

ReMINE

High performances prediction, detection and monitoring
platform for patient safety risk management

FP7 Contract: 216134



– Deliverable –

**D.3.4 Third Revision Data & Process model system
framework**

Document Information

Document Name: ReMINE_D3.4_IW_WP3
Revision: V0.5
Revision Date:
Author: Adrian Podea (Info World)
Contributors: ICCS, INDRA
Security: Confidential (Consortium Only)

Document Information

Consortium, European Commission

Approvals

	Name	Beneficiary	Date	Visa
<i>Project Coordinator</i>	Michele CARENINI	NOEMALIFE		
<i>Technical Manager</i>	Fernando GUMMA	NOEMALIFE		

Document History

Revision	Date	Modification	Author
Version 0.1	21/01/2011	ToC first draft	IW
Version 0.2	01/02/2011	Chapter 3,4	IW
Version 0.3	02/02/2011	ICCS contribution	ICCS
Version 0.4	04/02/2011	INDRA contribution	INDRA
Version 0.5	04/02/2011	Conclusions and final review	IW

- 1. Executive summary..... 5
- 2. Introduction..... 5
 - 2.1. Deliverable vs. Project objectives..... 5
 - 2.2. Deliverable structure and Partners contribution 6
- 3. Architecture overview 6
- 4. WP3 tools..... 7
 - 4.1. RLUS Client tool 7
 - 4.2. TOS Client tool 12
- 5. Process Mapper 14
 - 5.1. Process Mapper for Sacco 15
 - 5.1.1. User interface 27
 - 5.2. Process Mapper for Kauhajoki..... 37
 - 5.3. Process Mapper for Niguarda..... 44
 - 5.3.1. Introduction..... 44
 - 5.3.2. Overview..... 44
 - 5.3.3. Correlation..... 46
 - 5.3.4. “MsgManager” diagram 46
 - 5.3.5. “PatientAdmission” diagram 47
 - 5.3.6. “NeurologyAssessment” diagram..... 49
 - 5.3.7. “RadOrder” diagram..... 51
 - 5.3.8. “LabExam” diagram 51
 - 5.3.9. “ECGandAdministrationOfReperfusion” diagram 54
 - 5.3.10. “ReAssessment” diagram 56
 - 5.3.11. Supporting diagrams..... 56
 - 5.3.12. Alerts list 57
 - 5.4. Process Mapper for TRFT..... 57

5.4.1.	Introduction.....	57
5.4.2.	Overview	57
5.4.3.	TRFT message processing	58
5.4.4.	Utility processes	61
5.4.5.	Processes implementing the required functionality	61
5.5.	Business rules Engine (BRE) integration	63
6.	Conclusion	66
7.	Glossary	66

1. Executive summary

Deliverable D.3.4 Third Revision Data & Process model system framework represents the last deliverable from task T3.1 – Data & Process advanced store tool. This succeeds a list of previous deliverables related to task T3.1:

- D.3.1 Data&Process model System Semantic specification and rules
- D.3.2 Classification and identification risk event using BPM process
- D.3.3 First Revision Data&Process model System framework
- I.D. Second Revision of Data&Process model System framework

This deliverable aims at briefing the modifications and development that took place from the 2nd revision of the Data&Process model framework and the last (3rd) in order to achieve the objectives listed in task T3.1: Developing a technological infrastructure where it is possible store inside the Metadatabase in a semantic format all the information that cannot be acquired directly from the WP2, thus processes that are not in any digitalized form but that need to REMINE in order to achieve the overall goal to reduce risk and increase patient safety.

Modifications explained in this deliverable can be seen as enhancements to the WP3 services and the tools used to develop those services.

2. Introduction

2.1. Deliverable vs. Project objectives

Sub-Task.3.1.1 Semantic identification – objectives:

- Identification of the semantic structure to store the entire set of data arriving from WP2 (Data Capture and RAPS alerting) and WP1 (User requirement, Patient Care processes management modeling)
- Definition and developing of the semantic API interface layer to and from the Metadatabase
- Usage and access of External data, Rules and process definition

Sub-Task.3.1.2 Information acquisition and storage (DIS) – objectives:

- Developing a tool that using BPM process allow not technical user to input and store inside the Metadatabase all the information that are in paper format, human historical process (people common and ordinary process), oral procedures, de-facto rules. All this information will be stored using a semantic metadata structure and organization.

- Developing all the API interface need to connect this module with other REMINE module in order to exchange data and process information

2.2. Deliverable structure and Partners contribution

Chapters	Responsible partner
1. Executive summary	IW
2. Introduction	IW
3. Architecture overview	IW
4. WP3 Tools	IW
5. Process mapper	ICCS, INDRA
6. Conclusions	ALL
7. Glossary	ALL

3. Architecture overview

Figure 1 represents the WP3 place in the data-flow architecture of the whole ReMINE project. WP3 consists of three different components: the Process Mapper, the Database (database service with reasoner) and Data Mining & Knowledge Inference.

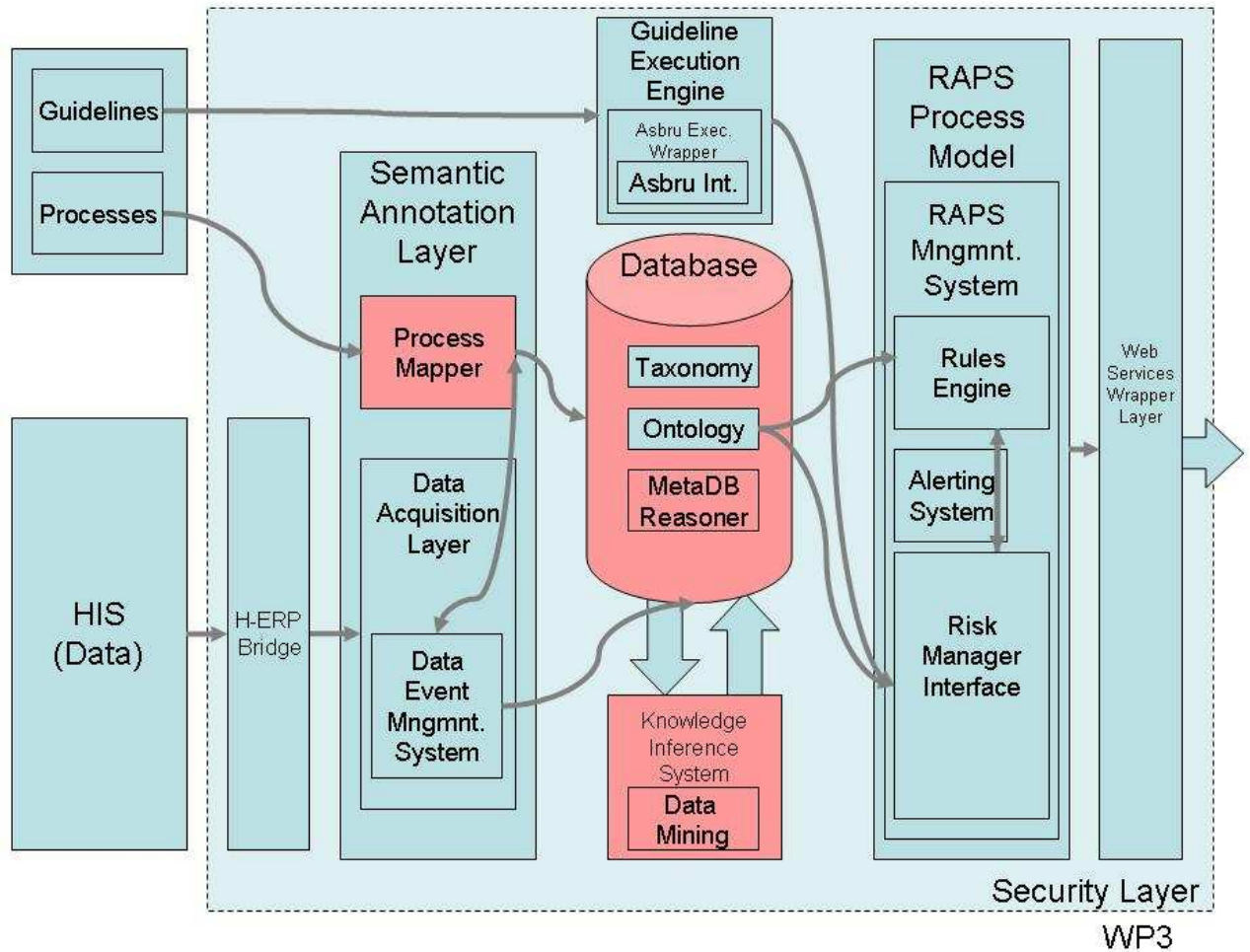


Figure 1 – The place of WP3 in ReMINE architecture

WP3 consists of the communication between the Data Acquisition Layer, the RLUS repository which is accessible through the exposed RLUS API, the additional available services (TOS and Security Service), the Metadatabase Reasoner, Master Patient Index service (MPI) used for patient management within the ReMINE project and the Data Mining and Knowledge Inference component, where the latter exposing in turn its results through an API for further utilization.

4. WP3 tools

For developing purposes and testing mechanisms the following client tools were developed. C#.Net and Java source code was provided to WP3 Services consumers for these tools to ease the development of their components. Below is a brief description of the tools with some screenshots.

4.1. RLUS Client tool

This client was developed to the all the methods on all of the RLUS Interfaces.

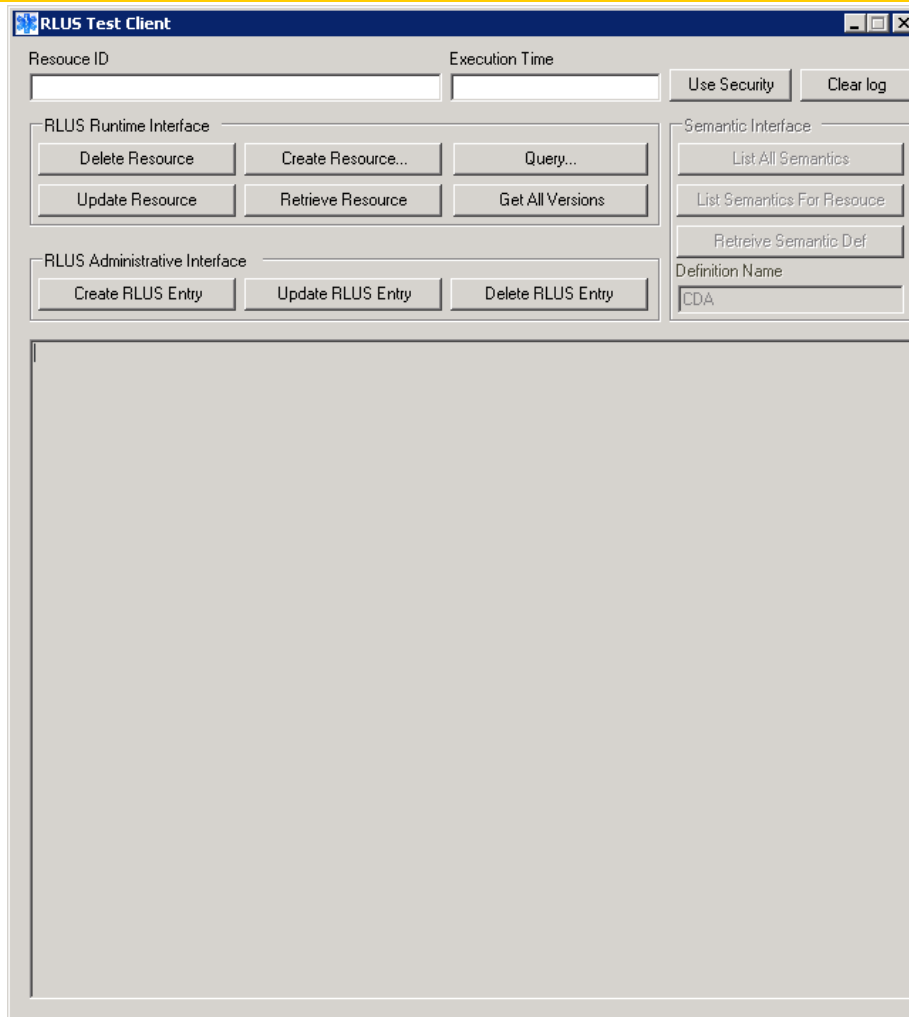


Figure 2 – .NET RLUS Main interface

The client is based on a XML serialization for data contracts and tests each method on the RLUSRuntime and RLUSAdministrative methods. Clicking each of the buttons will display an Open File Dialog box to select the XML required as the data contract for that method. Examples of data contracts have been provided in the form of:

- Entry: Data Contract ENTRY from RLUSRuntime interface
- Resources: Data Contract RESOURCE from RLUSAdministrative interface
- Query: Data Contract for queries required for both interfaces

The following methods from RLUS are tested:

- IRLUSAdministrative.CreateRLUSEntry
- IRLUSAdministrative.UpdateRLUSEntry
- IRLUSAdministrative.DeleteRLUSEntry
- IRLUSRuntime.LocateResourceByParam
- IRLUSRuntime.RetrieveResource
- IRLUSRuntime.CreateResource
- IRLUSRuntime.UpdateResource
- IRLUSRuntime.DeleteResource
- IRLUSRuntime.GetResourceVersions

Configuration of the RLU.Client is achieved through the standard WCF configuration file attached to this where connection parameters can be modified.

The RLU.Client will access any of these methods and send the XML as the data contract serialized and print the output. In the following example, a retrieve resource takes a resource id (CDA), accesses RLU through RLU Runtime interface and executes method Retrieve resource. This method returns a Base64 array of the binary data from RLU metadatabase. The client will convert this byte array in the stored string for CDAs and below it can be seen as a test CDA in the XML format.

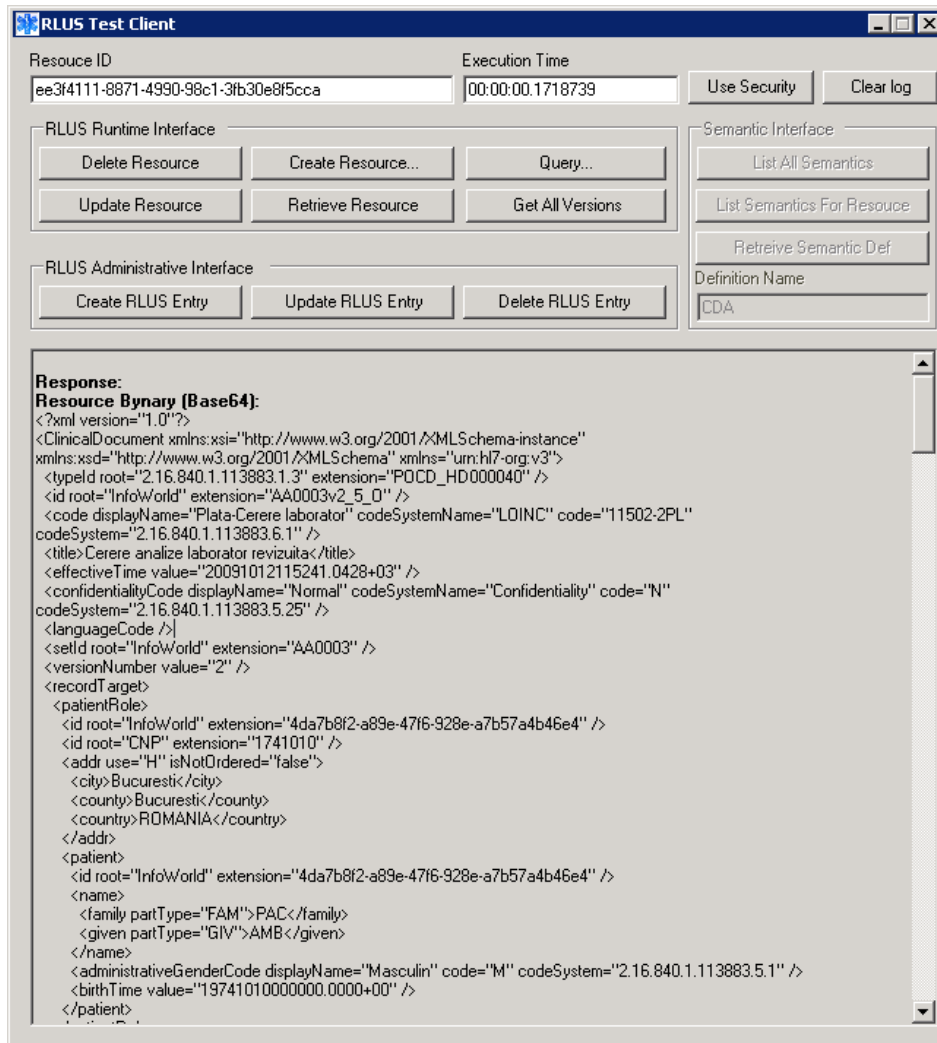


Figure 3 – .NET RLU Retrieve resource

RLU Client tool is also available in the java language, to help other partners to create their proxy and business logic easier. It offers the same functionality as the .net one including the XML data contracts for each method.

