        OPTIONAL {
                                ?instSBADM hl7rim:substanceAdministrationT_routeCodeVo-
                                cId ?routeCodeVocId
                        }
                        OPTIONAL {
                                ?instSBADM    hl7rim:substanceAdministrationT_routeCo-
                                deTitle ?routeTitle
                        }
                        OPTIONAL {
                                ?instSBADM    hl7rim:substanceAdministrationT_rateQuan-
                                tity ?rateQuantity
                        }
                         OPTIONAL {
                                ?instSBADM    hl7rim:substanceAdministrationT_rateQuan-
                                tityUnits ?rateQuantityUnits
                        }
                        OPTIONAL {
                                ?instSBADM    hl7rim:substanceAdministrationT_doseQuan-
                                tity ?doseQuantity
                        }
                        OPTIONAL {
                                ?instSBADM    hl7rim:substanceAdministrationT_doseQuan-
                                tityUnits ?doseQuantityUnits
                        }
                        OPTIONAL {
                                ?instSBADM    hl7rim:substanceAdministra-
                                tionT_doseCheckQuantity ?doseCheckQuantity
                        }
                        OPTIONAL {
                                ?instSBADM    hl7rim:substanceAdministra-
                                tionT_doseCheckQuantityUnits ?doseCheckQuantityUnits
                        }
                        OPTIONAL {
                                ?instSBADM    hl7rim:substanceAdministrationT_perio-
                                dIntervalTime ?periodIntervalTime
                        }
                        OPTIONAL {
                                ?instSBADM    hl7rim:substanceAdministrationT_perio-
                                dIntervalUnits ?periodIntervalUnits
                        }
                }
                FILTER (?creationTime > \"%sT00:00:00Z\"^^xsd:dateTime)
        }
```

The values of the result file (see the following example) that is sent as result by the Query Execution Service can automatically be pre-filled in the Antibiotic Treatment CRF.