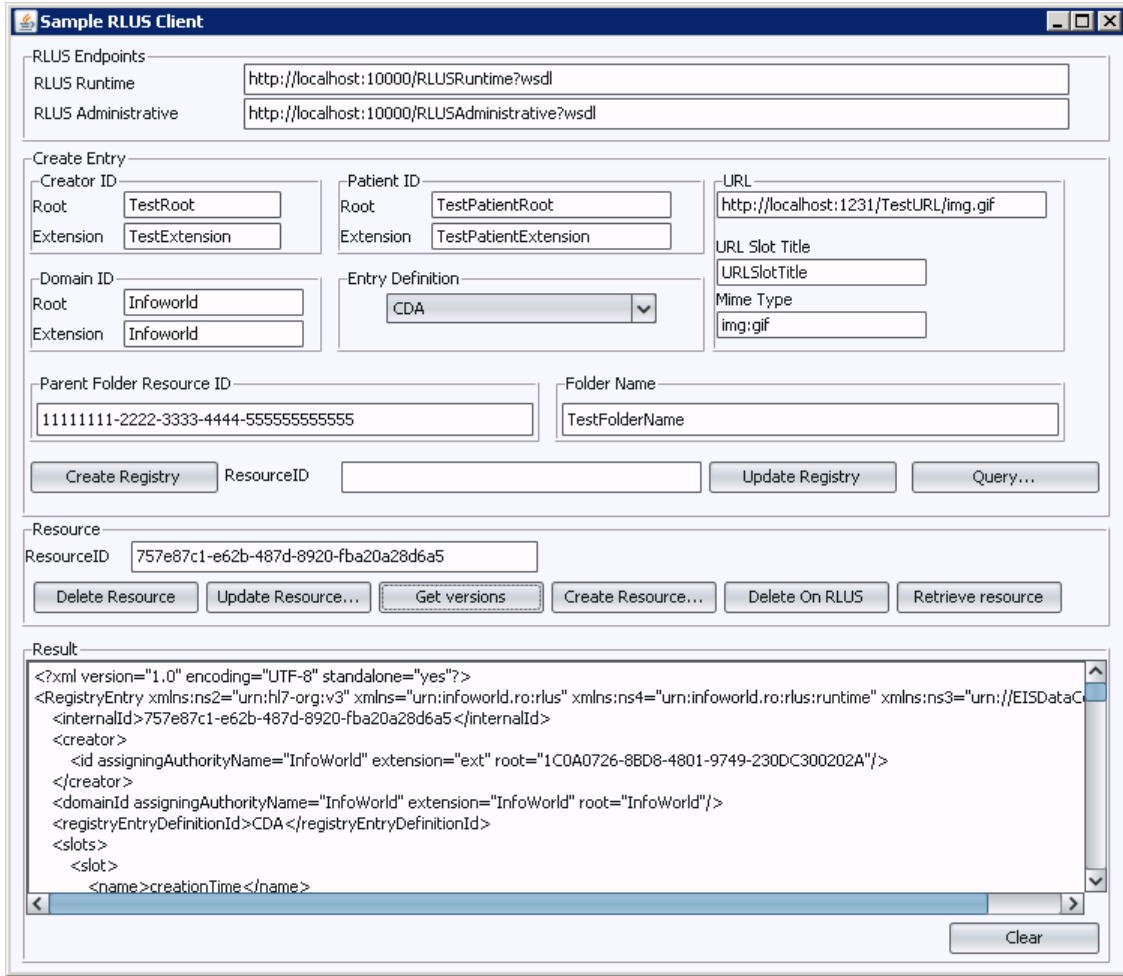


Figure 4 – .NET RLU Client Retrieve resource

Sample data contracts used in RLU client:

Query sample

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns4:LocateResourceByParam xmlns:ns2="urn:hl7-org:v3" xmlns="urn:infoworld.ro:rlus"
xmlns:ns4="urn:infoworld.ro:rlus:runtime" xmlns:ns3="urn://EISDataContracts">
  <ns4:query>
    <QueryByParameter maxResults="10" targetResourceDefinition="CDA">
      <select>
        <slot name="title"/>
      </select>
      <where>
        <slot name="creationTime">
          <filters>
            <dateTimeFilter>
              <value comparator="GT">2007-05-02T10:43:01.001Z</value>
            </dateTimeFilter>
          </filters>
        </slot>
      </where>
    </QueryByParameter>
  </ns4:query>
</ns4:LocateResourceByParam>
```

Create Resource

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