```
<?xml version="1.0" encoding="UTF-8"?>
<sparql>
    <head>
        <variable name="substanceCode"/>
        <variable name="patientId"/>
        <variable name="effectiveTime_start"/>
        <variable name="effectiveTime_end"/>
        <variable name="routeTitle"/>
        <variable name="doseQuantity"/>
        <variable name="rateQuantity"/>
        <variable name="rateQuantityUnits"/>
        <variable name="periodIntervalTime"/>
        <variable name="periodIntervalUnits"/>
    </head>
    <results>
        <result>
            <binding name="substanceCode">
                <literal>7577004</literal>
            </binding>
            <binding name="patientId">
                <literal>128f463277b55c4db84ec9c826bd6ff0</literal>
            </binding>
            <binding name="effectiveTime_start">
                <literal>2012-11-22 00:00:00.0</literal>
            </binding>
            <binding name="effectiveTime_end">
                <literal>2012-11-24 00:00:00.0</literal>
            </binding>
            <binding name="routeTitle">
                <literal>ORAL</literal>
            </binding>
            <binding name="doseQuantity">
                <literal>1</literal>
            </binding>
            <binding name="rateQuantity">
                <literal>1</literal>
            </binding>
            <binding name="periodIntervalTime">
                <literal>12</literal>
            </binding>
            <binding name="periodIntervalUnits">
                <literal>h</literal>
            </binding>
        </result>
     </results>
</sparql>
```

In order to enable the automatic mapping of the result file into the CRFs, we implemented a generic mechanism that enables enter and storage of SPARQL queries for a CRF, which can be linked to single items directly. As shown for the Antibiotic Treatment CRF in Figure 14, every requested parameter of a SPARQL query can be mapped to a specific item on the CRF. The item can be defined in the column “Question on CRF” by choosing it from all available items on the CRF. Furthermore, it is possible to define an optional SNOMED code and a mapping of the answer possibilities of the item.



Section Details

Name*: Anibiotics

Description: A section for antibiotics data

☐ Repeatable

Query:

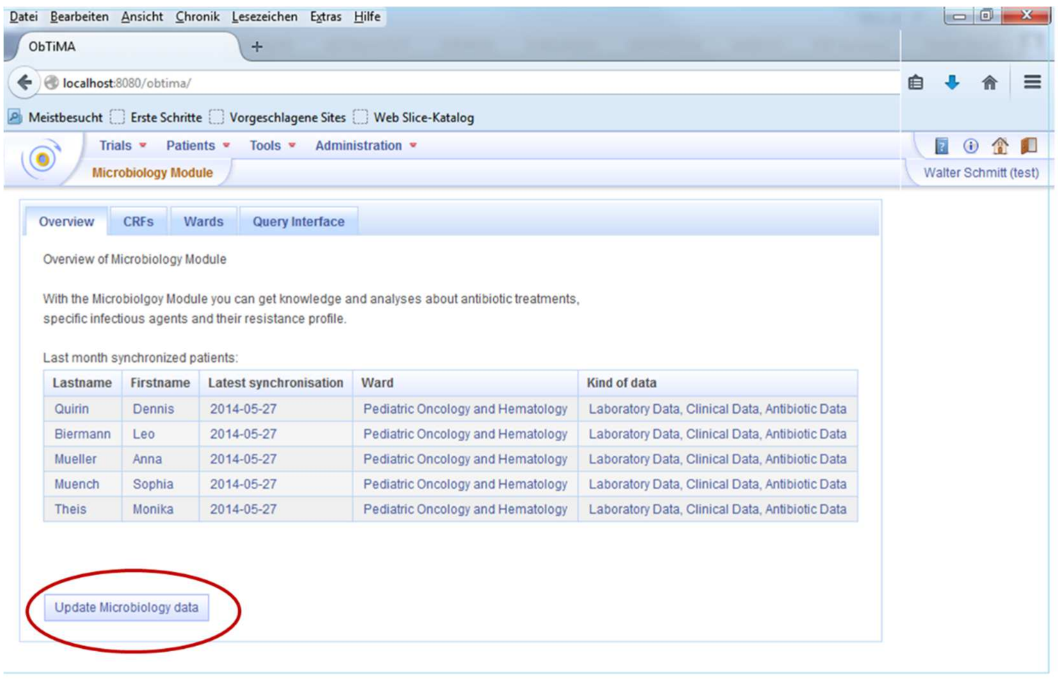
```
SELECT DISTINCT ?substanceCode
?patientId ?effectiveTime_start
?effectiveTime_end ?routeTitle
?doseQuantity ?rateQuantity
?rateQuantityUnits ?periodIntervalTime
WHERE { ?instPerson hl7rim:person_id
?patientId. ?instPerson
hl7rim:person_code '337915000'.
OPTIONAL{?instPerson
hl7rim:person_birthTime ?birthTime}
?instPerson hl7rim:person_role
```

SPARQL Query Parameter	Question on CRF	SNOMED code	Mapping
substanceCode	name of product		Antibiotic Substance
effectiveTime_start	treatment start		
effectiveTime_end	treatment end		
routeTitle	method		Antibiotic Taking Method Nci
doseQuantity	dose (tablet)		
rateQuantity	dose		
rateQuantityUnits	unit		
periodIntervalTime	interval		

Figure 14: Enter and storage of EURECA CDM based SPARQL queries in CRFs of Microbiology Module

5.7.2 Importing data from the local EURECA DW

Data from the local DW is automatically imported into the Microbiology Module by pressing the “Import Button” as shown in Figure 15. When pressing this button the DW is queried with the previously defined SPARQL queries.



Overview of Microbiology Module

With the Microbiolgoy Module you can get knowledge and analyses about antibiotic treatments, specific infectious agents and their resistance profile.

Last month synchronized patients:

Lastname	Firstname	Latest synchronisation	Ward	Kind of data
Quirin	Dennis	2014-05-27	Pediatric Oncology and Hematology	Laboratory Data, Clinical Data, Antibiotic Data
Biermann	Leo	2014-05-27	Pediatric Oncology and Hematology	Laboratory Data, Clinical Data, Antibiotic Data
Mueller	Anna	2014-05-27	Pediatric Oncology and Hematology	Laboratory Data, Clinical Data, Antibiotic Data
Muench	Sophia	2014-05-27	Pediatric Oncology and Hematology	Laboratory Data, Clinical Data, Antibiotic Data
Theis	Monika	2014-05-27	Pediatric Oncology and Hematology	Laboratory Data, Clinical Data, Antibiotic Data

Figure 15: Initiation of Data Import from the EURECA CDM

The return values of these queries are processed by the Microbiology Module and merged into one local data set. All received data can be automatically pre-filled into the corresponding CRFs utilising the defined mappings (see section 5.7.1 “Defining data sets”). The processes for sending and receiving are realised through SOAP messages via WebService technologies. The security aspects, as described in chapter 4.2, have been respected.

5.8 Reporting Episodes of Febrile Neutropenia

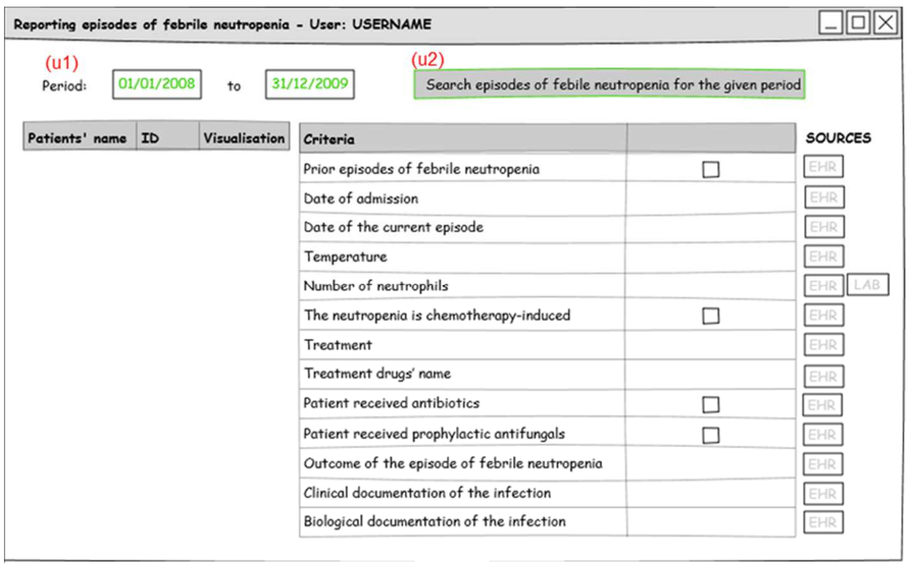
To report episodes of febrile neutropenia among a list of patients, the tool needs to access patients’ data through the EURECA CDM and be able to query it.

Febrile neutropenia is the event we would like to identify among a cohort of patients in a predetermined period. This is the conjunction the same day of neutropenia (abnormally low number of neutrophils) and fever. In clinical research, most often we are interested in chemotherapy-induced febrile neutropenia.

The Data Push Service pushes the required EHR, free text (in particular anatomical pathology, consultation, and hospital summary) data and structured data (in particular laboratory result, multidisciplinary team (MDT) summary) into the EURECA CDM (details deliverable 4.3/9.3 [3]). As other EURECA Services can also retrieve these datasets, they have been anonymised through the processes of the privacy framework (see chapter 4.1).

To work on a cohort, the user indicates in the Graphical User Interface of the Reporting Episodes of Febrile Neutropenia application the period of time (u1, u2 in *Figure 16*) for which s/he wants to find patients who have had an episode of febrile neutropenia. S/he initiates the import of data from the CDM by selecting the button "Search" (see u2 in Figure 16). The application builds a SPARQL query based on the selected period and sends it to the Query Execution Service. The Reporting Episodes of Febrile Neutropenia Service analyses and processes the return values of the Query Execution Service. This helps characterising the episode (information on diagnosis, information on the outcome, determine whether the episode is chemotherapy-induced).
For security aspects it has to be mentioned that the retrieving process of the data is carried out through the services of the Security layer (see chapter 4.2). The link to the patient ids is released through the functionalities of the Trusted Third Party (see chapter 4.1).

The user of the service should be able to validate the automatic extraction, while being able to see the data where the information (e.g. "the neutropenia is chemotherapy-induced") has been extracted from (Figure 17 and Figure 18).


Reporting episodes of febrile neutropenia - User: USERNAME

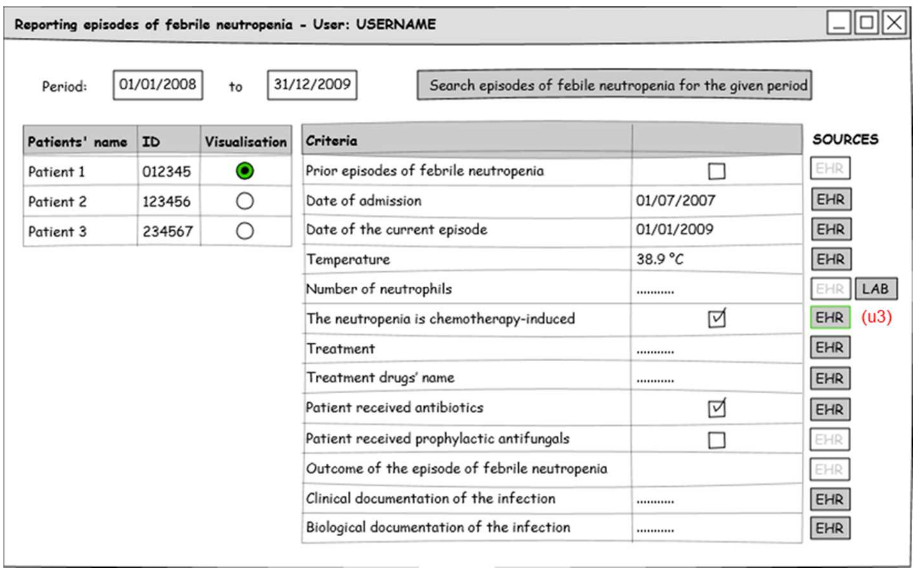
(u1)
Period: 01/01/2008 to 31/12/2009

(u2)
Search episodes of febile neutropenia for the given period

Patients' name	ID	Visualisation

Criteria		SOURCES
Prior episodes of febrile neutropenia	☐	EHR
Date of admission		EHR
Date of the current episode		EHR
Temperature		EHR
Number of neutrophils		EHR LAB
The neutropenia is chemotherapy-induced	☐	EHR
Treatment		EHR
Treatment drugs' name		EHR
Patient received antibiotics	☐	EHR
Patient received prophylactic antifungals	☐	EHR
Outcome of the episode of febrile neutropenia		EHR
Clinical documentation of the infection		EHR
Biological documentation of the infection		EHR

Figure 16: Screenshot showing the period query



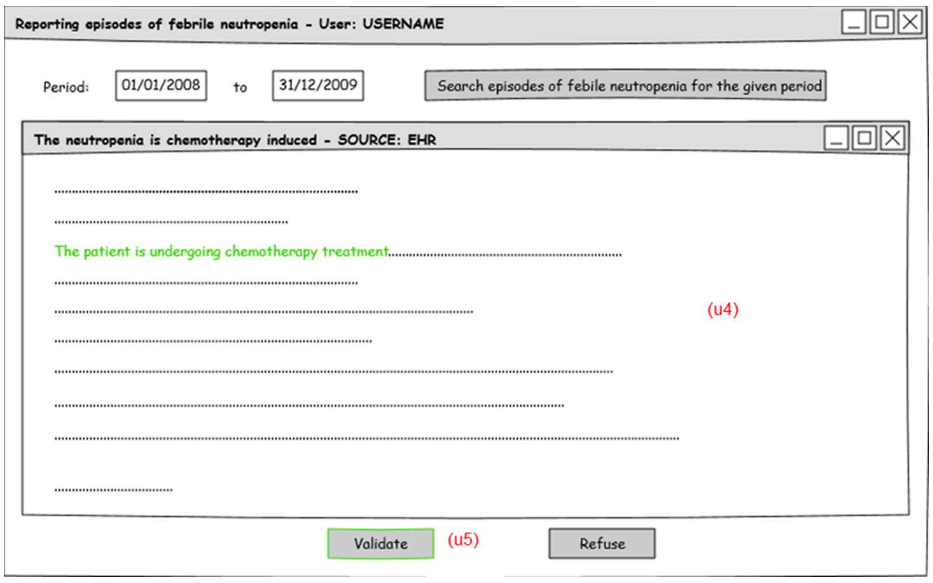
Reporting episodes of febrile neutropenia - User: USERNAME

Period: 01/01/2008 to 31/12/2009 | Search episodes of febile neutropenia for the given period

Patients' name	ID	Visualisation
Patient 1	012345	●
Patient 2	123456	○
Patient 3	234567	○

Criteria		SOURCES
Prior episodes of febrile neutropenia	☐	EHR
Date of admission	01/07/2007	EHR
Date of the current episode	01/01/2009	EHR
Temperature	38.9 °C	EHR
Number of neutrophils	....	EHR LAB
The neutropenia is chemotherapy-induced	☑	EHR (u3)
Treatment	....	EHR
Treatment drugs' name	....	EHR
Patient received antibiotics	☑	EHR
Patient received prophylactic antifungals	☐	EHR
Outcome of the episode of febrile neutropenia		EHR
Clinical documentation of the infection	....	EHR
Biological documentation of the infection	....	EHR

Figure 17: Screenshot showing data extraction from the EURECA CDM



Reporting episodes of febrile neutropenia - User: USERNAME

Period: 01/01/2008 to 31/12/2009 | Search episodes of febile neutropenia for the given period

The neutropenia is chemotherapy induced - SOURCE: EHR

....

....

The patient is undergoing chemotherapy treatment...

....

.... (u4)

....

....

....

....

....

Validate (u5) Refuse

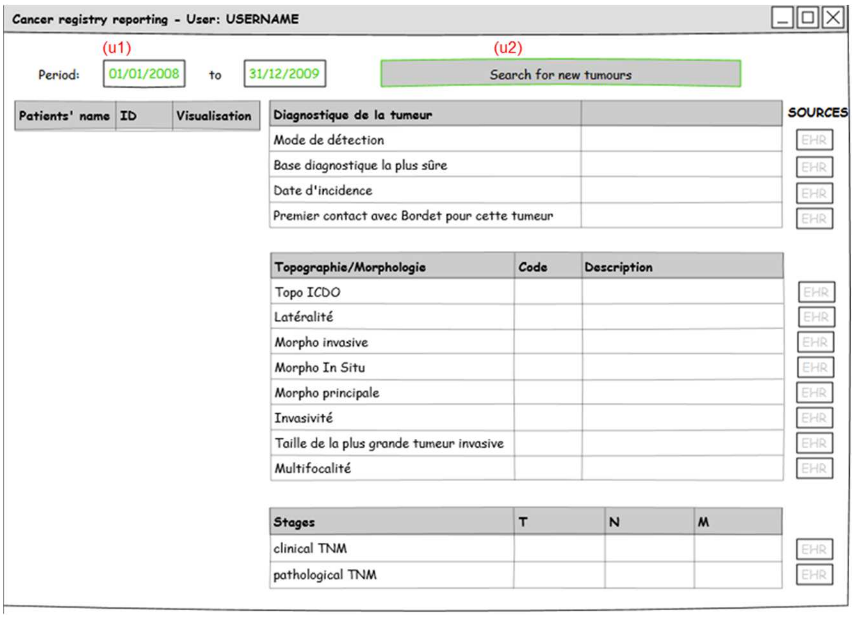
Figure 18: Screenshot showing the data where the information extracted comes from

5.9 Cancer Registry Reporting

The Cancer Registry Reporting service aims to extract and report a series of information characterising an incident tumour case (e.g. tumour detection mode, date of incidence, topography, morphology, laterality, invasiveness). The tool needs to access and query patients' data of the EURECA CDM through the Query Execution Service.

The user first needs to enter a period of time (u1, u2 in Figure 19) in which s/he wants to find all new cases of incident tumours at the Institute. The retrieving of the patient related data form the EURECA CDM is just as described for the Reporting Episodes of Febrile Neutropenia Service (see chapter 5.8).

The Cancer Registry Reporting service analyses the received data (e.g. from EHR data, anatomical pathology data, laboratory data) form the CDM and processes it to extract the information of interest (Figure 20), that will then be verified by data managers in charge of the cancer registry at the institute.



Cancer registry reporting - User: USERNAME

(u1) (u2)

Period: 01/01/2008 to 31/12/2009 Search for new tumours

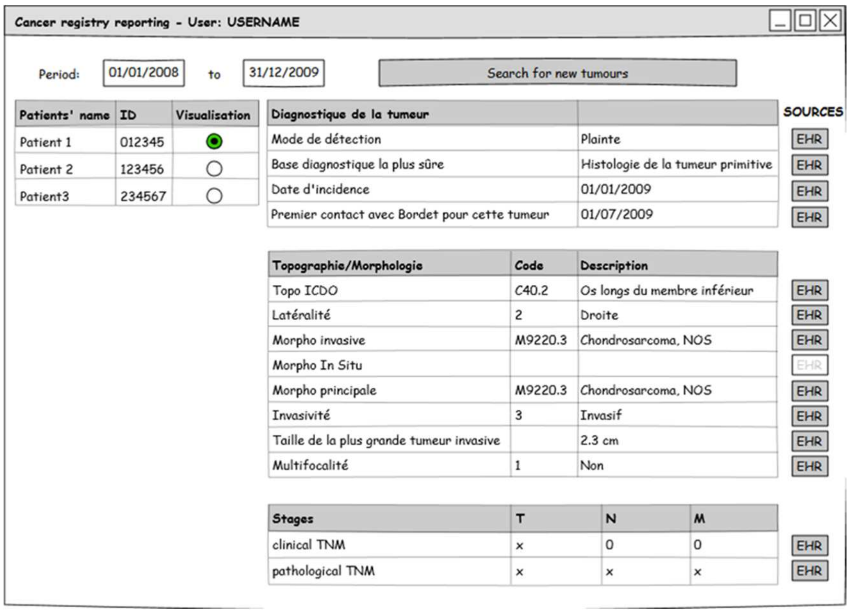
Patients' name	ID	Visualisation

Diagnostique de la tumeur		SOURCES
Mode de détection		EHR
Base diagnostique la plus sûre		EHR
Date d'incidence		EHR
Premier contact avec Bordet pour cette tumeur		EHR

Topographie/Morphologie	Code	Description	
Topo ICDO			EHR
Latéralité			EHR
Morpho invasive			EHR
Morpho In Situ			EHR
Morpho principale			EHR
Invasivité			EHR
Taille de la plus grande tumeur invasive			EHR
Multifocalité			EHR

Stages	T	N	M	
clinical TNM				EHR
pathological TNM				EHR

Figure 19: Screenshot showing the Cancer Registry Reporting GUI



Cancer registry reporting - User: USERNAME

Period: 01/01/2008 to 31/12/2009 Search for new tumours

Patients' name	ID	Visualisation
Patient 1	012345	●
Patient 2	123456	○
Patient3	234567	○

Diagnostique de la tumeur		SOURCES
Mode de détection	Plainte	EHR
Base diagnostique la plus sûre	Histologie de la tumeur primitive	EHR
Date d'incidence	01/01/2009	EHR
Premier contact avec Bordet pour cette tumeur	01/07/2009	EHR

Topographie/Morphologie	Code	Description	
Topo ICDO	C40.2	Os longs du membre inférieur	EHR
Latéralité	2	Droite	EHR
Morpho invasive	M9220.3	Chondrosarcoma, NOS	EHR
Morpho In Situ			EHR
Morpho principale	M9220.3	Chondrosarcoma, NOS	EHR
Invasivité	3	Invasif	EHR
Taille de la plus grande tumeur invasive		2.3 cm	EHR
Multifocalité	1	Non	EHR

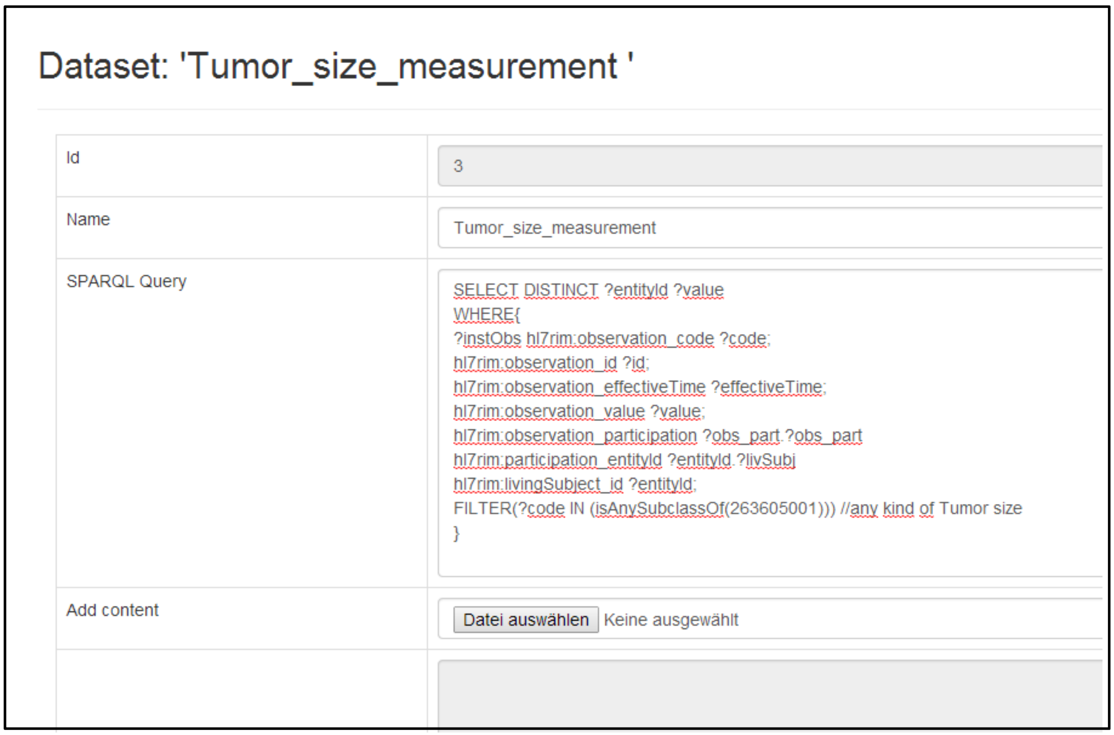
Stages	T	N	M	
clinical TNM	x	0	0	EHR
pathological TNM	x	x	x	EHR

Figure 20: Screenshot showing the Cancer Registry Reporting GUI

5.10Knowledge Discovery Framework

The EURECA Knowledge Discovery Framework (KDF) contains several data mining scripts, which can be executed over specific datasets. The user interaction with the KDF is realised by a Graphical User Interface (GUI). Details about the KDF can be found in deliverable 5.3 “Initial prototype of the generic knowledge discovery framework”. [4]
The required datasets for the KDF’s data mining processes can be acquired from the EURECA CDM. Therefore, the user has to type EURECA CDM specific SPARQL queries into the KDF’s GUI. These queries can be built with the help of the EURECA Normalization Service.

The following figure shows a SPARQL query that acquires the tumour size measurement of patients for a KDF script. This query has to be typed in manually into the KDF’s GUI (see Figure 21).



Dataset: 'Tumor_size_measurement '

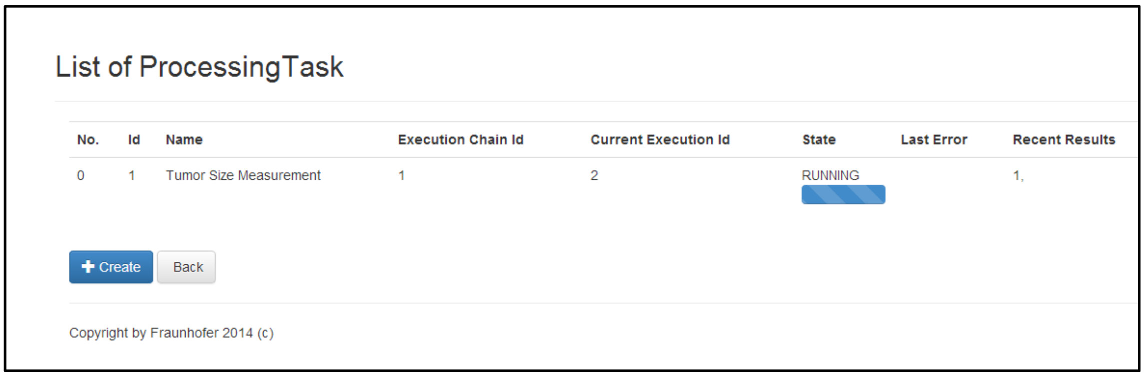
Id	3
Name	Tumor_size_measurement
SPARQL Query	(see below)
Add content	Datei auswählen Keine ausgewählt

```
SELECT DISTINCT ?entityId ?value
WHERE{
?instObs hl7rim:observation_code ?code;
hl7rim:observation_id ?id;
hl7rim:observation_effectiveTime ?effectiveTime;
hl7rim:observation_value ?value;
hl7rim:observation_participation ?obs_part.?obs_part
hl7rim:participation_entityId ?entityId.?livSubj
hl7rim:livingSubject_id ?entityId;
FILTER(?code IN (isAnySubclassOf(263605001))) //any kind of Tumor size
}
```

Figure 21: KDF GUI: EURECA CDM based SPARQL query for acquiring data for a data mining script

The KDF sends these queries through SOAP messages via WebService technologies to the EURECA Query Execution Service over services of the Security layer. The return values of the Query Execution Service compose the datasets for the data mining processes. The security aspects, as described in chapter 4.2, have been respected.

Please note, that most of the data mining processes needs data sets, which have to be retrieved by several SPARQL queries. Every retrieving processes as well as their status is shown in the KDF’s GUI (see Figure 22).



List of ProcessingTask

No.	Id	Name	Execution Chain Id	Current Execution Id	State	Last Error	Recent Results
0	1	Tumor Size Measurement	1	2	RUNNING		1,

Copyright by Fraunhofer 2014 (c)

Figure 22: KDF GUI: Status of a data retrieving for a specific SPARQL query

The following lines show the different data mining processes, which can be executed by the KDF. All data mining processes are described in detail in deliverable 5.4 “Initial services for hypothesis generation and association studies for safety risk and new research”. [5]
The GUIs of the execution of a data mining script are exemplary displayed in the description of the data mining script “Prediction of SAEs/SUSARs”.

5.10.1 Prediction of SAEs/SUSARs

For the SAE Prediction Service, there are two possible applications: The construction of a prediction model and the application of a prediction model.

For the construction of a prediction model, the SAE Prediction Service will request data from the EURECA CDM via the KDF. The user defines the requested data via a graphical user interface. The user indicates:

- for which patients data should be requested (u1 in Figure 23)
- which data fields/features should be requested for those patients (u2 and u3 in Figure 23)

The SAE Prediction Service then uses the data it has received via the data mining architecture to construct a prediction model (see Figure 24 for a screenshot of the GUI where the model is presented to the user).

For the application of a prediction model, the user indicates for which patient data should be requested (u20 in Figure 25). The SAE Prediction Service requests the required data fields/features from this patient via the KDF. It uses the received data to apply the prediction model and provide a risk score to the user (see Figure 26 for a screenshot where the risk score is presented to the user).

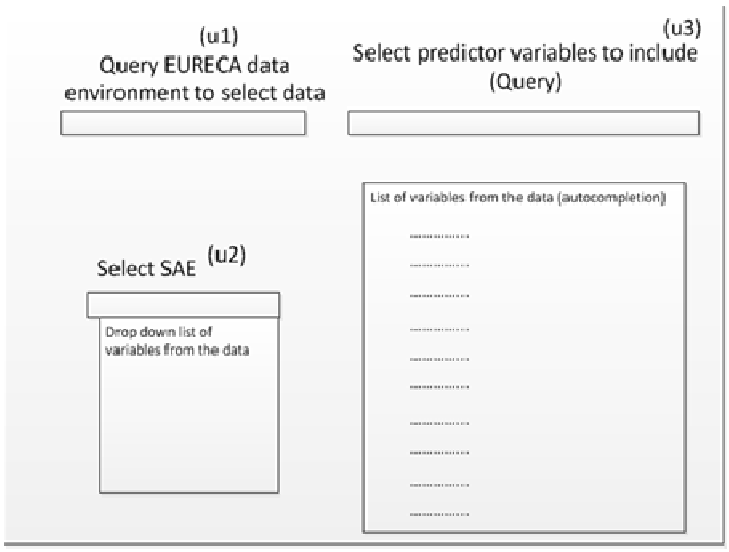

(u1)
Query EURECA data environment to select data
(u3)
Select predictor variables to include (Query)
List of variables from the data (autocompletion)
Select SAE (u2)
Drop down list of variables from the data

Figure 23: Screenshot for input of data

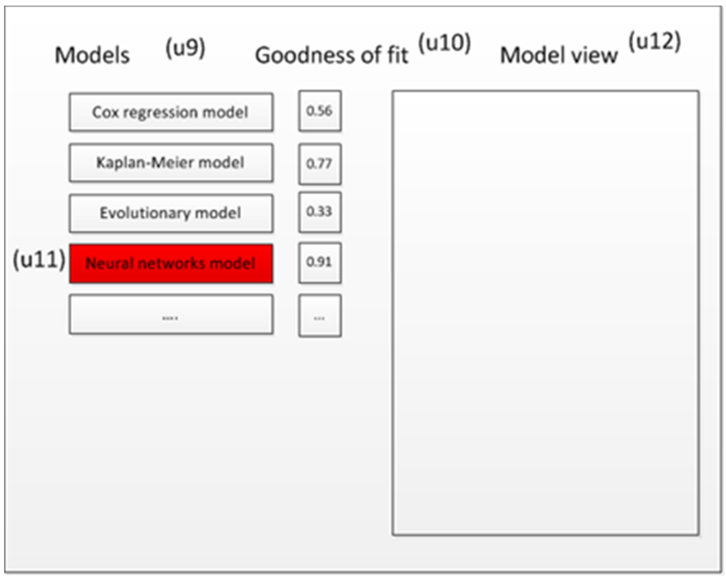


Figure 24: Screenshot showing the output of the modelling process

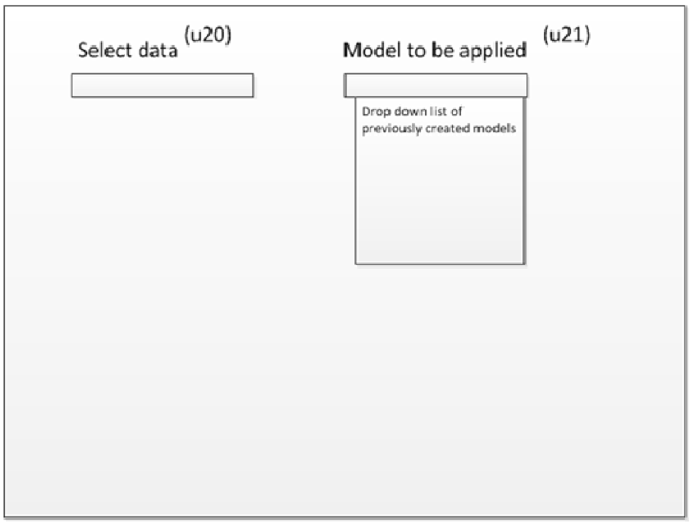


Figure 25: Screenshot for input of a new patient and the model, which should be applied to this patient

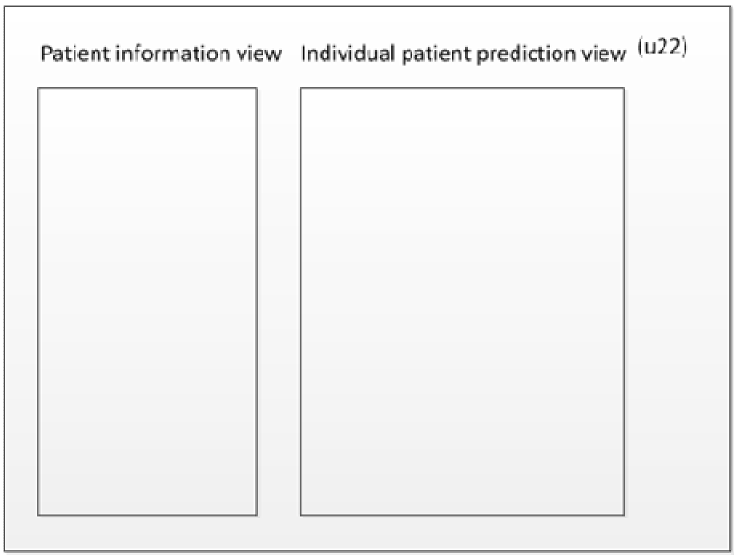


Figure 26: Screenshot showing the prediction for a new patient

5.10.2 Outcome Prediction

The focus of the Outcome Prediction process of the KDF lies on the training and application of prediction models, mainly using classification methods from the machine-learning domain. The received datasets from the EURECA CDM are thereby essential to train the outcome classifier in the first place, to update it and to validate it with independent datasets.

5.10.3 Diagnostic classifier

This data mining process focuses on the identification of diagnostic factors and subtypes of cancer patients by means of data-mining techniques such as clustering. As for the other data-mining processes, the EURECA CDM data (in particular EHR and PHR data) is accessed through the KDF, but the import of public available genomic datasets into the KDF is also needed.

5.10.4 Economic Analysis

This service is concerned with the analysis of clinical data from the economic perspective – how long do patients stay in the hospital, which patient groups are profitable, where are real or perceived quality problems that may lead to a loss of patient satisfaction? Obviously, a better access of relevant patient information may very much improve the hospital's administration ability to find good answers to improve the service quality and cost of the hospital.

From the data access viewpoint, all these scenarios follow a common pattern: the descriptive variables are all concerned with the single patient's case, which the KDF can retrieve from the EURECA CDM. For example, it can be found that patient X with diagnosis D and treatment T has been treated for W weeks. The dependent variable that is to be explained, explains the "economic value" of the case. The economic value usually depends on the cost of the treatment and the re-imbursement received, and is usually difficult to calculate. This will be realised by the Economic Analysis process.

6 Conclusion

During the EURECA project activities, a EURECA platform has been built up in order to bridge the gap between clinical care and research. EURECA specific end-user applications have been implemented in order to support research and to speed up knowledge transfer into the clinical every day work. This is in particular realised through the Semantic Interoperability Platform, which harmonises the data from different sites by the EURECA CDM. The EURECA CDM is terminology linked to the EURECA Core Dataset. In this deliverable we have described how the specific EURECA applications and services access the data through the functionalities of the EURECA platform in a uniform, secure and flexible way.

For example, we have described a service for enhancing the patient's safety in the daily clinical practice, the Microbiology Safety Service. This service integrates different data from laboratory and EHR utilizing the EURECA CDM. In this way, the detection and antibiotic treatment of resistant and multi-resistant agents becomes much more efficient.

As another example, we have described a service that enables an advanced recruitment procedure of suitable patients for a specific trial through the EURECA infrastructure, the Trial Recruitment Service. The EURECA CDM describes the corresponding patients in a uniform way. The detection of appropriate patients is realised with comparison of specific, eligibility criteria, which can be obtained from the EURECA Trial Registry.

We have shown that a variety of scenarios can profit from the data access capabilities through the integration of different data sources in a secure way. This constitutes in particular an enhancement for the clinical research and an improvement for the patients' safety.

7 Acronyms

AE	Adverse Event
CDA	Clinical Document Architecture
CRF	Clinical Record Form
CTCAE	Common Terminology Criteria for Adverse Events
EHR	Electronic Health Record
EURECA CDM	EURECA Common Data Model
EURECA DW	EURECA Data Warehouse
ETL	Extract Transform Load
GUI	Graphical User Interface
HGNC	HUGO (Human Genome Organisation) Gene Nomenclature Committee
HL7	Health Level 7 (message standard)
HL7 RIM	HL7 Reference Information Model
ICD	International Statistical Classification of Diseases and Related Health Problems
IHE	Integrating the Healthcare Enterprise
LOINC	Logical Observation Identifiers Names and Codes
MedDRA	Medical Dictionary for Regulatory Activities
MeSH	Medical Subject Headings
NCIt	National Cancer Institute Thesaurus
NLP	Natural Language Processing
PHR	Personal Health Record
RDB	Relational Databases
RDF	Resource Description Framework
R2RML	RDB to RDF Mapping Language
SAE	Serious Adverse Event
SOAP	Simple Object Access Protocol
SUSAR	Suspected Unexpected Serious Adverse Reaction
SNAQL	Snaggletooth Query Language
SNOMED	Systematized Nomenclature of Human and Veterinary Medicine
SPARQL	Simple Protocol and RDF Query Language
TTP	Trusted Third Party

XML	Extensible Markup Language

8 References

[1] Annex I – “Description of Work” of the Grant agreement for Collaborative Project for the EURECA project, 22nd August 2011
[2] Deliverable 2.2 “Initial EURECA architecture”, 01st February 2013
[3] Deliverable 4.3/9.3 “Initial proposal for the mapping formalism and mappings to EHR and CT models”, 12th April 2014
[4] Deliverable 5.3 “Initial prototype of the generic knowledge discovery framework”
[5] Deliverable 5.4 “Initial services for hypothesis generation and association studies for safety risks and new research”
[6] Deliverable 6.3 “Initial prototype of EURECA safety service based on EHR data”, 31th October 2013
[7] Deliverable 7.1 “Initial EURECA Legal and Ethical Requirements”, 31th December 2012
[8] Deliverable 7.3 “EURECA Security and Privacy Services”, 14th March 2014
[9] Freddy Priyatna, Oscar Corcho, Juan Sequeda. Formalisation and Experiences of R2RML-based SPARQL to SQL query translation using Morph. World Wide Web Conference (WWW 2014). (To Appear)

A. Appendix

A.1 “Neoplasm of female breast (disorder)” query template