```

<rlusrun:CreateResource xmlns="urn:infoworld.ro:rlus"
  xmlns:rlusrun="urn:infoworld.ro:rlus:runtime" xmlns:sch="http://www.ascc.net/xml/schematron"
  xmlns:hl7="urn:hl7-org:v3" xmlns:mif="urn:hl7-org:v3/mif"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:infoworld.ro:rlus
D:\RLUS\src\RLUS\Schemas\Resource.xsd">
  <rlusrun:resource>
    <ResourceMetadata>
      <creator>
        <id root="1C0A0726-8BD8-4801-9749-230DC300202A"
          assigningAuthorityName="InfoWorld" extension="ext" />
      </creator>
      <domainId root="InfoWorld" extension="InfoWorld" />
      <registryEntryDefinitionId>CDA</registryEntryDefinitionId>
      <slots>
        <slot>
          <name>creationTime</name>
          <slotValues>
            <dateTime>2007-09-05T19:20:00</dateTime>
          </slotValues>
        </slot>
        <slot>
          <name>title</name>
          <slotValues>
            <string>Resource test 1</string>
          </slotValues>
        </slot>
        <slot>
          <name>firstVersionCreationTime</name>
          <slotValues>
            <dateTime>2007-09-05T19:00:00</dateTime>
          </slotValues>
        </slot>
        <slot>
          <name>classCode</name>
          <slotValues>
            <cs>COND</cs>
          </slotValues>
        </slot>
        <slot>
          <name>typeCode</name>
          <slotValues>
            <cv code="195967001" codeSystem="2.16.840.1.113883.6.96"
              codeSystemName="SNOMED CT" displayName="Asthma" />
          </slotValues>
        </slot>
        <slot>
          <name>uri</name>
          <slotValues>
            <string>http://localhost/123456</string>
          </slotValues>
        </slot>
        <slot>
          <name>mimeType</name>
          <slotValues>
            <string>image:gif</string>
          </slotValues>
        </slot>
        <slot>
          <name>confidentialityCode</name>
          <slotValues>
            <cv code="N" codeSystem="2.16.840.1.113883.5.25" />
          </slotValues>
        </slot>
        <slot>
          <name>healthCareFacilityTypeCode</name>
          <slotValues>
            <cv code="GIM" codeSystem="2.16.840.1.113883.5.10588"
              displayName="General internal medicine clinic" />
          </slotValues>
        </slot>
        <slot>
          <name>patient</name>
          <slotValues>

```

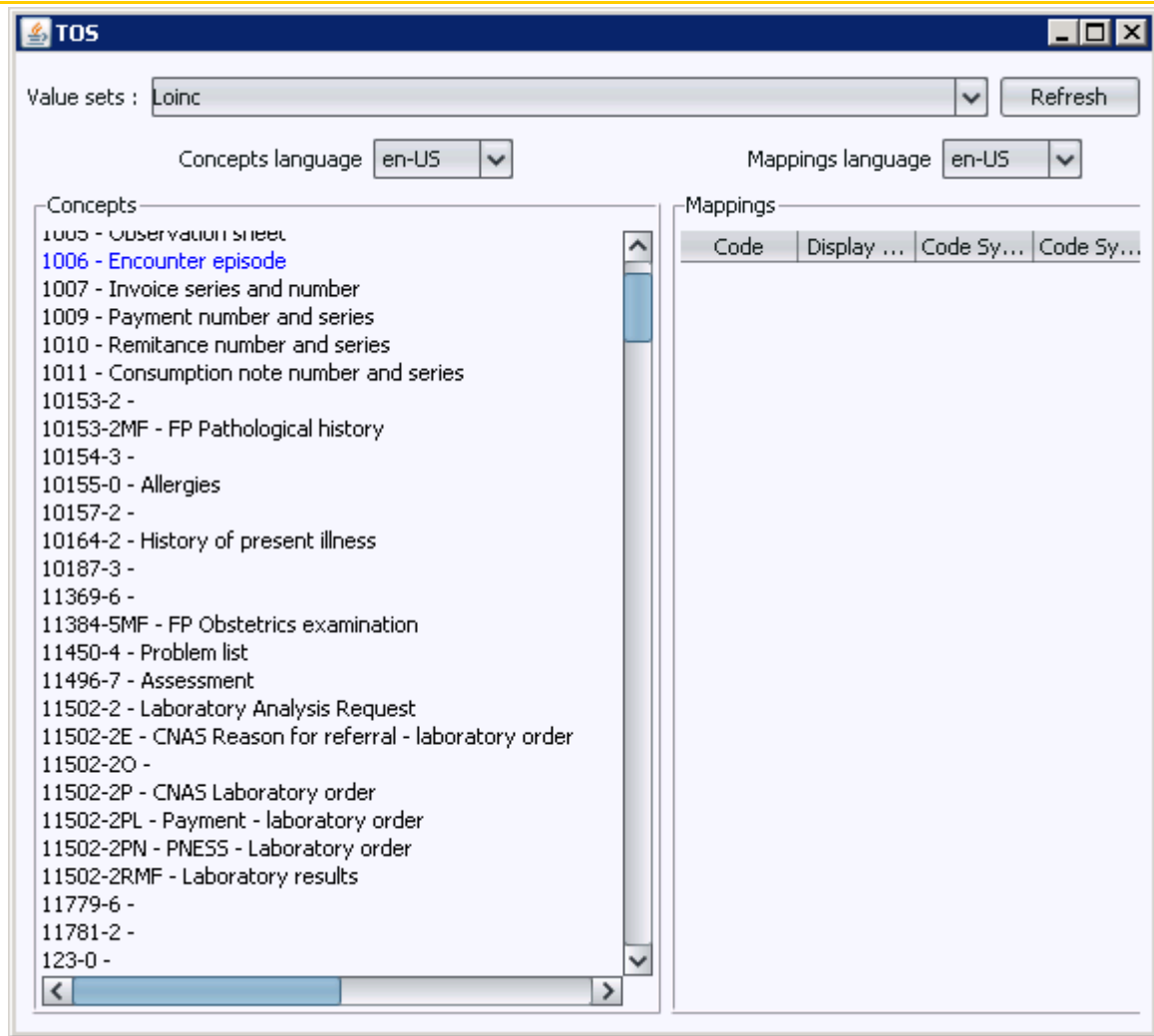



Figure 5 – Java TOS Client

In the above figure all concepts were loaded from the valueSet Loinc (a subset of the official LOINC used for ReMINE). In the .Net version a description of concepts in the right hand side of the ui can be seen as in the below figure.

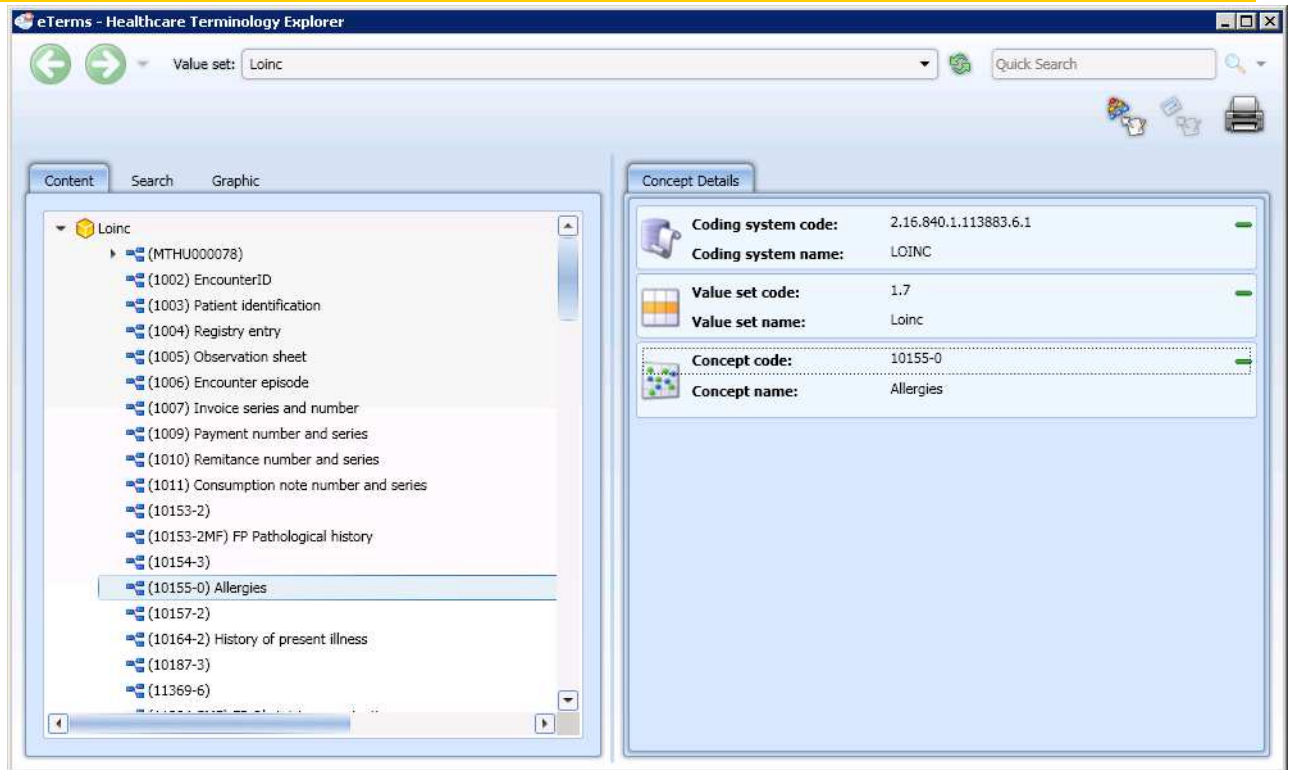


Figure 6 – .Net TOS Client

The TOS clients include a client business side to correctly access TOS for these queries. The source code is available for all partners accessing TOS in their components.

5. Process Mapper

The Process Mapper Module is responsible for allowing a not technical user to input information in the ReMINE system and receive the appropriate feedback during the execution of his/her daily work inside the hospital. This sub-module is part of the ‘Semantic Annotation Layer’ and will ‘integrate’ the healthcare organization’s processes and data via a dedicated application/tool.

This module serves two main functions; these are the initial mapping of the healthcare organization’s processes and the association of those with data and Data Events/Adverse Events. First the processes are modeled in BPMN format, then are transformed in BPEL format for being executable and be able to trigger the appropriate web services and finally are associated to data and Data Events and are enriched with KPIs (Key Performance Indicators). The process simulation (BPEL transformation) is implemented in the RAPS Process Model and these two modules are dependant to each other. This sub-module is a dedicated module for not technical users. The users via the Process Mapper will be able to insert data and receive messages. All this information will be stored using a semantic metadata structure and organization. Also according to the data inserted, this module will associate processes with Data Events and according to these having as output risk identification/prediction and alert triggering for risk management (corrective actions).

5.1. Process Mapper for Sacco

The scenario focuses on the different tasks the clinical staff can carry out for each patient during labour and delivery assistance: bracelet assignation, FHR monitoring activity, obstetrician assessment activity and epidural administration. They consider the risk for patient safety due to a delayed execution of clinical protocol, because these tasks provides medical staff with important information that enable a correct assessment and the selection of the best clinical pathway.

Besides the tasks for each patient, the framework offers a bunch of different processes to control the contextual information of the obstetrics service. In other words, the system tracks the available medical resources and the kind of activity they are carrying out at a given time.

The scenario description and the steps that were used in process modeling are described in the D7.2- Pilots Test Environment Definition. The primary users of this scenario are the midwives and obstetricians that are in charge of filling out the different forms.

The steps modeled in the Sacco process are the following:

Diagnosis of active and low risk labour

- If new woman arrives at Obstetric A&E
 - Midwife fills in the A&E report (“Isolabella”) with anagraphic data.
 - Obstetrician on duty is called.
 - Obstetrician fills in the A&E report and the REMINE forms (checklists) for admission
 - Midwife prepares the woman for the first FHR monitoring.
 - First FHR monitoring is carried out (see “FHR monitoring execution” steps).
 - Obstetrician makes the diagnosis and decides about the hospitalization.

Bracelet/Barcode assignation

- The midwife can register the barcode of the bracelet belonging to the woman
 - Midwife creates the barcode bracelet and associates it to the woman
 - Midwife applies the barcode bracelet on the woman

FHR monitoring execution

- Midwife approaches the woman with the EFM device
- Midwife prepares the woman for the monitoring

- Midwife activates the registration of the FHR monitoring by filling out the form with the following information:
 - Midwife identification
 - The EFM device identification
- When the FHR monitoring is finished, the midwife stops the registration of the FHR monitoring by filling out the proper form, with the following information:
 - Midwife identification

Obstetrician Assessment

- Obstetrician registers the results of the assessment into the “obstetrician assessment” form.

Epidural Administration

- Each time the administration of the epidural to the woman is needed
 - The obstetrician will indicate it through the “epidural” form
 - ReMINE registers that the epidural administration was carried out

“Exit”

- If the baby is born or if woman is discharged or transferred to another ward
 - Obstetrician has to fill in the REMINE “Exit” form
 - REMINE registers that the specific woman left the clinical pathway.

For the Sacco main process, the one in charge to register and monitor the patient activity, three BPM Intalio Projects have been created:

- Sacco Project, is the core of the scenario and the one to manage the business part .
- Database Manager Project, to manage the data
- RLUS Integration Project, for communicate Sacco Project with DAL/DEMS message

It is not necessary to separate them but in this way the diagrams are more clear and easier to understand and even to model the process is less complex.

Besides this main process to manage the patients, six processes to handle all the contextual information have been developed:

- Rooms, to manage the number of delivery and labour rooms available in the hospital
- Device Management, to track the usage of the different devices
- Midwife Management, to control the available midwives at a given time
- Obstetrician Management, to control the available obstetricians at a given time

- OSS Management, to control the available OSS at a given time
- Caesarean Section Management, to monitor the different obstetricians carrying out a caesarean section

The Sacco Project is going to be described in this section. The Database Manager Project and the RLUS Integration Project are services for providing data storage and DEMS/DAL communication to the main project, thus we will focus on Sacco Project.

The projects handling the contextual information will be treated in the next section (user interface), for explaining their functionality.

About the Sacco Project, it contains five BPDs, four of them are templates that are using by the main one, the main one is called Sacco.bpm. In the next screenshot the five diagrams are shown.

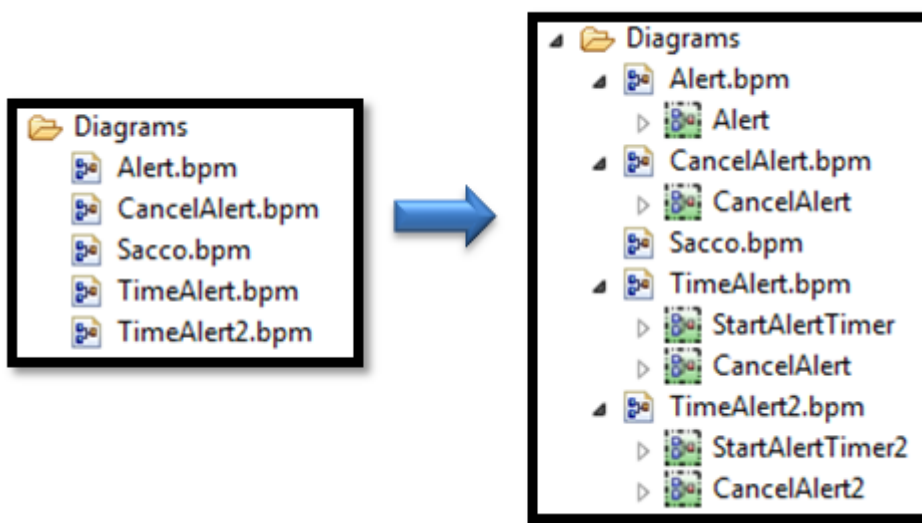


Figure 7 – Process Explorer for Sacco Diagrams

Sacco BPD

Sacco BPD has seven pools inside:

- DEMS: is the pool that starts the process sending a message with the patient information for the admission and the rest of the tasks.
- User: This pool is for the user interaction with the system. All the forms and information like notifications shown to the user are placed in this pool. The users are the obstetricians and the midwives.
- REMINE: this pool represents the system. It is the responsible of all the business logic of the process. This is the only executable pool within this diagram.
- TEMPO: contains all the calls to the web services provided by the TEMPO interface, developed by Intalio to manage their components.

D.3.4 Third Revision Data & Process model system framework

- Internal Database: is the pool that sends the orders to the internal database of the process. Contains all the calls to the database
- BRE: Contains the calls to the services that provide access to the different tables which contain the business rules.
- RLUS: Contains the calls to the web-services to handle the ReMINE metadatabase.

The diagram is as follows:

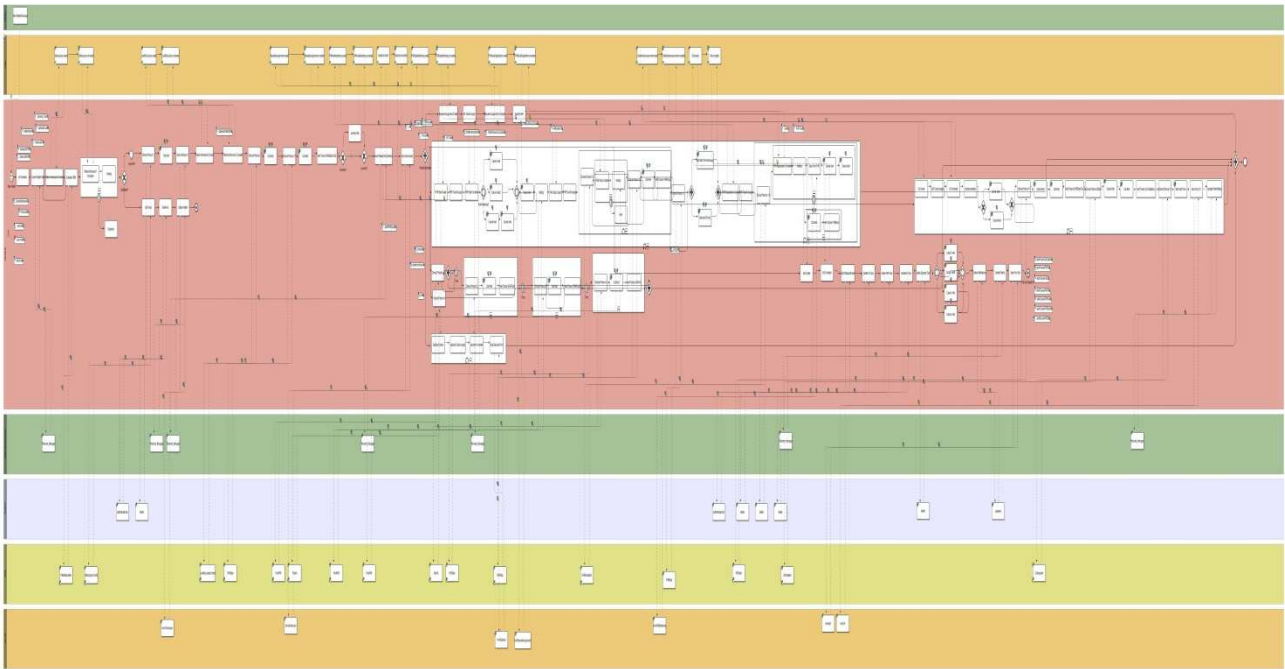


Figure 8 – Sacco BPD

Because the diagram is very big we have decided to split the diagram in some chunks in order to describe it step by step. The only pool shown in this section is the REMINE (the main one) to make it cleaner.

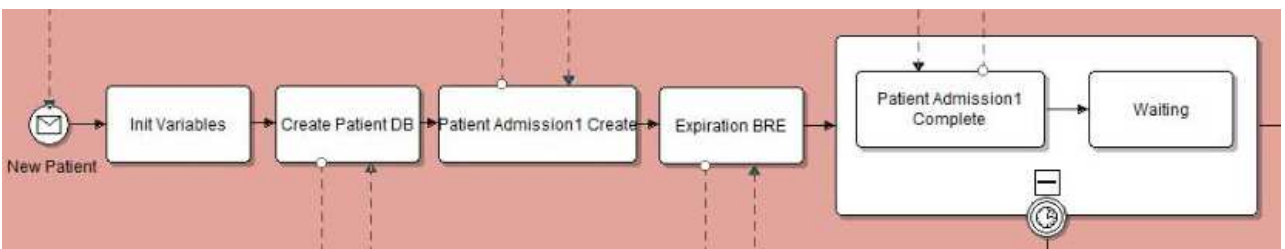


Figure 9 – Start

DEMS sends a message with the new patient and the process starts. Then the patient is stored in the internal database and the first form (Admission 1) is created. The task “Expiration BRE” calls the service to check the business rule which sets the expiration time. The expiration time is set to a timer event. This timer triggers if the first form is not completed within such time.

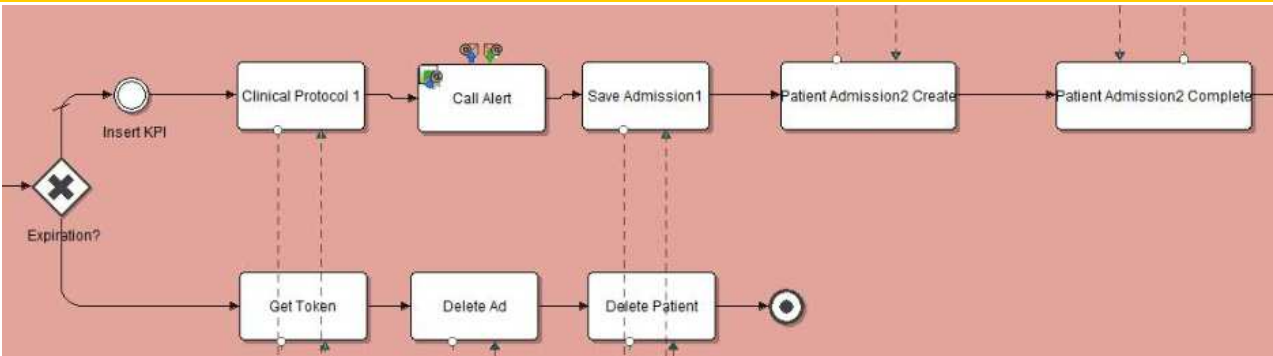


Figure 10 – Expiration time and Second form

Afterwards the process checks if the expiration time has been reached or otherwise the user has submitted the form on time.

In the first case, the process must end. Therefore deletes the first form and the patient from the database.

In the case the user has completed the first form, the process continues with the following actions:

- Checks the business rules for deciding the active labour type and sends the corresponding message
- Saves in ReMINE metadatabase (RLUS) the first form
- Creates the second admission form (Admission 2)

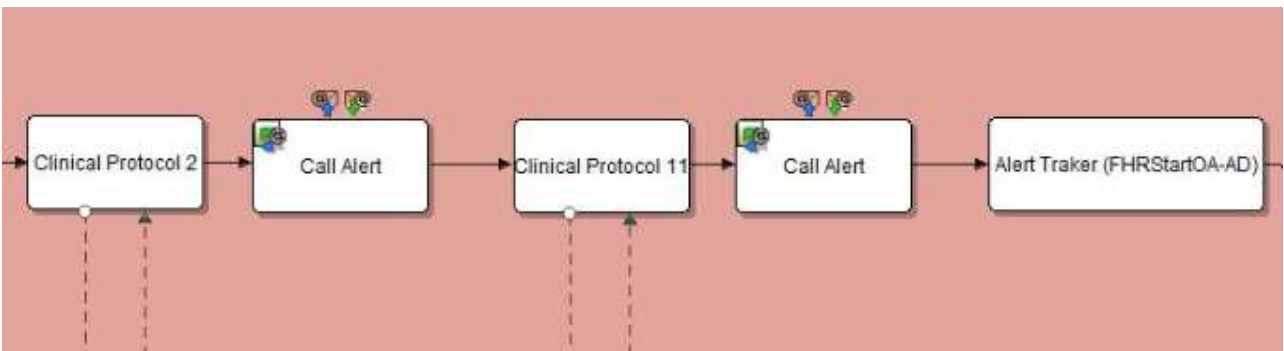


Figure 11 – Low risk notifications

Once the second admission form has been submitted, the system checks the business rules for deciding if the patient is “low risk” or not; and inform to the user with the corresponding notifications.

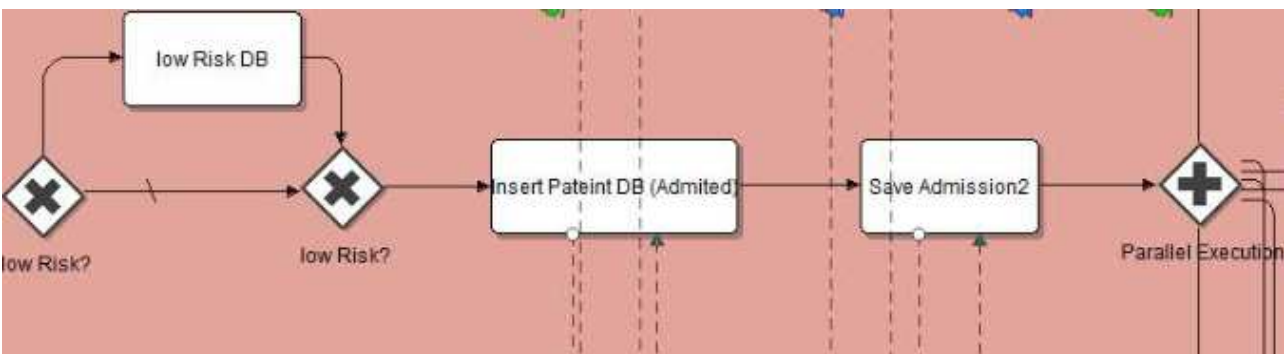


Figure 12 – Save second form

The information coming from the second form is stored in three different components:

- The kind of patient (“low risk” or not) is stored in KPI’s database for the reports
- The user is marked as “admitted” in the internal database
- All the information submitted in the second form is stored in the metadatabase (RLUS)

After the patient has been admitted (the two admission forms has been filled out), there is a parallel gateway that provokes the execution of four branches at the same time:

1. Bracelet Assignment: it creates the form for the bracelet assignment and receives the data filled in by the nurse. Once is filled out the information is saved in the ReMINE metadatabase.

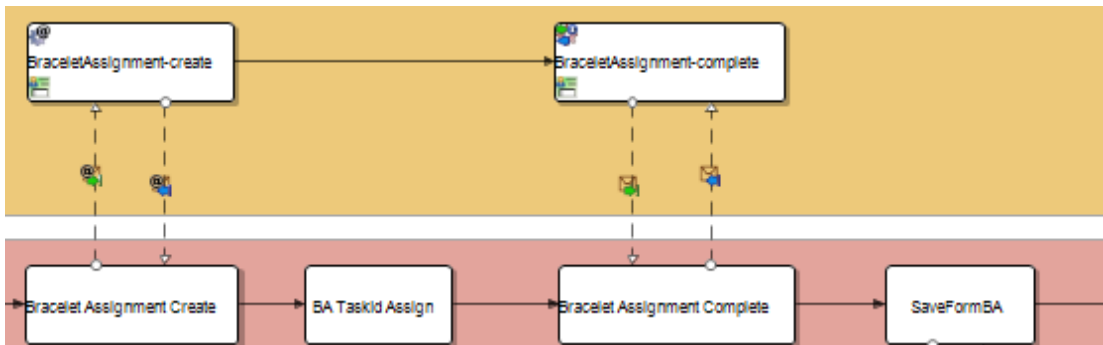


Figure 13 – Bracelet Assignment

2. FHR Monitoring: handle all the processes related with the monitorings.

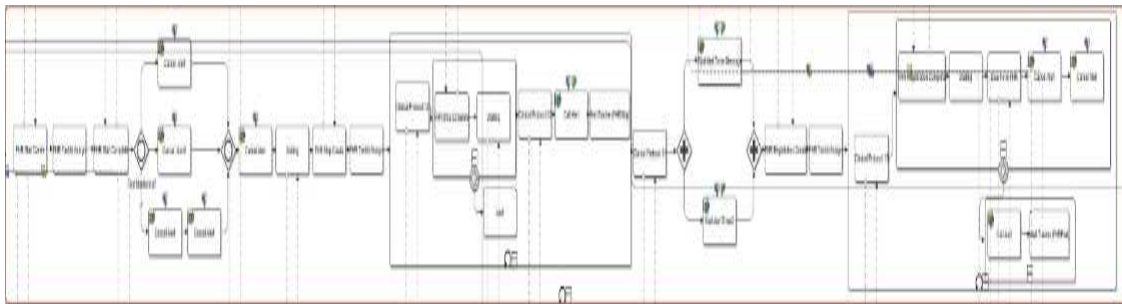


Figure 14 – FHR Monitoring

For a better explanation of how it works, it will be divided in several parts:

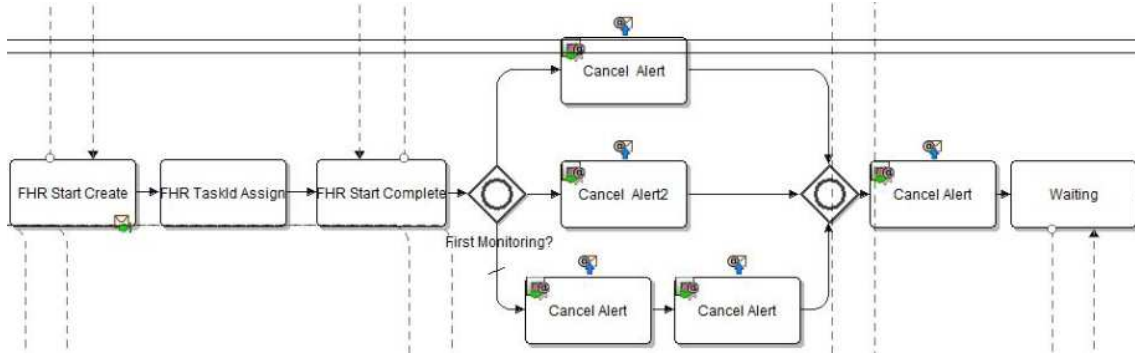


Figure 15 – FHR Monitoring: 1

2.1. The process creates the “Start FHR monitoring” form. Once is completed cancels all the alerts which are not useful anymore.

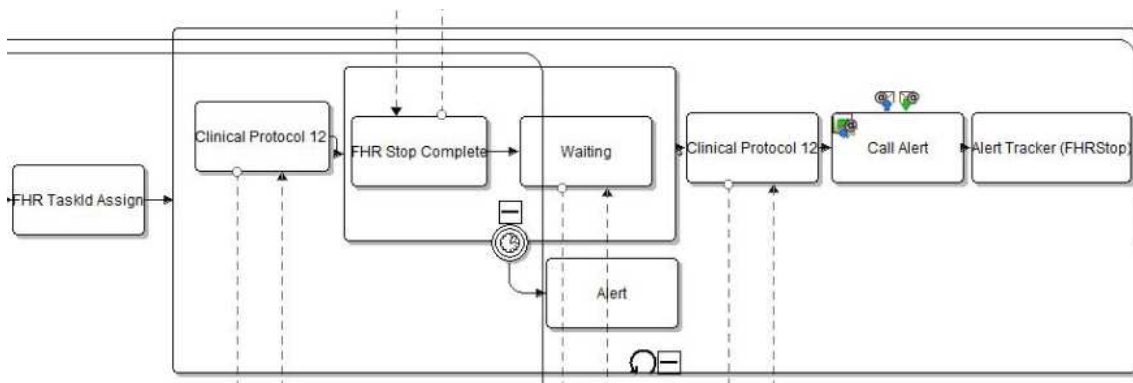


Figure 16 – FHR Monitoring: 2

2.2. After the “Start FHR monitoring” has been submitted, the system creates the “Stop FHR monitoring” form. Then with the information extracted from the forms and the whole process, the process checks the clinical protocol and sends and alert, if suitable.

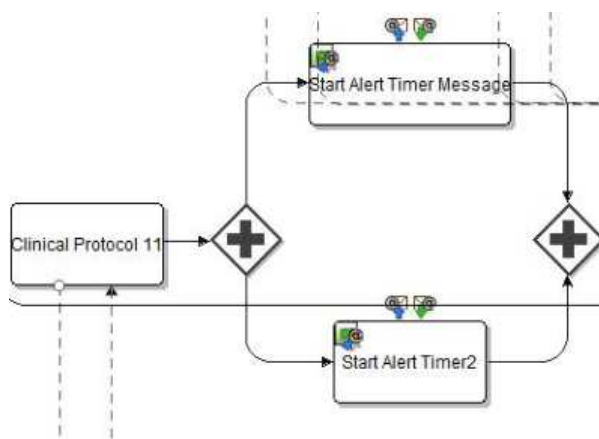


Figure 17 – FHR Monitoring: 3

2.3. The process checks one of the business rules tables, for notifying any violation of the protocol. The process gets the time requirement and starts two timers which will trigger an alert if they are not cancelled before the time has gone.

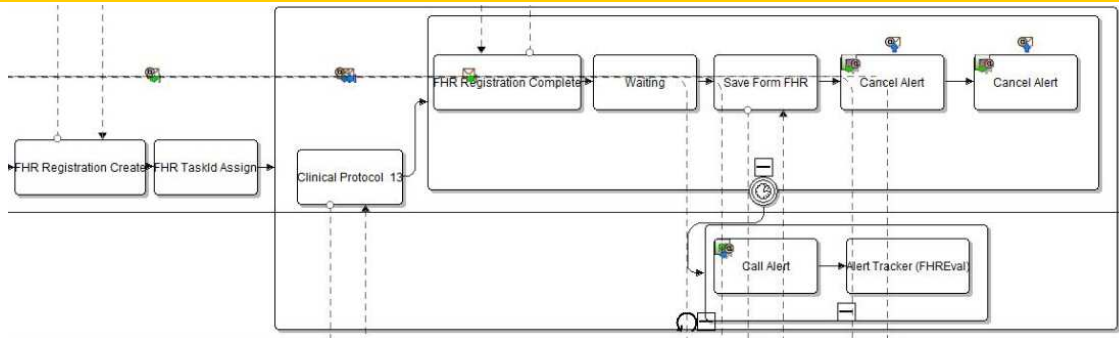


Figure 18 – FHR Monitoring: 4

2.4. After the “Stop FHR monitoring” has been submitted, the system creates the “Registration FHR monitoring” form. Then with the information extracted from the forms and the whole process, the process checks the clinical protocol and sends an alert, if suitable. Also cancels all the alerts which are not valid anymore.

2.5. Finally the system stores all the information of the monitoring in RLUS.

The FHR monitoring bunch is a looping task, therefore it will be repeated until the patient exits the scenario.

3. Obstetrician Assessment: handle all the processes related with the obstetrician assessments.



Figure 19 – Obstetrician Assessment

For a better explanation of how it works, it will be divided in several parts and explained step by step:

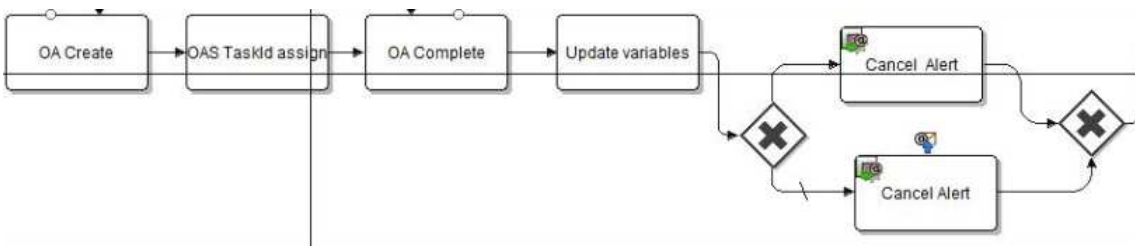


Figure 20 – Obstetrician Assessment: 1

3.1. The process creates the form, and once it is completed, it cancels two alerts not valid anymore after the user has submitted the form. One of them cancels an alert waiting to be triggered; and the other one cancels an alert that is not needed anymore (unless they have the same name, they are calling two different templates. See TimeAlert BPD and CancelAlert BPD).



Figure 21 – Obstetrician Assessment: 2

3.2. After the form is submitted, the process checks twice the business rules tables to make sure that there are any violation of the clinical protocols. Before any notification is sent to the user, the system deletes the previous ones as they are not valid anymore.

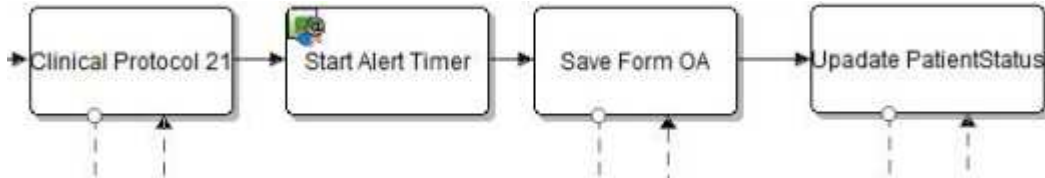


Figure 22 – Obstetrician Assessment: 3

3.3. The process checks other business rules table, for notifying any violation of the protocol. The process gets the time requirement and starts a timer which will trigger an alert if this is not cancelled before the time has gone. Finally the information of the form is saved in ReMINE metadatabase and also in the internal database.

The obstetrician assessment bunch is a looping task, therefore it will be repeated until the patient exits the scenario.

4. Epidural Administration: handle all the processes related with the epidural administration. Creates the form to be submitted when the epidural is administrated to the patient. Once is completed the information is saved in the metadatabase. It is a looping task, it will be created until the patient exits the scenario.

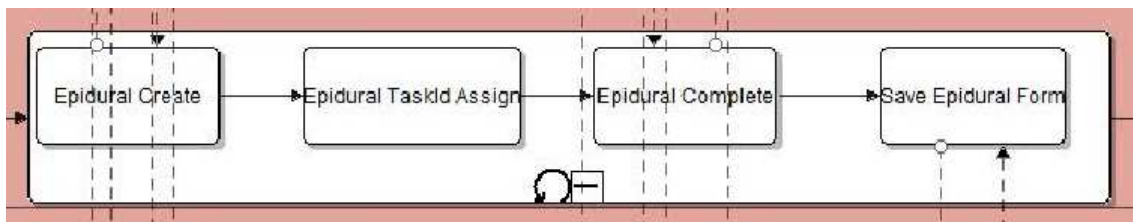


Figure 23 – Epidural Administration

5. Exit: it shows the “Exit” to be filled in by the obstetrician and inserts a number in the database to indicate that the execution for that patient is finished.

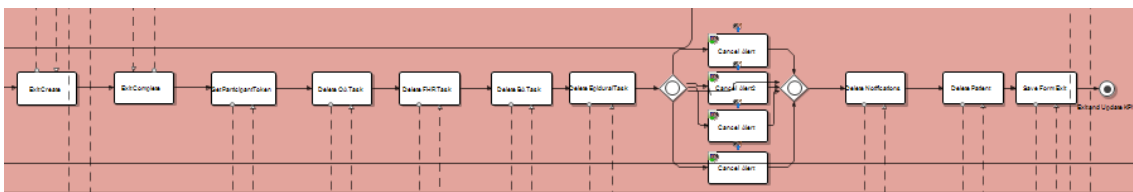


Figure 24 – Exit

We are going to divide the picture in chunks to explain them step by step:



Figure 25 – Exit: 1

5.1. Once the midwife or the obstetrician has filled out the “Exit” form, the rest of the tasks are deleted and become unavailable for the user.

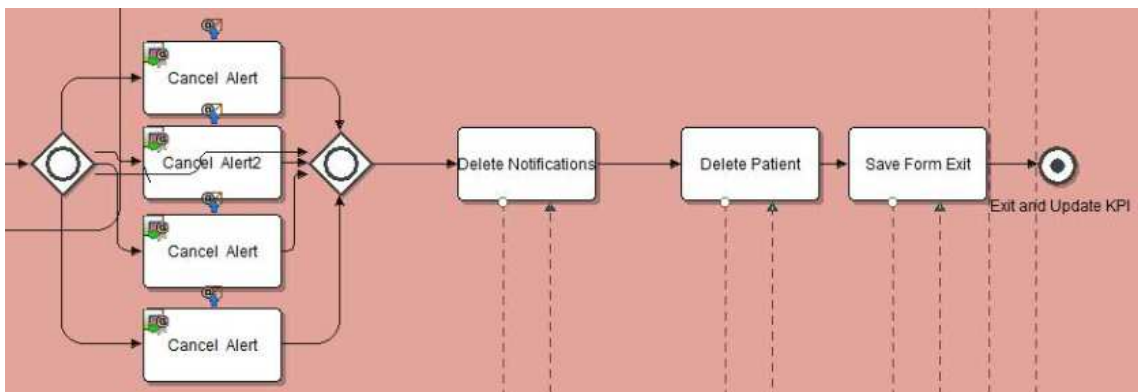


Figure 26 – Exit: 2

- 5.2. All the pending alerts (processes with timers) waiting to be trigger, are cancelled to avoid them to be sent afterwards.
- 5.3. All the notifications are deleted so they are not visible anymore for the user
- 5.4. The patient is also deleted from the internal database
- 5.5. The information of the “Exit” form is stored in the metadatabase.
- 5.6. The process terminates

Alert and CancelAlert BPD’s

Alert BPD has the following functionality:

- 1. Send the corresponding notification to the user and it is shown in "Notifications" tab
- 2. Send the corresponding notification to the AMS in case it has been mapped for it by means of a business rule table
- 3. Returns the value of the instance which identifies the notification sent to the user; and also the identification in AMS. In order to be able to track the notifications to delete/cancel them in the future.

The diagram of Alert is as follows:

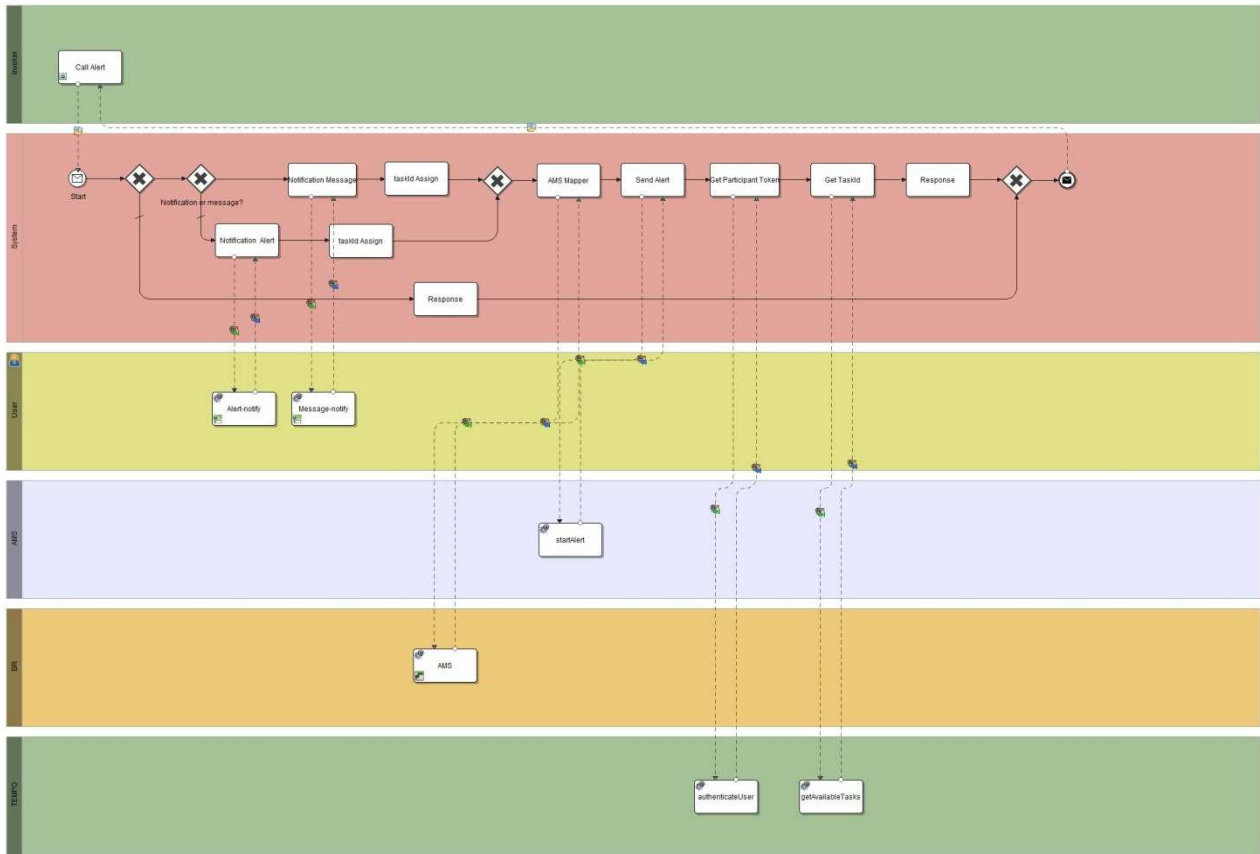


Figure 27 – Alert BPD

On the other hand CancelAlert BPD gets the two trackers of an alert (identification for AMS and Notifications) and cancels them.

The diagram of CancelAlert is as follows:

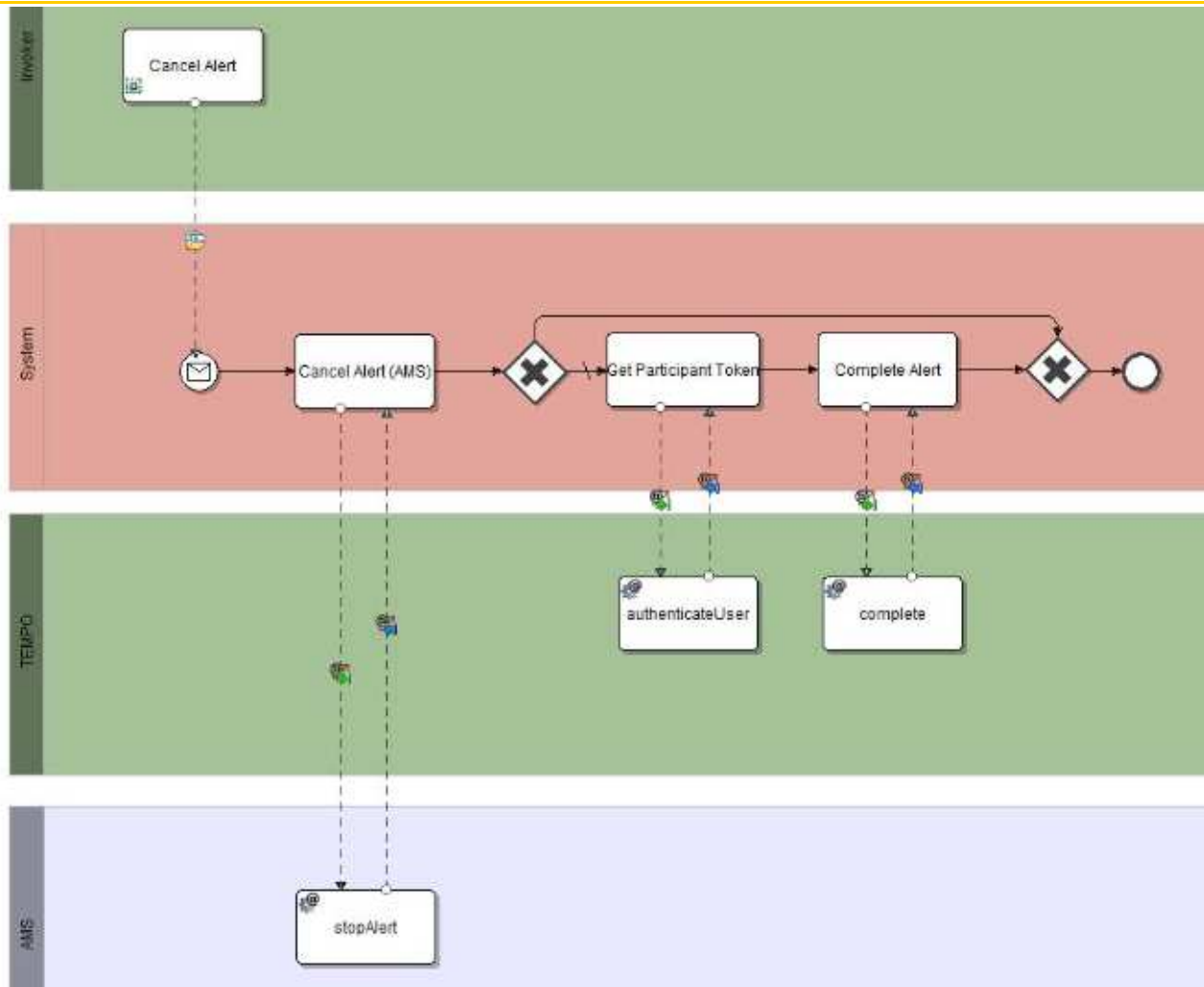


Figure 28 – CancelAlert BPD

TimeAlert and TimeAlert2 BPD's

Both diagrams are exactly the same, but it was needed to implement both for avoiding Intalio issues. TimeAlert BPD has the following functionality:

1. Starts a timer (with the value passed from Sacco process) and sends back its instance ID to Sacco process.
2. Waits until the timer reaches its value or until a message from Sacco process cancels the alert. In the first case the alert is triggered by calling the Alert BPD.
3. Then it will wait until the alert will be cancelled by Sacco process, for deleting it from AMS and notification system in the ReMINE workflow.

The diagram of TimeAlert is as follows:

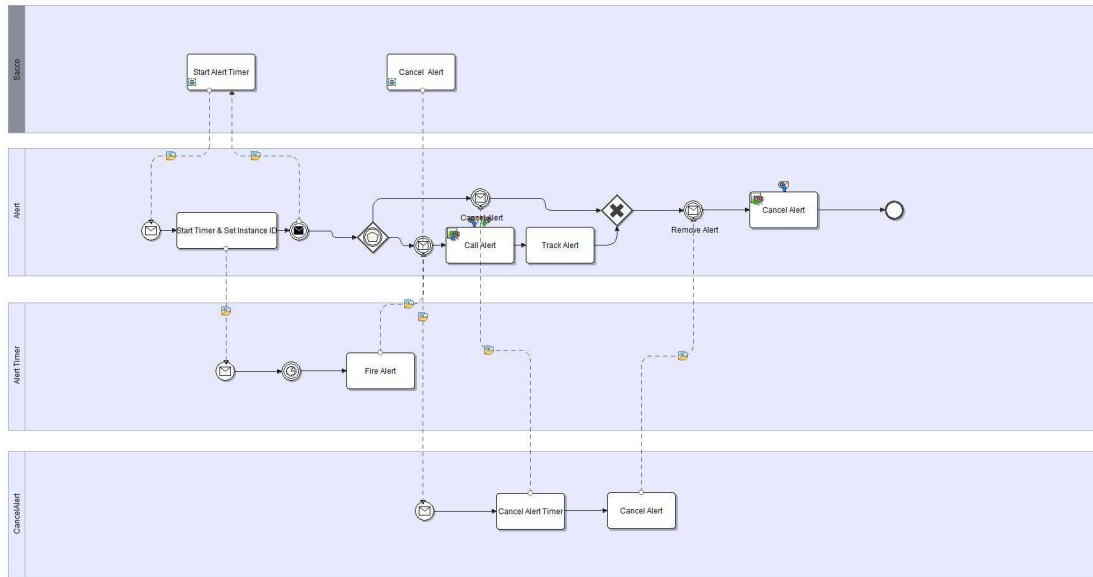


Figure 29 – TimeAlert BPD

5.1.1. User interface

The user interface for the obstetricians and midwives depending directly from the Process Mapper is divided in four different tabs:

- Patient List tab
- Tasks tab
- Notifications tab
- Processes tab

1. Patients tab

The ‘Patients’ tab shows all the patients that are waiting to be admitted to the service. By clicking the Admission link we will enter in the admission form.

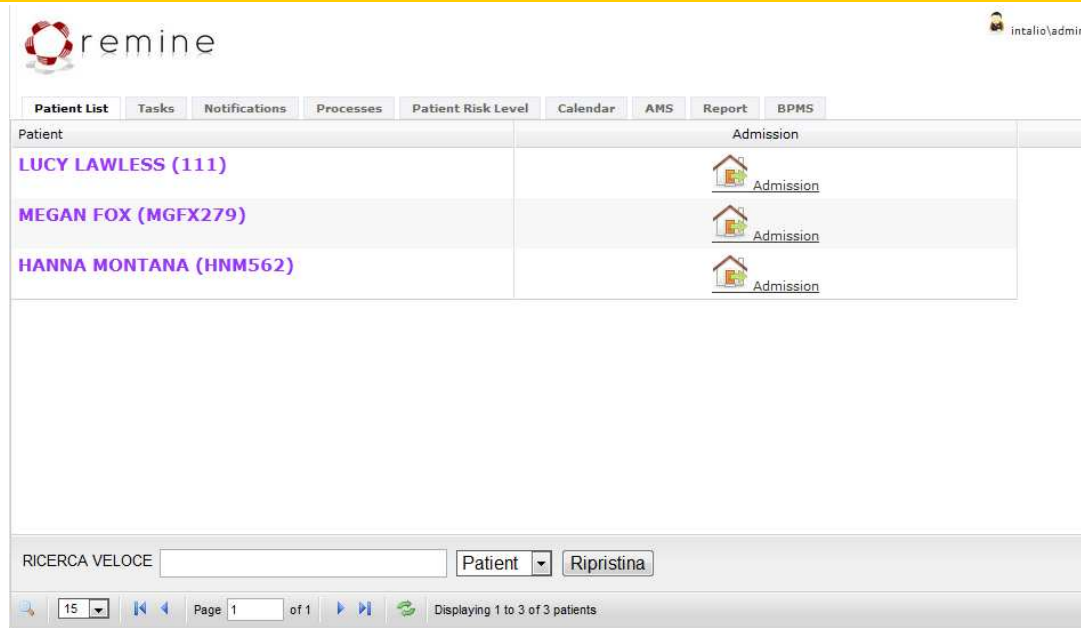


Figure 30 – Patients tab

Admission

The following figures depict the two forms associated with the 'Admission'. Once the first is completed, the second one will be opened automatically.



Figure 31 – Admission 1/2. Active Labour



Figure 32 – Admission 2/2. Low Risk Labour

2. Tasks tab

In the initial interface the end-user by pressing the tab ‘Tasks’, will be able to choose the required task for a given patient. The different tasks for each patient are:

- Bracelet Assignment
- FHR Monitoring (Start, Stop or Registration)
- Obstetrician Assessment
- Epidural Administration
- Exit

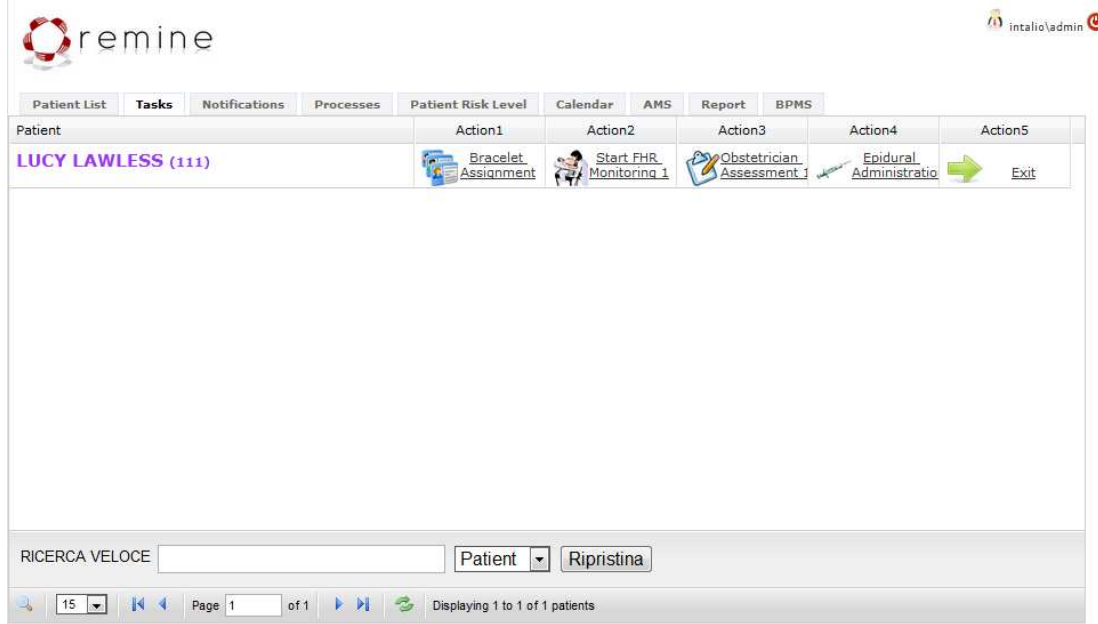


Figure 33 – Tasks tab

Bracelet Assignment

The following figure represents the form to assign a bracelet number to a patient.

Figure 34 – Bracelet Assignment

Start FHR Monitoring

Figure 35 – Start FHR Monitoring

Here the names or codes of the midwife, patient and device corresponding to this monitoring will be fulfilled. Once this form is completed, in the tab 'Tasks' a new task will be created to allow stopping FHR monitoring.

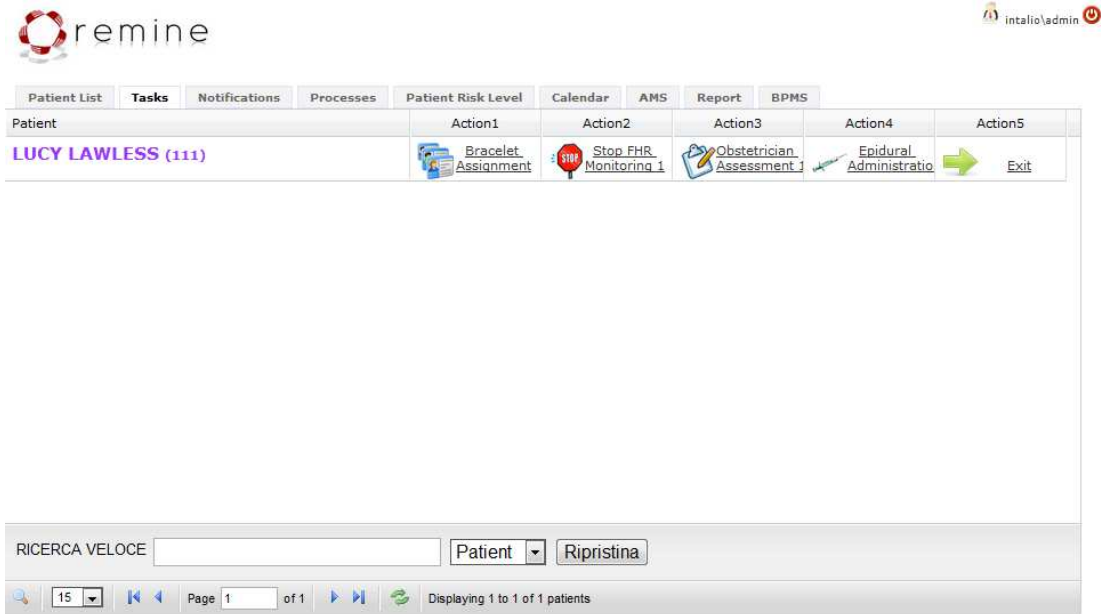


Figure 36 – Patient Tasks. After completed a 'Start FHR Monitoring' task

Stop FHR Monitoring

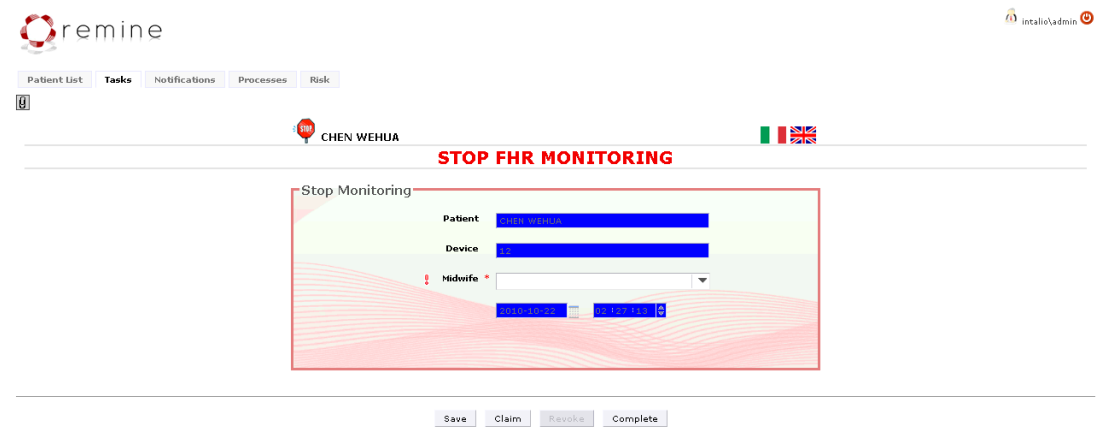


Figure 37 – Stop FHR Monitoring

This form stops the monitoring associated with the patient selected and device introduced in the “Start” form. Once is completed, it is removed from the tasks and the Registration FHR Monitoring task appears.

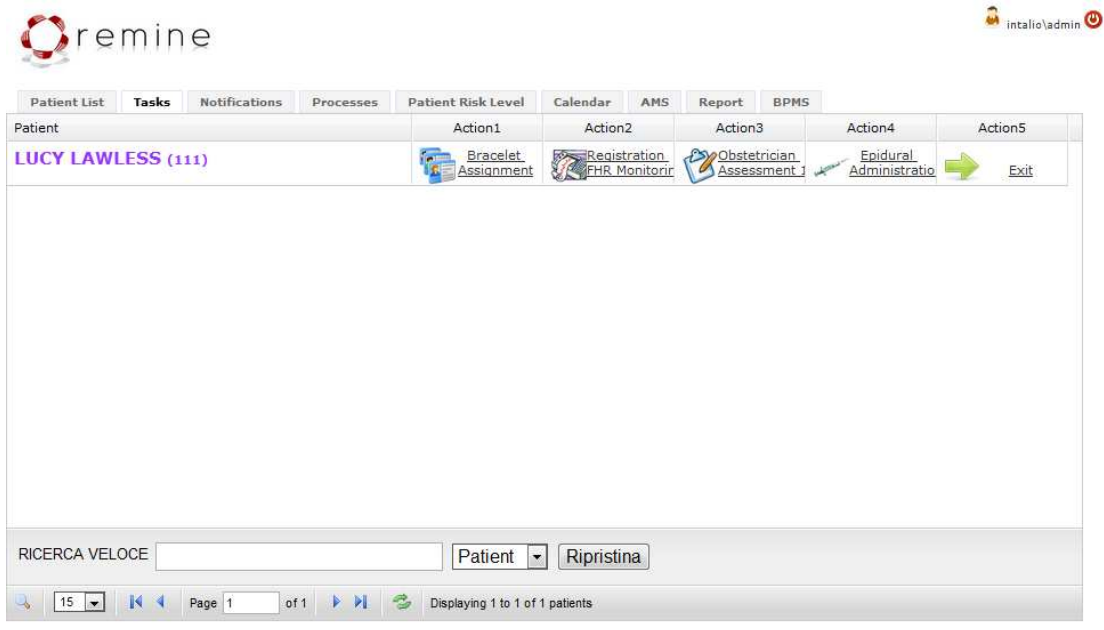


Figure 38 – Patient Tasks. After complete a 'Stop FHR Monitoring' task

Registration FHR Monitoring

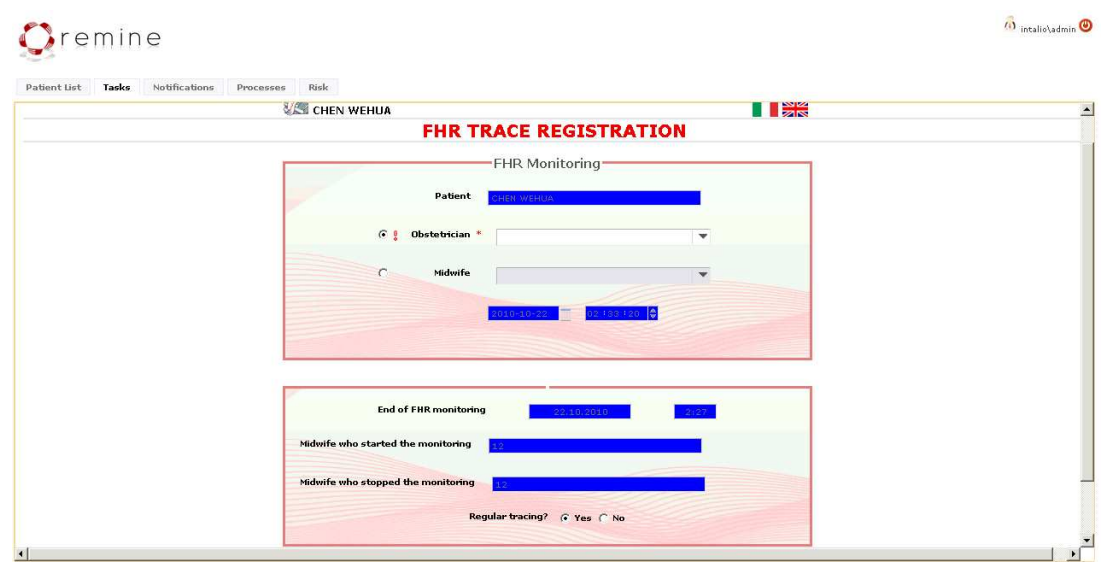


Figure 39 – Trace FHR Monitoring

This form is used to score the regular tracing of the FHR monitoring. When the Registration FHR Monitoring is completed, a new Start FHR Monitoring form is created in order to be ready for the next monitoring.

Obstetrician Assessment

Figure 40 – Obstetrician Assessment

Once this form is completed by the obstetrician, it will be automatically removed from the tasks. Also a new task to perform another obstetrician assessment will be created with the consecutive number written down in the task description.

Epidural Administration

Figure 41 – Epidural Administration

Once this form is completed by the obstetrician, it will be automatically removed from the tasks. Also a new task to perform another obstetrician assessment will be created with the consecutive number written down in the task description.

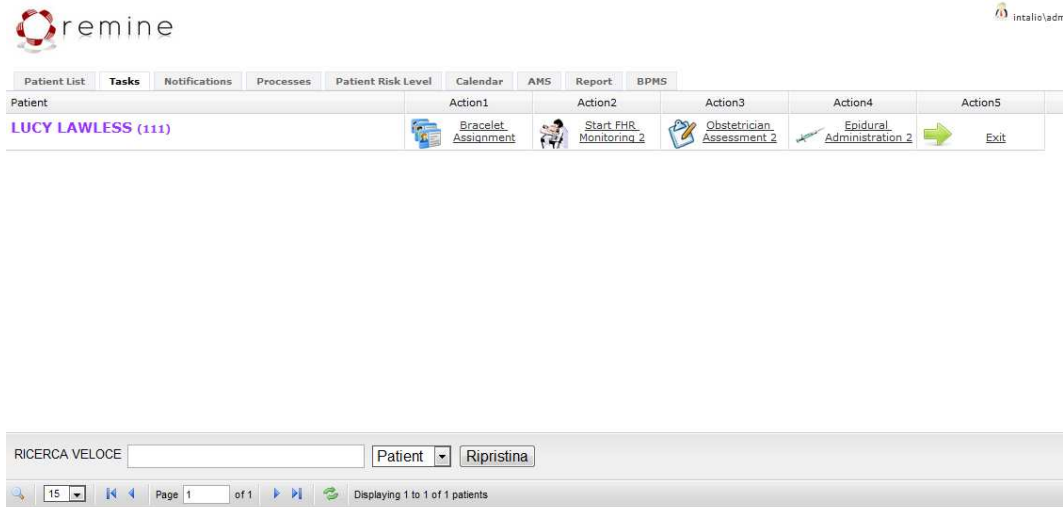


Figure 42 – Patient Tasks. After complete the first 'Obstetrician Assessment' task; the first FHR Monitoring cycle and the first 'Epidural Administration' task

A new 'Obstetrician Assessment' task, 'Epidural Administration' task and Start FHR monitoring are created with a consecutive number in the description for identifying them.

Exit

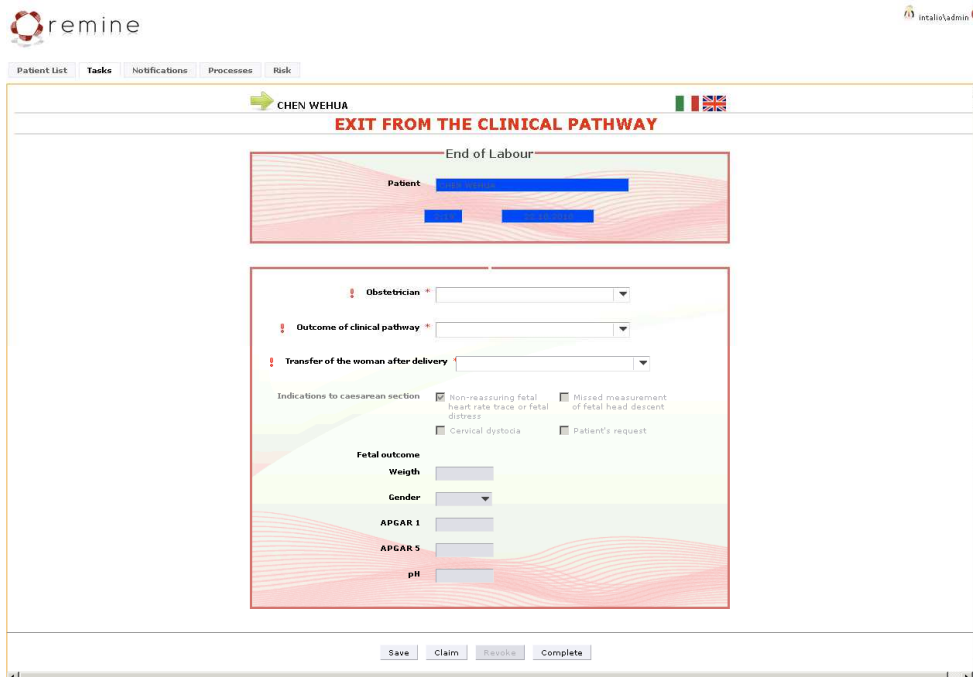


Figure 43 – Exit

The obstetrician through this form can exit from the scenario of a given patient. Therefore all the tasks and notifications associated to this patient will be removed from the ReMINE workflow.

3. Notifications tab

The 'Notifications' tab shows all the alerts triggered for each patient. They are divided in four groups, depending on the source/type of the alert:

- Admission
- FHR Monitoring
- Obstetrician Assessment
- Prediction

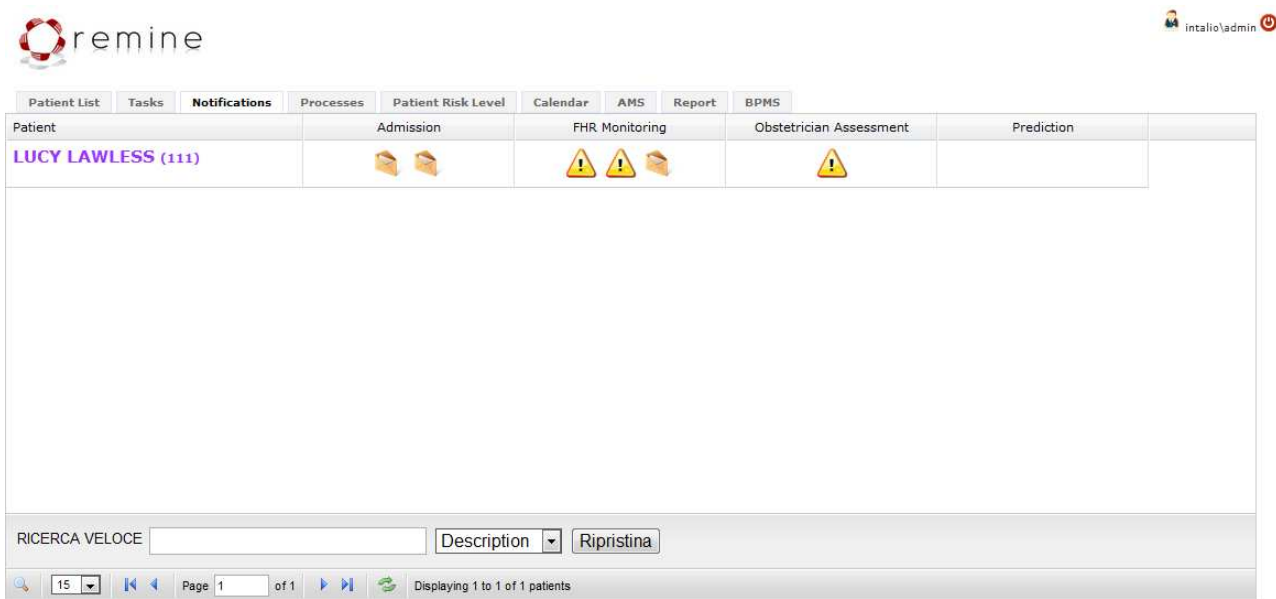


Figure 44 – Notifications tab

4. Processes tab

The 'Processes' tab shows all the processes to handle the contextual data of the hospital. They are divided in four groups, according to the projects developed (see previous section):

- Rooms
- Device Management
- Medical Staff Management: midwives, obstetricians and OSS
- Caesarean Section Management

Description	Action1	Action2	Action3
Rooms	Rooms		
Dispositivo Prenotato	Inizio Uso	Fine Uso	
Registro Presenza	OSS	Ginecologi	Obstetrica
Taglio Cesareo	Fine	Inizio	

RICERCA VELOCE Patient

15 Page 1 of 1 Displaying 1 to 4 of 4 types of processes

Figure 45 – Processes tab

5.2. Process Mapper for Kauhajoki

The scenario of the Kauhajoki Hospital focuses on patients hospitalized in the Acute Care for Elderly and especially in drug administration which is the most critical task in this ward. The primary users of this scenario are the nurses and more specifically the admission nurse and the nurse responsible for drug preparation and administration.

The processes modeled for Kauhajoki Hospital are the following: a) Admission, b) Drug Preparation, c) Drug Administration and d) Nurse Identification. For the description of the functionalities of process mapper for the Kauhajoki scenario we will present one of the scenario processes, the Nurse Identification process and we will focus in the tasks related in the CDA creation and storage in the database. Also are presented some general cases and forms for data input for the other three processes and for the Incident Report that was created for validation purposes.

For the Nurse Identification process execution 3 pools are created in Intalio Designer. One pool is created for the database, one for the ReMINE system and finally one for the user. The pool for the ReMINE system is defined as executable. In this pool are identified the functionalities of ReMINE that are represented as tasks and the Data Mapper is used for the mapping of the data flow between tasks.

The executable model for the Nurse Identification process is presented in the following figure:

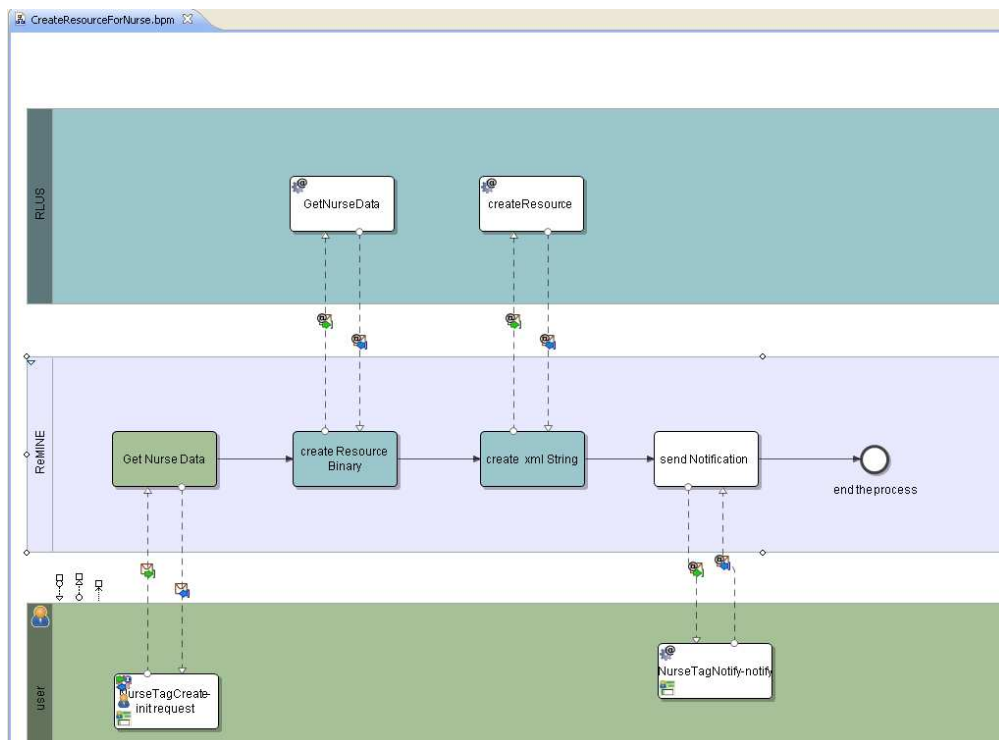


Figure 46 – Nurse Identification Executable process

Through the task “Create Resource Binary” a web service, that creates a CDA document for the nurse, is called. The parameters that are given are the nurse name and nurse’s RFID tag. The web service

D.3.4 Third Revision Data & Process model system framework

was developed in Java Eclipse. The XML DOM was used for the creation of the xml file. The XML DOM defines a standard way for accessing and manipulating XML documents. The DOM presents an XML document as a tree-structure.

The corresponding web service “CreateCDAforNurse-create_cda_process.wsdl” in the Process Explorer of the Intalio Designer is presented in the following figure:

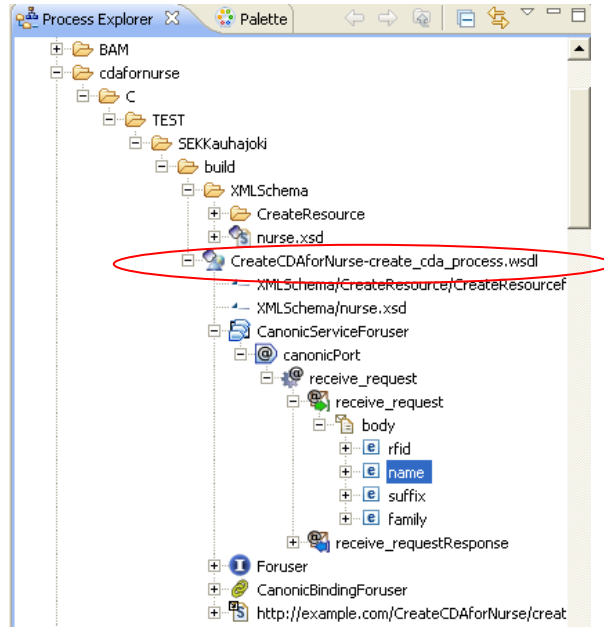


Figure 47 –The web service for the CDA creation

The source code that was developed for the creation of the web service is split in two parts the CDA creator (for structuring the CDA) and the data generator (input parameters). The Data Mapper for the Task “create Resource Binary” is presented in the following figure. As explained before the parameters that are given in the mapper are the nurse name and the nurse rfid tag.

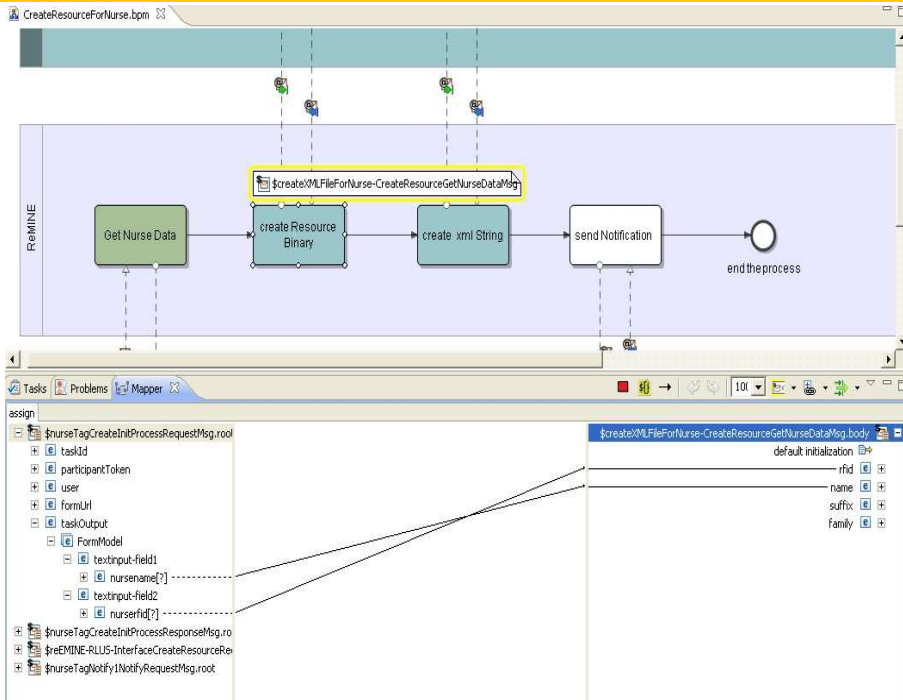


Figure 48 –Data Mapper for the task Create Resource Binary

Through the task ‘Create xmlString’ a web service, which through RLU stores in the database a new CDA. In this case a CDA for the nurse. This web service “ReMINE-RLUS-Interface.wsdl” in the Intalio Process Explorer is presented in the next figure:

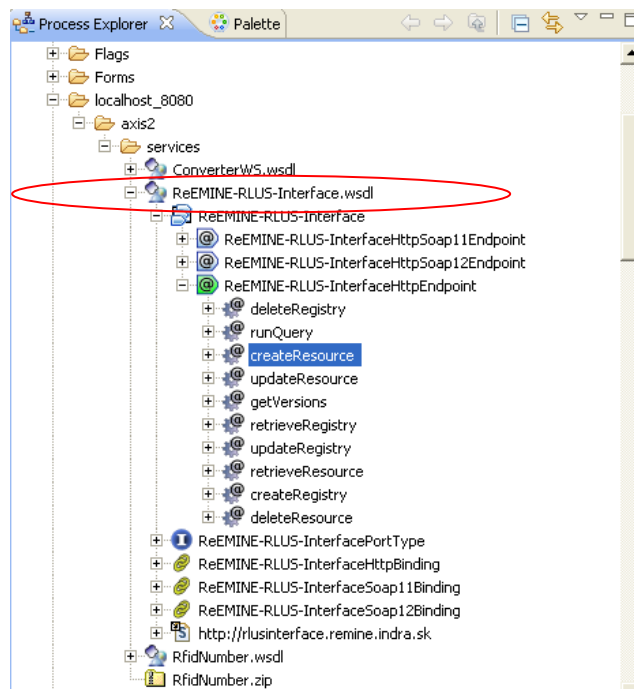


Figure 49 –The web service “ReMINE-RLUS-Interface”

In the next figure is presented the Data Mapper for the task “Create xmlString” where all the parameters are defined. The Data Mapper creates a data flow from left to right and provides the ability to graphically define all data expressions. Mappings consist in connecting elements/operators to other elements/operators.

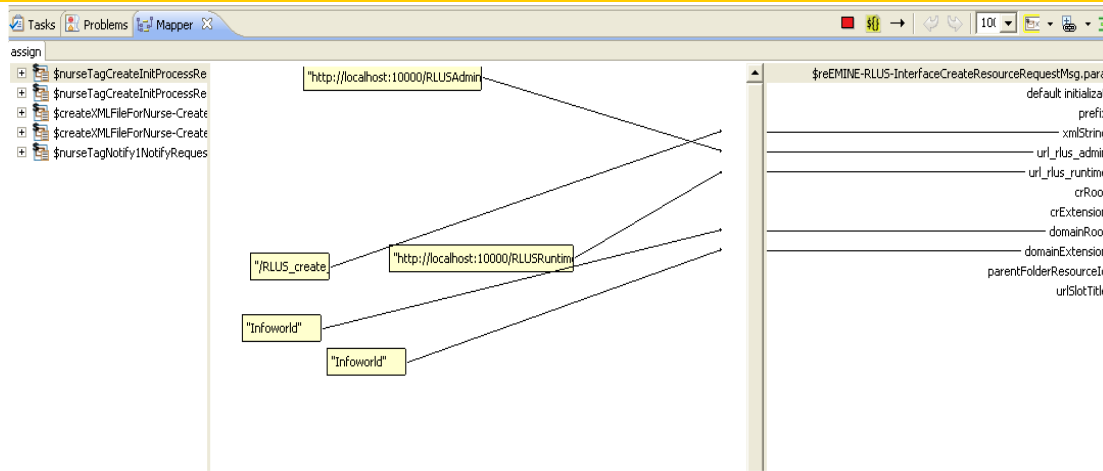


Figure 50 –Data Mapper for the task Create xmlString

Some general cases where the Data Mapper is used are presented in the next paragraphs. One of the cases that the Data Mapper is used is in looping sub-processes. The looping sub-process starts with a counter from the first patient and ends when all patients are processed. In the Data Mapper the first counter and the final counter were defined.

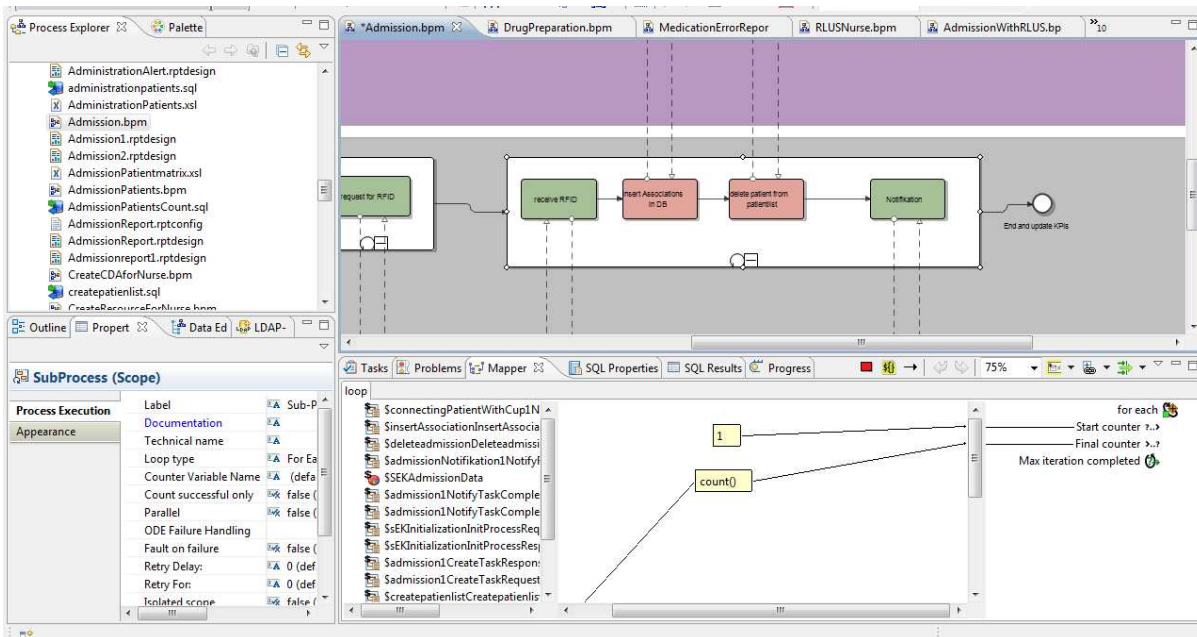


Figure 51 –Data Mapper for looping sub-processes (Admission Process)