```
<template version="1.00" id="002" description="Template for querying observations">
        <templateClass>Observation</templateClass>
        <rimClasses>
                <rimRelationship>
                        <rimClass>Observation</rimClass>
                        <rimAttribute>code</rimAttribute>
                </rimRelationship>
        </rimClasses>
        <inputConcept>126927001</inputConcept>
        <normalizedOutput>Observation_code : 108369006 | Observation_targetSiteCode :
91532001</normalizedOutput>
        <sparqlQuery>
                <![CDATA[
        SELECT DISTINCT ?id ?code ?patientId ?birthTime ?effectiveTime ?targetSiteCode
                ?targetSiteCodeTitle $$optionalAttributes$$
                WHERE {
                        ?instPerson    hl7rim:person_id ?patientId;
                                       hl7rim:person_code '337915000';
                                       hl7rim:person_birthTime ?birthTime;
                                       hl7rim:person_participation ?instPart2.?instPart2
                                       hl7rim:participation_act ?instAct.?instAct
                                       hl7rim:act_code ?code;
                        OPTIONAL{
                                ?instAct hl7rim:act_effectiveTime ?effectiveTime
                        }
                        FILTER (?code IN (isAnySubclassOf(108369006)))
                        ?instAct       hl7rim:act_actTargetSiteCode ?instTarget.
                        ?instTarget    hl7rim:actTargetSiteCode_code
                        ?targetSiteCode;       hl7rim:actTargetSiteCode_title ?tar-
                getSiteCodeTitle.
                FILTER (?targetSiteCode IN (isAnySubclassOf(91532001)))
                                $$OptionalStructures$$
                }
                ]]]]>><![CDATA[
        </sparqlQuery>
        <optionals>
                <OptionalStructure id="classCode">
                        <OptionalHeaderAttributes                          id="classCode"
tag="$$optionalAttributes$$">?classCode
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter id="classCode" tag="$$OptionalStructures$$">
                                <queryFilter>
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_classCode ?classCode}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="subClassCode">
                        <OptionalHeaderAttributes                       id="subClassCode"
tag="$$optionalAttributes$$">?subClassCode
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter id="subClassCode" tag="$$OptionalStructures$$">
                                <queryFilter>
```