Another case is the integration with the alerting. In the Data Mapper the parameters that are given is the name of the alert and the reason of the alert (for example an alert for an incorrect cup). In the next figure is presented the task for sending an alert during drug preparation process.

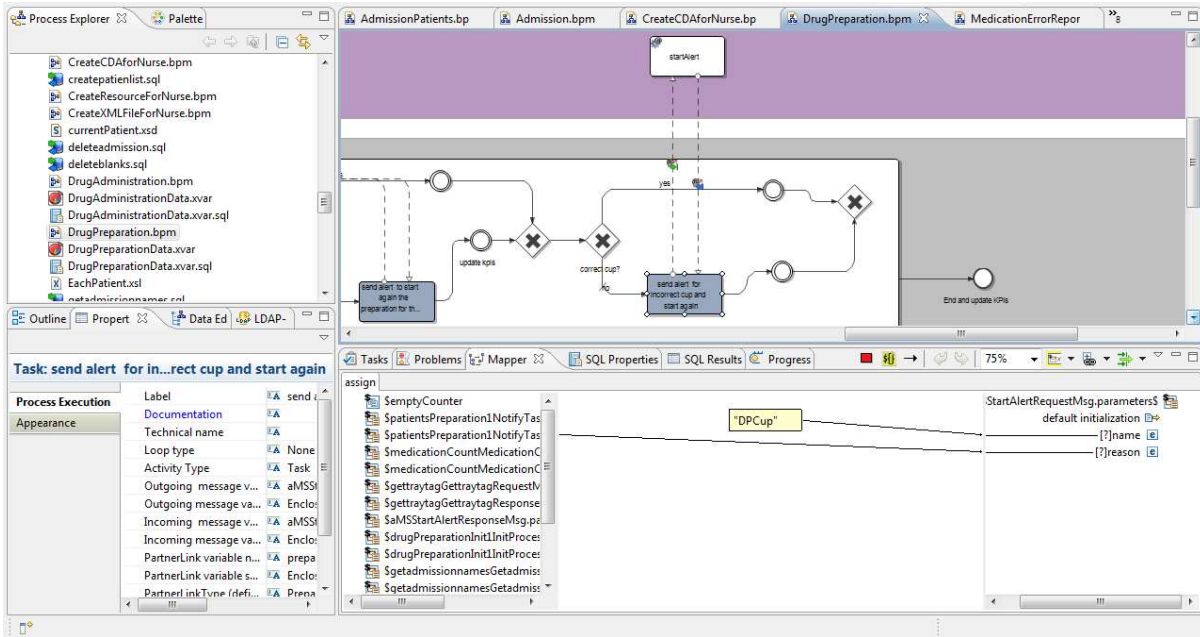


Figure 52 –Data Mapper for alerting integration (Drug Preparation Process)

For validation purposes for the Kauhajoki Hospital an incident report was implemented using the Intalio BPM tool. In the next screenshots are presented the Data Mapper for data input in the database (different database from ReMINE database where the user input from the incident report is stored there) and the interface for the incident report that the user will fill after ReMINE usage. It is integrated in the same interface with the processes for this hospital.

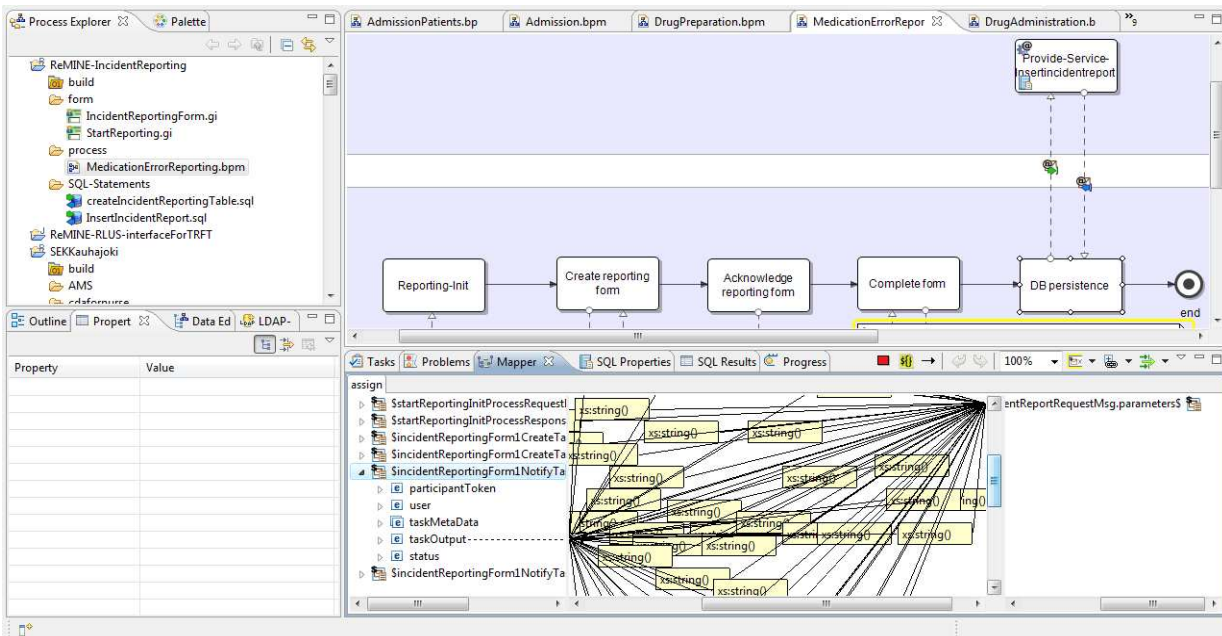


Figure 53 –Data Mapper for data input in the database (Incident Report Process)

Figure 54 –Sample of the fields of the incident report

For the Kauhajoki application all the forms in every process have been implemented in two languages (English and Finnish). The user has the ability to select the required language in every interface (two flags on top of every screen). For every form was developed a js-file (javascripts) –Logic.js and there between other there are two functions 1. English () and 2. Finnish (). When the user presses the flag for English everything translated in English and the same for the Finnish.

One of the forms translated in Finnish:

Figure 55 – Form in Finnish

The users via dedicated interfaces are able to choose the right process choose and afterwards are able to handle all patients separately and chose a specific task to execute for every patient. Every task corresponds in the execution of a process. After choosing a task new interfaces are opened where the user can input data or for example call an RFID tag serial number. All these data are inserted in the processes and according to the data inserted, the processes are exporting results such as alerts. When a task is

completed then this task disappears from the task list and only pending tasks are presented. During the process execution also web services are called for example a web service for the RFID tag or a web service for triggering the proper alert. In the next figures are presented examples of the data input interfaces.

Figure 56 – Form for nurse information input

Description	State	Created	Due	Priority	Attachments
create RFID for patient: Arnold Ameche	✓	3/2/2011 12:16 uu			
create RFID for patient: Bob Altman	✓	3/2/2011 12:16 uu			

Figure 57 – Tasks' list interface

Figure 58 – Data input (RFID tag input – scanning)

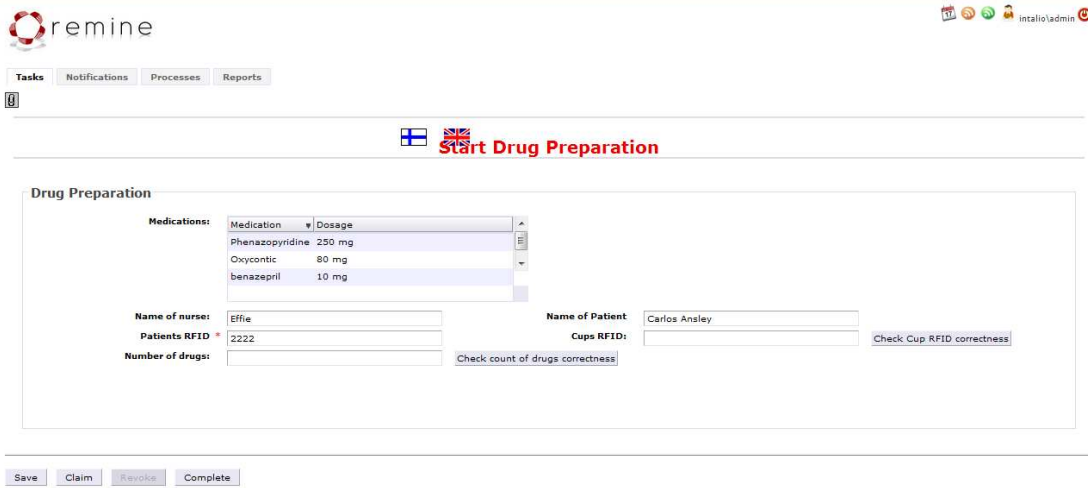


Figure 59 – Data input (drug preparation for a specific patient)

5.3. Process Mapper for Niguarda

5.3.1. Introduction

This section describes the purpose and realization of the model for Niguarda pilot. This model provides monitoring of hospital processes. In principle it checks if each step of a hospital process is executed in the right order and at the right time, if not this model notify to the responsible person(s) that the hospital process have not been executed right.

5.3.2. Overview

Let’s start with “How the model gets data”. First of all DEMs stores data to RLUS. RLUS then sends back resource ID of stored data to DEMs. It is the blue connection in figure 71. Then DEMs sends invoking message to Process Mapper(Intalio process made by INDRA). It is the red connection in figure 71.

This invoking message has these elements:

- Resource ID
- Event type
- Domain Root
- Domain Extension

Then process in Process Mapper uses the resource ID to retrieve actual data from RLUS. It is the black connection in figure 71.

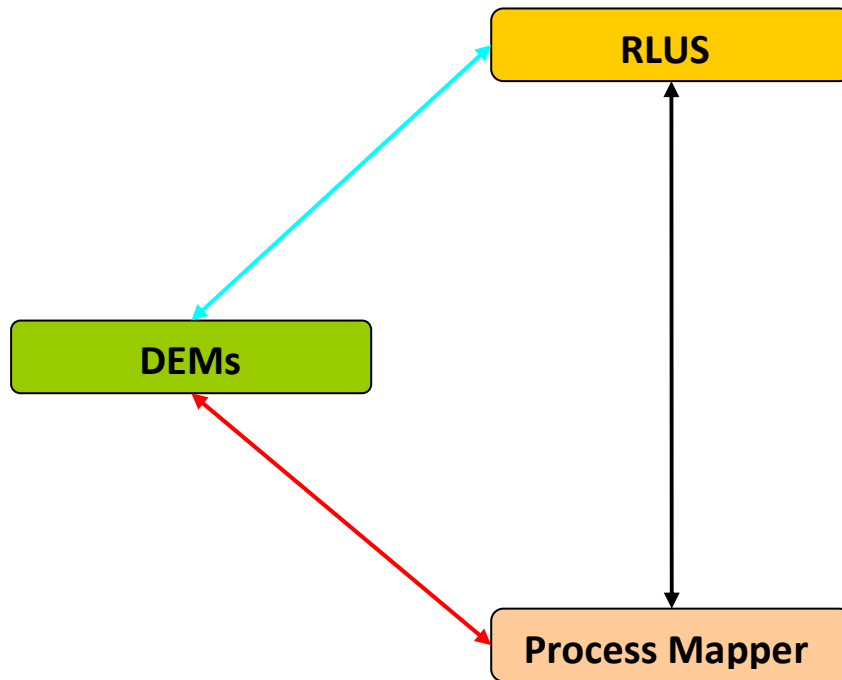


Figure 71 – How process mapper receives data

Let’s look at what actually happens in processes now. First of all main model called “NiguardaMsgSorting” has these diagrams: “MsgManager”, “PatientAdmission”, “NeuroAssessment”, “RadOrder”, “LabExam”, “ECGandAdministrationOfReperfusion” and “ReAssessment”. MsgManager diagram is entrance for whole model. This is the place where model receives invoking message with resource ID we mentioned above. It is represented by thick black connection between DEMs element and MsgManager element in figure 72. then MsgManager requests data from RLU by using resource ID field from invoking message. Requesting and retrieving of data from RLU is represented by two thin black connections between RLU and MsgManager in figure 72. Then MsgManager sends data retrieved from RLU to another diagram. This decision is made according to content of the eventType field from invoking message. This sending of data to another diagram is represented by green connections in figure 72. PatientAdmission and ReAssessment diagrams have a special position in model. PatientAdmission is triggering all the monitoring processes for patient. These processes run over diagrams: “NeuroAssessment”, “RadOrder”, “LabExam” and “ECGandAdministrationOfReperfusion”. Triggering of these processes is represented by blue connections in figure 72. ReAssessment diagram is triggered when patient is leaving the clinical pathway. If patient still has some running monitoring processes it should terminate these processes. This termination is represented by red connection in figure 72.

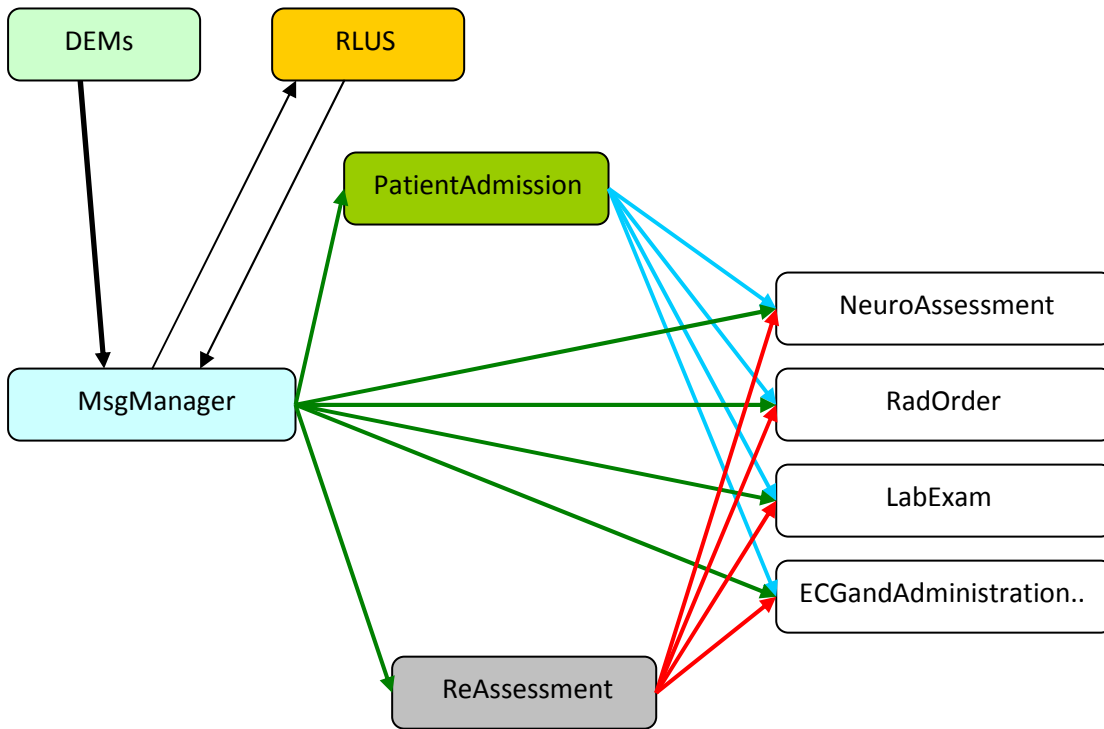


Figure 72 – Communication in “NiguardaMSGSorting”

5.3.3. Correlation

Correlation provides that all received messages go to the correct process instances. Now, Niguarda model uses only one correlation set which has one property. This property is set on “IDs” parameter of incoming messages. Every kind of message that Niguarda model can receive has “IDs” parameter. So if there is a new incoming message with “IDs” parameter, it is delivered to the instance of monitoring process which were started with the message with the same “IDs” value.

5.3.4. “MsgManager” diagram

There is no scenario for this model. It handles all incoming messages and forwards them to the correct instances of monitoring processes. It is possible to send messages directly to models, but we decided to receive every incoming message in the model MsgManager, because then you need only one wsdl for sending messages to Niguarda model. Also if we would like to set some business activity monitoring (BAM) over incoming messages, it would be better to set them in one model, no to do the same thing for 5 models. So let’s have a closer look on the beginning of the diagram (figure 73). In figure 73 there is green box. That is the point where the message with resource id from DEMs come. Then MsgManager retrieves data from RLUS via RLUS-interface webservice. As you can see, there is RLUS retrieve task in red ellipse in figure 73. There we are mapping resource id from invoking message as parameter to retrieveResource (also in red ellipse), which is method of RLUS-interface webservice. Then model receives data as a string encoded in Base64 code. In the next part of the model (blue ellipse in figure 73) model uses another webservice to

decode retrieved data. Then there is exclusive gateway component called “fork”(black circle in figure 73). There is decision made about the destination of decoded data(it means which process should model send the decoded data?). This decision is made according to content of eventType field in invoking message(green box).

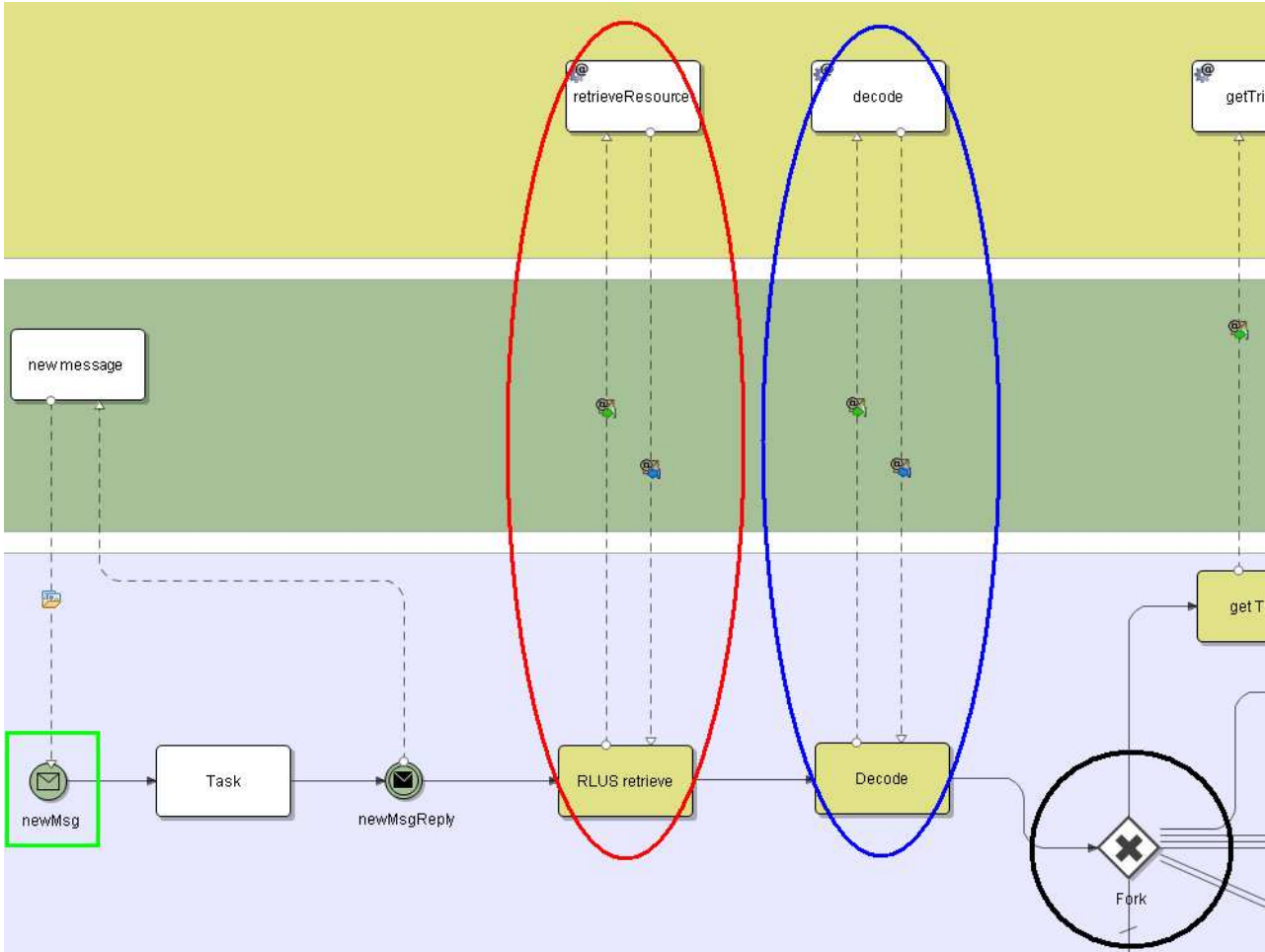


Figure 73 – Part of MsgManager

5.3.5. “PatientAdmission” diagram

The scenario for patient admission is very simple. Triage nurse does an assessment of the patient that has arrived at the A&E. Triage nurse alerts a neurologist consultant and fills the electronic A&E report with the patient data and the priority code. If symptoms and priority code match with the ones defined by the stroke protocol, ReMINE (process model) starts the countdown for administration of the reperfusion treatment.

So this diagram is also very simple. Process (ReMINE) is triggered by receiving of “Triage” message(ADT^A04). This Triage message comes from MsgManager. After that process analyze the data from “Triage” msg to find out if the patient will need some examinations, so it can start monitoring processes. If the patient will need some examinations, the process will trigger processes for monitoring hospital activities. You can see exclusive gateway where is this decision made in red circle in figure 74. Triggering of monitoring processes is made by using “Triage” msg to invoke monitoring processes over

niguarda process model. These monitoring processes are running over these diagrams “NeuroAssessment”, “RadOrder”, “LabExam” and “ECGandAdministrationOfReperfusion”. To make it more clear the message sending in blue ellipse in figure 74 is the very same thing as blue connection between PatientAdmission and NeuroAssessment in figure 72.

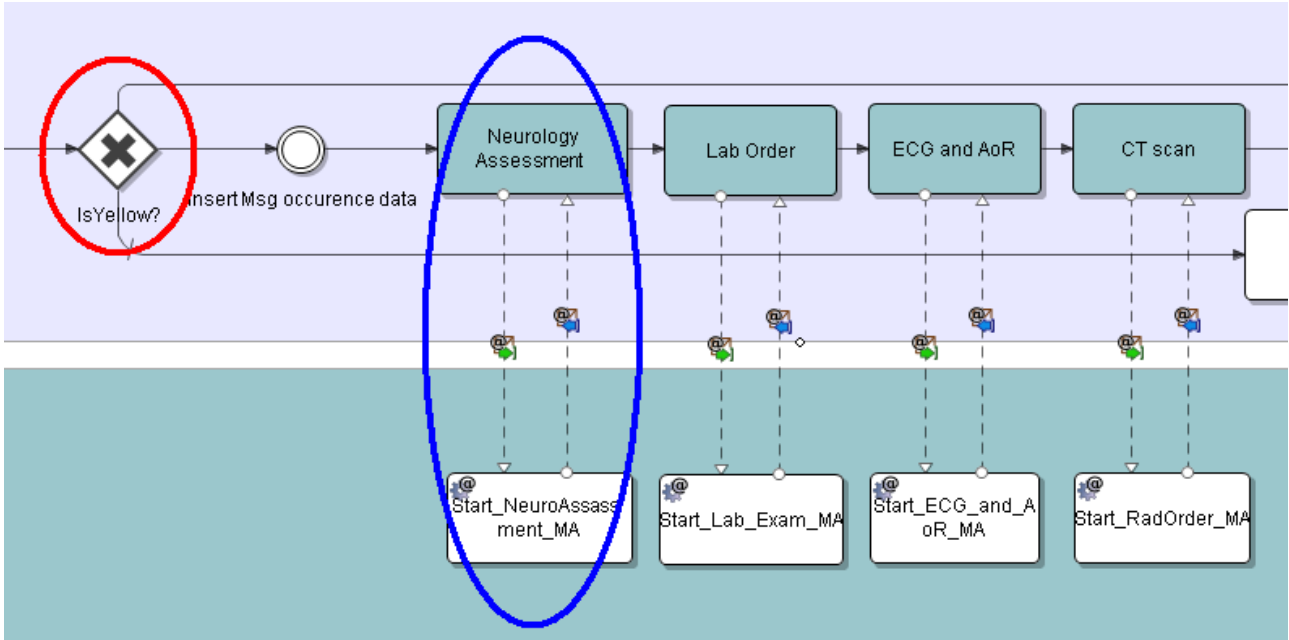


Figure 74 – Triggering of Monitoring processes

“Triage msg”(ADT^A04) has these elements:

- ID’s (Patient Id + Visit Id)
- Patient demographics
- Priority code
- Diagnosis allowing to detect the patient as “yellow stroke”
- Clinical information collected for the patient
- Time of the end of triage
- Author of the triage

More detailed description of “Triage msg” you can find in document “ReMINE_D7.2_NoemalLife_WP7_V1.0_Final.pdf”.

5.3.6. "NeurologyAssessment" diagram

The whole idea of this part of scenario is that, neurology consultant has 30 minutes from start of triage to do an assessment of the patient. If the neurology consultant did not manage to do the assessment monitoring process (ReMINE) should send a message to the responsible person via AMS(alerting message system).



Figure 75 – Beginning of NeuroAssessment diagram

Process starts by receiving "Triage" msg from PatientAdmission process, it is the green box in figure 75. Then process needs to get expiration time value from Clinical protocol. Clinical protocol is represented by another diagram which contains decision table with clinical protocol data. This query is made right after the

process triggering(tasks in red elipse in figure 75). Then process continue to its logic part.

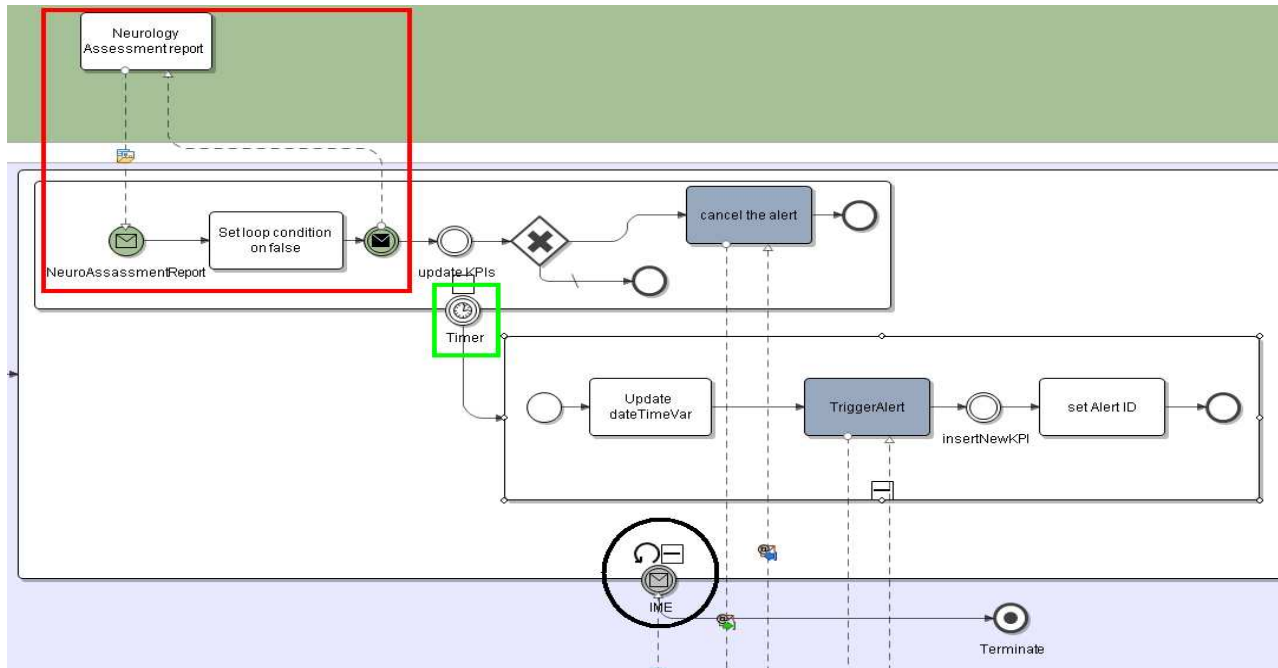


Figure 76 – NeuroAssessment diagram logic

The part of Neuroassessment diagram where is the logic executed, you can see in figure 76. As you can see, process wait for incoming NeuroAssessment msg(ADT^A08). It is the part of diagram in the red box in figure 76. If the message would come on time, KPI(key performance indicators) data are updated(update KPIs task in figure 76) and then process is normally finished. If the message did not come on time, ptocess should trigger alert in AMS. This is handled by intermediate timer component which you can see in green box in figure 76. This component uses data received from clinical protocol before(red elipse in figure 75). So when the NeuroAssessment msg did not come on time this timer component will trigger the execution of the second subprocess. It is the subprocess which contain tasks “update dateTime Var”, “TriggerAlert“, “InsertNewKPI” and “set Alert ID”. The task “TriggerAlert” send data to supporting diagram which purpose is to trigger alerts in AMS. When the alert is triggered process still needs to wait for NeuroAssessment msg. This is handled by loop which is set on the subprocess that encapsulate whole this part of diagram. You can see it in black circle in figure 76. There is also intermediate message receiver, where can be received the message from ReAssessment process. This message will cause the termination of the process.

“NeuroAssessment msg”(ADT^A08) has these elements:

- ID’s (Patient Id + Visit Id)
- Patient demographics
- Clinical information collected for the patient
- Clinical information related to the executed assessment

- Time of assessment
- Doctor in charge of patient

More detailed description of “Triage msg” you can find in document “ReMINE_D7.2_NoemaLife_WP7_V1.0_Final.pdf”.

5.3.7. “RadOrder” diagram

This part of the scenario is very same as the neurology assessment part. Neurology consultant has 30 minutes (according to the scenario) from the triage start to the order cerebral CT scan. If the CT scan order is not registered in that time ReMINE process alerts to the neurology consultant or to a different person.

So the principle of this model is the same as in the model above. The only difference is that this model has to receive different kind of message. After triggering this monitoring process, it waits on “Rad Order”(ORM^O01) message. If this message did not arrive on time the process will alert a responsible person and wait for the message to come. If the message comes on time the process should update KPI data and finish.

“RadOrder msg”(ORM^O01) has these elements:

- ID’s (Patient Id + Visit Id)
- Patient demographics
- Order ID
- Order date and time
- Ordered examination
- Doctor in charge of patient

More detailed description of these messages you can find it in document “ReMINE_D7.2_NoemaLife_WP7_V1.0_Final.pdf”.

5.3.8. “LabExam” diagram

If the result of the triage says that it is necessary to make the patient blood exams, the neurology consultant has a certain time(according to the scenario 30 min from the triage start) to order the blood exams. If any laboratory exam order is not registered in that time our process should alert to some

responsible person. In case that the blood exam order was registered, there should be a response in some amount of time (set by the hospital risk manager) from the laboratory advising that the blood samples were checked-in. If the blood samples are not checked-in on time, the process should alert to some responsible person and wait for the “LabCheckin” message. After the “LabCheckin” message arrives, there are 45 minutes (according to scenario) from the triage start to get results of the blood examination. If the “LabResults” message did not arrive on time, the process should alert to some responsible person and wait again for “LabResults”. If “LabResults” arrives on time, the monitoring process should end. And as we mentioned before, the expiration times should be loaded from the Clinical protocol.

The scenario is very similar to the scenarios above. The main difference is that this model receives not 1 message but 3 after the process was triggered. Therefore in the model there will appear a logical part of diagram encapsulate into sub-process for each kind of message that can be received. This parts are very similar to diagram in figure 76. So there is be diagram part for handling LabOrder(OML^O33) then for LabCheckIn(OML^O21) and then for LabResults(OUL^R22). You can see it in figure 77. It can be done this way because we know that messages should come exactly in this order.

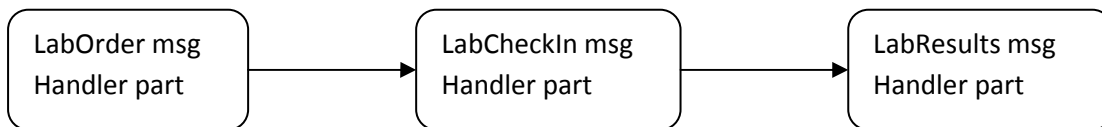


Figure 77 – LabExam messages order

As we said before the parts of the diagram for handling messages are similar to the diagram in figure 76, but the part for the handling LabResults diagram is slightly different. The difference is in the part after the LabResult msg is received(the rest is very similar), where should be done the check of the concrete lab result tests. These tests are Glycaemia, Platelets, ptt and pt. This is done by calling another supporting diagram called “LabResultAnalysis”. Then if the result value is out of range there is alert triggered in AMS as you can see in the figure 78. After all 3 messages are received is the process finished, or it can be terminated by ReAssessment process.

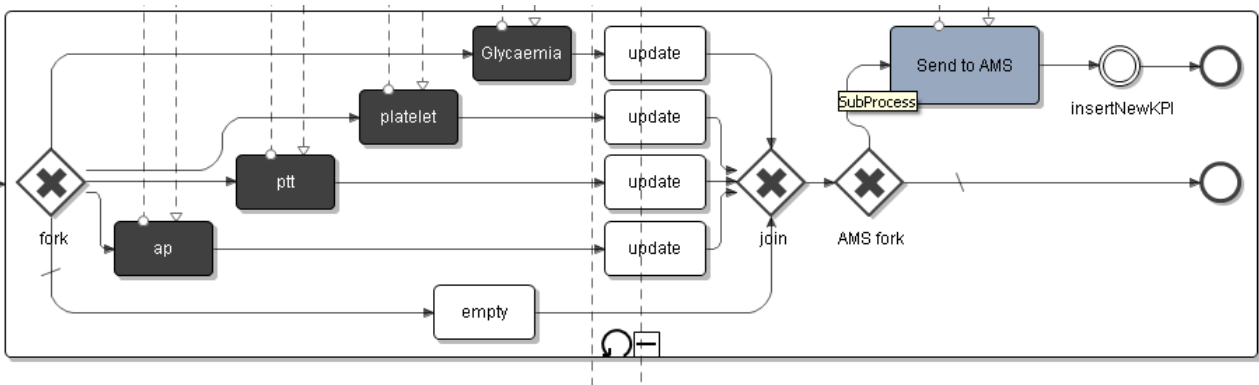


Figure 78 – Calling of test check diagram

“LabOrder msg” has these elements:

- ID’s (Patient Id + Visit Id)

-
- Patient demographics
 - Order ID
 - Order date and time
 - Ordered examination
 - Collected sample ID's
 - Doctor in charge of patient

“LabCheckin msg” has these elements:

- ID's (Patient Id + Visit Id)
- Patient demographics
- Order ID
- Order status
- Collected sample ID's
- Samples status
- CheckIN date and time
- CheckIN author

“LabResults msg” has these elements:

- ID's (Patient Id + Visit Id)
- Order ID
- Order status

- Tests code
- Tests result
- Test status
- Tests completion date and time
- Author of the tests

More detailed description of these messages you can find it in document “ReMINE_D7.2_NoemaLife_WP7_V1.0_Final.pdf”.

5.3.9. “ECGandAdministrationOfReperfusion” diagram

This part of the scenario is different. There are 3 different messages: ‘ECGExecution’(ADT^A08), ‘Treatment’(ADT^A08) and ‘AandEDischarge’(ADT^A03). There is no time out for ‘ECGExecution’ message, but it must be received before one of ‘Treatment’ or ‘AandEDischarge’ message is received. If it’s not received before, ReMINE should alert to the responsible person. Then, if in 70 minutes from the start of triage it is not received ‘Treatment’ message ReMINE should alert to the responsible person. In case that one of the monitoring activity message is received, the process ends. If ‘AandEDischarge’ message does arrive after the first 80 minutes from triage start, the model also alerts to the responsible person. If “Treatment” msg is received after 90 minutes from triage start process should notify responsible person. To end the monitoring activity process the model needs to receive just one of ‘AandEDischarge’ or ‘Treatment’ messages.

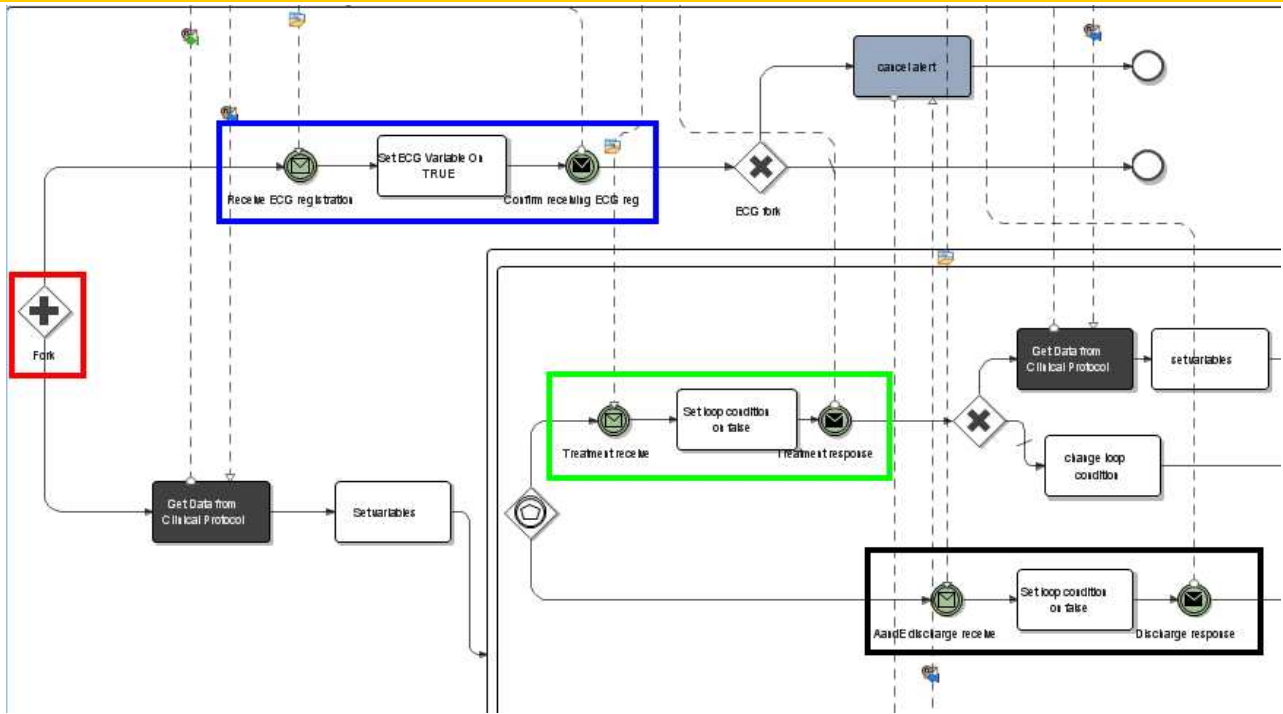


Figure 79 – ECGandAdministrationOfReperfusion diagram message receiving logic

The model uses a parallel gateway to make 2 paths. You can see this parallel gateway in red box in figure 79. First path is for receiving ECG msg(blue box in figure 79) and the other one is for treatment msg(green box in figure 79) and discharge msg(black box in figure 79). But in the second path if treatment msg is received it is not possible to receive discharge and vice versa. This is solved by component called exclusive event based gateway. It is the component before the green and black box in figure 79. This component causes that only the path where is the msg received will be executed.