```
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_subClassCode ?subClassCode}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="moodCode">
                        <OptionalHeaderAttributes                          id="moodCode"
tag="$$optionalAttributes$$">?moodCode
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter id="moodCode" tag="$$OptionalStructures$$">
                                <queryFilter>
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_moodCode   ?moodCode}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="codeVocId">
                        <OptionalHeaderAttributes                         id="codeVocId"
tag="$$optionalAttributes$$">?codeVocId
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter id="codeVocId" tag="$$OptionalStructures$$">
                                <queryFilter>
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_codeVocId ?codeVocId}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="codeOrig">
                        <OptionalHeaderAttributes                          id="codeOrig"
tag="$$optionalAttributes$$">?codeOrig
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter id="codeOrig" tag="$$OptionalStructures$$">
                                <queryFilter>
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_codeOrig ?codeOrig}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="codeOrigVocId">
                        <OptionalHeaderAttributes                     id="codeOrigVocId"
tag="$$optionalAttributes$$">?codeOrigVocId
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter id="codeOrigVocId" tag="$$OptionalStructures$$">
                                <queryFilter>
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_codeOrigVocId ?codeOrigVocId}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="actionNegationInd">
                        <OptionalHeaderAttributes                  id="actionNegationInd"
tag="$$optionalAttributes$$">?actionNegationInd
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter                            id="actionNegationInd"
tag="$$OptionalStructures$$">
                                <queryFilter>
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_actionNegationInd ?actionNegationInd}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="title">
                        <OptionalHeaderAttributes                              id="title"
tag="$$optionalAttributes$$">?title
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter id="title" tag="$$OptionalStructures$$">
                                <queryFilter>
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_title ?title}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="text">
                        <OptionalHeaderAttributes                               id="text"
tag="$$optionalAttributes$$">?text
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter id="text" tag="$$OptionalStructures$$">
                                <queryFilter>
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_text ?text}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="statusCode">
                        <OptionalHeaderAttributes                         id="statusCode"
tag="$$optionalAttributes$$">?statusCode
                                $$optionalAttributes$$</OptionalHeaderAttributes>
                        <OptionalFilter id="statusCode" tag="$$OptionalStructures$$">
                                <queryFilter>
                                        <![CDATA[
                                                OPTIONAL{?instAct
        hl7rim:act_statusCode ?statusCode}
                                                $$OptionalStructures$$
                                        ]]]]>><![CDATA[
                                </queryFilter>
                        </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="activityTime">
                        <OptionalHeaderAttributes                       id="activityTime"
tag="$$optionalAttributes$$">?activityTime
```

```
                              $$optionalAttributes$$</OptionalHeaderAttributes>
                       <OptionalFilter id="activityTime" tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            OPTIONAL{?instAct
       hl7rim:act_activityTime ?activityTime}
                                            $$OptionalStructures$$
                                     ]]]]>><![CDATA[
                              </queryFilter>
                       </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="availabilityTime">
                       <OptionalHeaderAttributes                 id="availabilityTime"
tag="$$optionalAttributes$$">?availabilityTime
                              $$optionalAttributes$$</OptionalHeaderAttributes>
                       <OptionalFilter                           id="availabilityTime"
tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            OPTIONAL{?instAct
       hl7rim:act_availabilityTime ?availabilityTime}
                                            $$OptionalStructures$$
                                     ]]]]>><![CDATA[
                              </queryFilter>
                       </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="uncertaintyCode">
                       <OptionalHeaderAttributes                  id="uncertaintyCode"
tag="$$optionalAttributes$$">?uncertaintyCode
                              $$optionalAttributes$$</OptionalHeaderAttributes>
                       <OptionalFilter                            id="uncertaintyCode"
tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            OPTIONAL{?instAct
       hl7rim:act_uncertaintyCode ?uncertaintyCode}
                                            $$OptionalStructures$$
                                     ]]]]>><![CDATA[
                              </queryFilter>
                       </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="valueFilter">
                       <OptionalHeaderAttributes    tag="$$optionalAttributes$$">?value
?units $$optionalAttributes$$</OptionalHeaderAttributes>
                       <OptionalFilter id="valueFilter" tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            ?instAct
       hl7rim:act_actObservationValues ?instValues.
                                            ?instValues
hl7rim:actObservationValues_value ?value;

       hl7rim:actObservationValues_units ?units.
                                            FILTER   (xsd:double(?value)   $$operator$$
$$comparedValue$$ )
                                            $$OptionalStructures$$
                                     ]]]]>><![CDATA[
                              </queryFilter>
                              <filterAttributes>
                                     <attribute                     id="operator:eq"
tag="$$operator$$">=</attribute>
```

```
                                    <attribute                          id="operator:nq"
tag="$$operator$$">!=</attribute>
                                    <attribute                          id="operator:lt"
tag="$$operator$$"><![CDATA[<]]]]>><![CDATA[</attribute>
                                    <attribute                          id="operator:gt"
tag="$$operator$$">></attribute>
                                    <attribute                        id="operator:lteq"
tag="$$operator$$"><![CDATA[<=]]]]>><![CDATA[</attribute>
                                    <attribute                        id="operator:gteq"
tag="$$operator$$">>=</attribute>
                                    <attribute                                id="value"
tag="$$comparedValue$$"></attribute>
                            </filterAttributes>
                    </OptionalFilter>
            </OptionalStructure>
            <OptionalStructure id="valueFilterOptional">
                    <OptionalHeaderAttributes tag="$$optionalAttributes$$">?control
                            ?value   ?valueType   ?units   ?refRangeMin   ?refRangeMax
?valueCode
                            ?valueCodeVocId ?valueTitle ?refRange ?fullValue
                            $$optionalAttributes$$</OptionalHeaderAttributes>
                    <OptionalFilter                            id="valueFilterOptional"
tag="$$OptionalStructures$$">
                            <queryFilter>
                                    <![CDATA[
                                     OPTIONAL{?instAct      hl7rim:act_observationAct
?instObs.
                                       ?instObs
hl7rim:observationAct_actObservationValues ?instValues.
                                       ?instValues                                     a
hl7rim:actObservationValues
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_control ?control}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_value ?value}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_valueType ?valueType}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_units ?units}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_refRangeMin ?refRangeMin}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_refRangeMax ?refRangeMax}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_code ?valueCode}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_codeVocId ?valueCodeVocId}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_title ?valueTitle}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_refRange ?refRange}
                                       OPTIONAL{?instValues
hl7rim:actObservationValues_fullValue ?fullValue}
                                     }
                                      FILTER     (xsd:double(?value)     $$operator$$
$$comparedValue$$ )
                                     $$OptionalStructures$$
                                    ]]]]>><![CDATA[
                            </queryFilter>
                            <filterAttributes>
                                    <attribute                          id="operator:eq"
tag="$$operator$$">=</attribute>
```

```
                                    <attribute                       id="operator:nq"
tag="$$operator$$">!=</attribute>
                                    <attribute                       id="operator:lt"
tag="$$operator$$"><![CDATA[<]]]]>><![CDATA[</attribute>
                                    <attribute                       id="operator:gt"
tag="$$operator$$">></attribute>
                                    <attribute                     id="operator:lteq"
tag="$$operator$$"><![CDATA[<=]]]]>><![CDATA[</attribute>
                                    <attribute                     id="operator:gteq"
tag="$$operator$$">>=</attribute>
                                    <attribute                             id="value"
tag="$$comparedValue$$"></attribute>
                            </filterAttributes>
                    </OptionalFilter>
            </OptionalStructure>
            <OptionalStructure id="interpretationCode">
                    <OptionalHeaderAttributes            id="interpretationCode"
tag="$$optionalAttributes$$">?interpretationCode
        $$optionalAttributes$$</OptionalHeaderAttributes>
                    <OptionalFilter                      id="interpretationCode"
tag="$$OptionalStructures$$">
                            <queryFilter>
                                    <![CDATA[
                                            ?instAct
        hl7rim:act_actObservationInterpretationCode ?instInterp.
                                            ?instInterp
        hl7rim:actObservationInterpretationCode_code ?interpretationCode.
                                            FILTER       (?interpretationCode       =
"%Observation_interpretationCode%")
                                            $$OptionalStructures$$
                                    ]]]]>><![CDATA[
                            </queryFilter>
                            <filterAttributes>
                                    <attribute                             id="value"
tag="%Observation_interpretationCode%"></attribute>
                            </filterAttributes>
                    </OptionalFilter>
            </OptionalStructure>
            <OptionalStructure id="interpretationCodeOptional">
                    <OptionalHeaderAttributes       id="interpretationCodeOptional"
tag="$$optionalAttributes$$">?interpretationCode           ?interpretationCodeVocId
?interpretationCodeTitle $$optionalAttributes$$</OptionalHeaderAttributes>
                    <OptionalFilter                 id="interpretationCodeOptional"
tag="$$OptionalStructures$$">
                            <queryFilter>
                                    <![CDATA[
                                            ?instAct
        hl7rim:act_observationAct ?instObs.
                                                            ?instObs
                    hl7rim:observationAct_actObservationInterpretationCode
?instInterp.
                                                            ?instInterp
        hl7rim:actObservationInterpretationCode_code ?interpretationCode}
                                                            OPTIONAL{?instInterp
        hl7rim:actObservationInterpretationCode_codeVocId ?interpretationCodeVocId}
                                                            OPTIONAL{?instInterp
        hl7rim:actObservationInterpretationCode_title ?interpretationCodeTitle}
                                            FILTER       (?interpretationCode       =
"%Observation_interpretationCode%")
                                            $$OptionalStructures$$
                                    ]]]]>><![CDATA[
                            </queryFilter>
```

```
                              <filterAttributes>
                                     <attribute                             id="value"
tag="%Observation_interpretationCode%"></attribute>
                              </filterAttributes>
                       </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="methodCode">
                       <OptionalHeaderAttributes                       id="methodCode"
tag="$$optionalAttributes$$">?methodCode                           ?methodCodeTitle
$$optionalAttributes$$</OptionalHeaderAttributes>
                       <OptionalFilter id="methodCode" tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            ?instAct
        hl7rim:act_actMethodCode ?instMethod.
                                            ?instMethod
        hl7rim:actMethodCode_code ?methodCode;

        hl7rim:actMethodCode_title ?methodCodeTitle.
                                            FILTER          (?methodCode           IN
(isAnySubclassOf(%Observation_methodCode%)))
                                            $$OptionalStructures$$
                                     ]]]]>><![CDATA[
                              </queryFilter>
                              <filterAttributes>
                                     <attribute                             id="value"
tag="%Observation_methodCode%"></attribute>
                              </filterAttributes>
                       </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="methodCodeOptional">
                       <OptionalHeaderAttributes               id="methodCodeOptional"
tag="$$optionalAttributes$$">?methodCode
                              ?methodCodeTitle ?methodCodeVocId ?methodCodeOrig
                              ?methodCodeOrigVocId
$$optionalAttributes$$</OptionalHeaderAttributes>
                       <OptionalFilter                         id="methodCodeOptional"
tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            OPTIONAL{?instAct
        hl7rim:act_observationAct ?instObs.
                                                          ?instObs
                              hl7rim:observationAct_actMethodCode ?instMethod.
                                                          ?instMethod               a
hl7rim:actMethodCode
                                                          OPTIONAL{?instMethod
                hl7rim:actMethodCode_code ?methodCode}
                                                          OPTIONAL{?instMethod
                hl7rim:actMethodCode_title ?methodCodeTitle}
                                                          OPTIONAL{?instMethod
                hl7rim:actMethodCode_codeVocId ?methodCodeVocId}
                                                          OPTIONAL{?instMethod
                hl7rim:actMethodCode_codeOrig ?methodCodeOrig}
                                                          OPTIONAL{?instMethod
                hl7rim:actMethodCode_codeOrigVocId ?methodCodeOrigVocId}
                                            }
                                            FILTER          (?methodCode           IN
(isAnySubclassOf(%Observation_methodCode%)))
                                            $$OptionalStructures$$
                                     ]]]]>><![CDATA[
                              </queryFilter>
                              <filterAttributes>
                                     <attribute                              id="value"
tag="%Observation_methodCode%"></attribute>
                              </filterAttributes>
                       </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="targetSiteCode">
                       <OptionalHeaderAttributes                    id="targetSiteCode"
tag="$$optionalAttributes$$">?targetSiteCode                       ?targetSiteCodeTitle
$$optionalAttributes$$</OptionalHeaderAttributes>
                       <OptionalFilter                              id="targetSiteCode"
tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            ?instAct
       hl7rim:act_actTargetSiteCode ?instTarget.
                                            ?instTarget
       hl7rim:actTargetSiteCode_code ?targetSiteCode;

       hl7rim:actTargetSiteCode_title ?targetSiteCodeTitle.
                                            FILTER         (?targetSiteCode         IN
(isAnySubclassOf(91532001)))
                                            $$OptionalStructures$$
                                     ]]]]>><![CDATA[
                              </queryFilter>
                              <filterAttributes>
                                     <attribute id="value" tag="91532001"></attribute>
                              </filterAttributes>
                       </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="targetSiteCodeOptional">
                       <OptionalHeaderAttributes           id="targetSiteCodeOptional"
tag="$$optionalAttributes$$">?targetSiteCode                       ?targetSiteCodeTitle
?targetSiteCodeVocId ?targetSiteCodeOrig
                              ?targetSiteCodeOrigVocId
$$optionalAttributes$$</OptionalHeaderAttributes>
                       <OptionalFilter                     id="targetSiteCodeOptional"
tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            OPTIONAL{?instAct
       hl7rim:act_observationAct ?instObs.
                                                   ?instObs
                       hl7rim:observationAct_actTargetSiteCode ?instTarget.
                                                   ?instTarget                          a
hl7rim:actTargetSiteCode
                                                   OPTIONAL{?instTarget
       hl7rim:actTargetSiteCode_code ?targetSiteCode}
                                                   OPTIONAL{?instTarget
       hl7rim:actTargetSiteCode_title ?targetSiteCodeTitle}
                                                   OPTIONAL{?instTarget
       hl7rim:actTargetSiteCode_codeVocId ?targetSiteCodeVocId}
                                                   OPTIONAL{?instTarget
       hl7rim:actTargetSiteCode_codeOrig ?targetSiteCodeOrig}
                                                   OPTIONAL{?instTarget
       hl7rim:actTargetSiteCode_codeOrigVocId ?targetSiteCodeOrigVocId}
                                            }
                                            FILTER         (?targetSiteCode         IN
(isAnySubclassOf(91532001)))
                                            $$OptionalStructures$$
                                     ]]]]>><![CDATA[
                              </queryFilter>
                              <filterAttributes>
                                     <attribute id="value" tag="91532001"></attribute>
                              </filterAttributes>
                       </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="entity">
                       <OptionalHeaderAttributes                           id="entity"
tag="$$optionalAttributes$$">
                              ?entityId     ?entityClassCode     ?entityDeterminerCode
?entityCode ?entityTitle $$optionalAttributes$$
                       </OptionalHeaderAttributes>
                       <OptionalFilter id="entity" tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            ?instAct
       hl7rim:act_participation ?instPart.
                                            ?instPart
       hl7rim:participation_entity ?instEntity.
                                            ?instEntity
       hl7rim:entity_id ?entityId;

       hl7rim:entity_classCode ?entityClassCode;

       hl7rim:entity_determinerCode ?entityDeterminerCode;

       hl7rim:entity_code ?entityCode;

       hl7rim:entity_title ?entityTitle.
                                            FILTER (?entityCode != '337915000')
                                            FILTER          (?entityCode          IN
(isAnySubclassOf(%Entity_code%)))
                                            $$OptionalStructures$$
                                     ]]]]>><![CDATA[
                              </queryFilter>
                              <filterAttributes>
                                     <attribute                            id="value"
tag="%Entity_code%"></attribute>
                              </filterAttributes>
                       </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="entityOptional">
                       <OptionalHeaderAttributes                   id="entityOptional"
tag="$$optionalAttributes$$">?roleId
                              ?partType   ?roleClassCode   ?roleCode   ?roleCodeVocId
?roleCodeOrig
                              ?roleCodeOrigVocId ?roleTitle ?entityId ?entityClassCode
                              ?entityDeterminerCode    ?entityCode    ?entityCodeVocId
?entityCodeOrig
                              ?entityCodeOrigVocId      ?entityTitle      ?entityName
$$optionalAttributes$$
                       </OptionalHeaderAttributes>
                       <OptionalFilter                             id="entityOptional"
tag="$$OptionalStructures$$">
                              <queryFilter>
                                     <![CDATA[
                                            OPTIONAL{?instAct
       hl7rim:act_participation ?instPart.
                                                   ?instPart
       hl7rim:participation_actId ?id.
                                                   ?instPart
       hl7rim:participation_roleId ?roleId.
```

```
                                            ?instPart
        hl7rim:participation_typeCode ?partType.
                                            ?instPart
        hl7rim:participation_role ?instRole.
                                            ?instRole
        hl7rim:role_id ?roleId.
                                            OPTIONAL{?instRole
        hl7rim:role_classCode ?roleClassCode.}
                                            OPTIONAL{?instRole
        hl7rim:role_code ?roleCode.}
                                            OPTIONAL{?instRole
        hl7rim:role_codeVocId ?roleCodeVocId.}
                                            OPTIONAL{?instRole
        hl7rim:role_codeOrig ?roleCodeOrig.}
                                            OPTIONAL{?instRole
        hl7rim:role_codeOrigVocId ?roleCodeOrigVocId.}
                                            OPTIONAL{?instRole
        hl7rim:role_title ?roleTitle.}
                                            ?instRole
        hl7rim:role_entityId ?entityId.
                                            ?instRole
        hl7rim:role_entity ?instEntity.
                                            ?instEntity
        hl7rim:entity_id ?entityId.
                                            OPTIONAL{?instEntity
        hl7rim:entity_classCode ?entityClassCode.}
                                            OPTIONAL{?instEntity
        hl7rim:entity_determinerCode ?entityDeterminerCode.}
                                            OPTIONAL{?instEntity
        hl7rim:entity_code ?entityCode.}
                                            OPTIONAL{?instEntity
        hl7rim:entity_codeVocId ?entityCodeVocId.}
                                            OPTIONAL{?instEntity
        hl7rim:entity_codeOrig ?entityCodeOrig.}
                                            OPTIONAL{?instEntity
        hl7rim:entity_codeOrigVocId ?entityCodeOrigVocId.}
                                            OPTIONAL{?instEntity
        hl7rim:entity_title ?entityTitle.}
                                            OPTIONAL{?instEntity
        hl7rim:entity_name ?entityName. }.
                                    }.
                                    FILTER (?entityCode != '337915000')
                                    FILTER          (?entityCode          IN
(isAnySubclassOf(%Entity_code%)))
                                    $$OptionalStructures$$
                            ]]]]>><![CDATA[
                        </queryFilter>
                        <filterAttributes>
                            <attribute                              id="value"
tag="%Entity_code%"></attribute>
                        </filterAttributes>
                    </OptionalFilter>
                </OptionalStructure>
                <OptionalStructure id="relationShip">
                    <OptionalFilter                     id="relationshipsFilter"
tag="$$OptionalStructures$$">
                        <queryFilter>
                                    <![CDATA[
                                            ?actRel  hl7rim:actRelationship_idA
?id;

        hl7rim:actRelationship_idB ?idB;
```

```
    hl7rim:actRelationship_typeCode $$comparator$$.
                            FILTER(?idB                     IN
("$$comparedValue$$"))
                            $$OptionalStructures$$
                        ]]]]>><![CDATA[
                    <!--  $$comparedValue$$  should  be  a  comma  ,
separated set of quoted 'value' -->
                </queryFilter>
                <filterAttributes>
                    <attribute             id="relationship:belongTo"
tag="$$comparator$$">belongTo
                    </attribute>
                    <!--      <attribute      id="relationship:epoch"
tag="$$comparator$$">epoch</attribute>
                        <attribute
id="relationship:plannedActivity" tag="$$comparator$$">plannedActivity</attribute> --
>
                    <attribute                              id="value"
tag="$$comparedValue$$"></attribute>
                </filterAttributes>
            </OptionalFilter>
        </OptionalStructure>
        <OptionalStructure id="patientFilter">
            <OptionalFilter id="patientFilter" tag="$$OptionalStructures$$">
                <queryFilter>
                    <![CDATA[
                        FILTER (?patientId = $$comparedValue$$)
                        $$OptionalStructures$$
                    ]]]]>><![CDATA[
                </queryFilter>
                <filterAttributes>
                    <attribute                              id="value"
tag="$$comparedValue$$"></attribute>
                </filterAttributes>
            </OptionalFilter>
        </OptionalStructure>
        <OptionalStructure id="void">
            <OptionalHeaderAttributes                        id="void"
tag="$$optionalAttributes$$"></OptionalHeaderAttributes>
            <OptionalFilter id="void" tag="$$OptionalStructures$$">
                <queryFilter></queryFilter>
            </OptionalFilter>
        </OptionalStructure>
    </optionals>
</template>
```