Alert triggering and timeout cases are handled similar as in the diagram above .

“ECG execution msg”(ADT^A08) has these elements:

- ID’s (Patient Id + Visit Id)
- Patient demographics
- Examination executed
- Examination date and time
- Doctor in charge of patient

“Treatment msg”(ADT^A08) has these elements:

- ID’s (Patient Id + Visit Id)
- Patient demographics
- Treatment executed

- Treatment date and time
- Doctor in charge of patient

“A&EDischarge msg”(ADT^A03) has these elements:

- ID’s (Patient Id + Visit Id)
- Patient demographics
- Clinical information collected for the patient
- Time of discharging
- Doctor in charge of patient

More detailed description of these messages you can find it in document “ReMINE_D7.2_NoemalLife_WP7_V1.0_Final.pdf”.

5.3.10. “ReAssessment” diagram

For now there is no scenario for this model. The only thing we know is that incoming of “ReAssessment msg” causes (if the data of msg fulfills some conditions) the end of niguarda monitoring. So the diagram just trigger supporting diagram called “ClosePathway” which sends termination msg to all monitoting processes. If some of that processes runs it will be terminated.

“ReAssessment msg” has these elements:

- ID’s (Patient Id + Visit Id)
- Patient demographics
- Clinical information collected for the patient
- Clinical information related to the executed assessment
- Time of assessment
- Doctor in charge of patient

5.3.11. Supporting diagrams

For now there are 3 supporting diagrams: “ClosePathway”, “LabResultAnalysis” and “SentAlertHandler”. ClosePathway is used by Reassessment and ECGandAdministrationOfReperfusion diagrams. It is for canceling patient from clinical pathway and this could happened when ReAssessment or Discharge event happened. LabResultAnalysis is used by LabExam diagram to analyse actual results which came in LabResult msg and SentAlertHandler is used by each diagram to trigger alert in AMS.

5.3.12. Alerts list

Niguarda model diagrams can trigger several types of alerts. These alerts are defined in AMS and there are also defined persons who should receive alerts. Here is the list:

FIBTREATMENT_TIMEOUT_ALERT (triggered when treatment msg was not received on time)

DISCHARGE_TIMEOUT_ALERT (triggered when discharge msg was not received on time)

ECG_LATECOMING_ALERT (triggered when ecg msg was not received before treatment msg)

LABORDER_TIMEOUT_ALERT (triggered when laborder msg was not received on time)

LABCHECKIN_TIMEOUT_ALERT (triggered when labcheckin msg was not received on time)

LABRESULTS_TIMEOUT_ALERT (triggered when labresults msg was not received on time)

LABRESULTS_VALUE_ALERT (triggered when lab test results are out of range)

NEUROASSESSMENT_TIMEOUT_ALERT (triggered when neuroassessment msg was not received on time)

RADORDER_TIMEOUT_ALERT (triggered when radorder msg was not received on time)

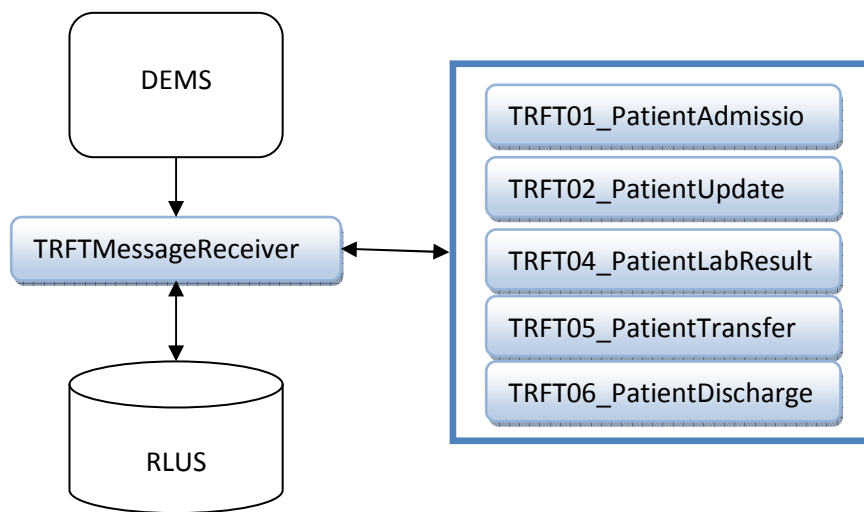
5.4. Process Mapper for TRFT

5.4.1. Introduction

This section describes the purpose and realization of the model for TRFT pilot. This model provides monitoring of hospital processes. In principle it checks if each step of a hospital process is executed in the right order and at the right time, if not this model notifies the responsible person that the hospital process have not been executed right. Processes are implemented by the deliverable D7.2.

5.4.2. Overview

The pilot consists of several Intalio Projects: the main project and several utility projects. The main project contains 15 diagrams, 6 of them are message processing processes the rest of them are utility processes and processes implementing the required functionalities.



5.4.3. TRFT message processing

DEMS messages are passed to the “*TRFTMessageReceiver*” process, which retrieves the proper resources, extracts the CDA documents from it and then passes it to the proper message handler process (see table below):

Message handler process name	DEMS message type
TRFT01_PatientAdmission	ADT^A01
TRFT02_PatientUpdate	ADT^A08
TRFT04_PatientLabResult	ORU^R01
TRFT05_PatientTransfer	ADT^A02
TRFT06_PatientDischarge	ADT^A03

There are the following restrictions regarding the message processing:

1. In the first place, a patient admission (TRFT01) message needs to be received for each patient at the start of the clinical pathway
2. After the admission, there may come several update (TRFT02), transfer (TRFT05) or laboratory result (TRFT04) messages for the patients in any order.
3. The clinical pathway is completed by a discharge message (TRFT06)

5.4.3.1. TRFT01_PatientAdmission

This is a message handler process. It handles ADT^A01 message. When the message is received it is the starting point of the monitoring activity, after processing the message it starts up the a laboratory result waiting process.

Data extracted from the admission message:

- ID's
- Patient demographics
- Patient address
- Admitting ward

5.4.3.2. TRFT02_PatientUpdate

This is a message handler process. It handles ADT^A08 message. When the patient's data are changed the patient's medical records should be updated. On the beginning, previous data are retrieved and updated with the new ones. Then the patient's infection alert code is determined and finally the patient's updated record is saved.

Data extracted from the update message:

- ID's
- Patient demographics
- Patient address
- Admitting ward
- Date and time of admission

5.4.3.3. TRFT04_PatientLabResult

This is a message handler process. It handles ORU^R01 messages. This process is started up at the admission and waits for corresponding laboratory results.

The incoming messages are correlated by the patient's identifier, so laboratory result waiting for one patients' results will not be affected by results for other patients – it ensures correct paring of the messages.

During the process there are alerts sent to ICT in case the laboratory tests take too much time or the patient's risk code is changed from green to red (as defined in the D7.2 deliverable section *Monitoring of MRSA screening*).

After the laboratory result is processed the patient's risk code is re-determined and the patient record is updated with the actual infection alert code then the process is restarted to be able to receive the next result messages. This is repeated until the receiving the terminating message from the process: *TRFT06_PatientDischarge*.

Data extracted from the laboratory result message:

- ID's

- Patient demographics
- Order ID
- Order date and time
- Ordered tests
- Ordering ward
- Ordering person
- Order status
- Test performed
- Test result
- Test execution date and time

5.4.3.4. TRFT05_PatientTransfer

This is a message handler process. It handles ADT^A02 message. When patient is transferred from one room (ward, cubicle) to another the patient's medical records should be updated. This process updates patient data based on the captured transfer message and re-determines the patient's infection risk code because the risk level can depend on the type of the ward the patient is admitted into.

Data extracted from the transfer message:

- ID's
- Patient demographics
- Patient address
- Old ward
- New ward
- Date and time of transfer

5.4.3.5. TRFT06_PatientDischarge

This is a message handler process. It receives ADT^A03 message. This event is released when a patient is discharged. This process stops the laboratory result waiting by sending a terminating message to the proper instance of the *TRFT04_PatientLabResult* process.

Data extracted from the discharge message:

- ID's
- Patient demographics

- Patient address
- Discharging ward
- Date and time of discharge

5.4.4. Utility processes

5.4.4.1. RankPatientRiskLevel

The TRFT pilot is focusing on infection control, because of this, one of the most important processes is the one ranking the patients to different infection risk levels. This process works with the incoming patient record. It collects all necessary data then invokes the **Errore. L'origine riferimento non è stata trovata.** WS which contains the ranking rules. This makes it possible to change the ranking rules without un-deploying the main project just by deploying the modified rules.

5.4.4.2. Scheduler

This process ensures that the daily reports and the cubicle status reports processes are started up periodically (daily). After the deployment this process is started and it starts up all the necessary scheduled processes.

5.4.5. Processes implementing the required functionality

5.4.5.1. Creation of a contact list

This functionality is defined in the deliverable: *D7.2 – Pilots Test environment definition – 6.1 – 4* Creation of a contact list (page 106)

5.4.5.1.1. CreateContactList

When an infection is detected ward manager reports it to the *ReMINE* via a form. After the report *ReMINE* creates a list of possible infected patients. On this list are patients who stayed in the same ward with the patient reported as infected in the same timeframe. These reports are sent to the infection control team who can decide whether to take further actions or not (outside *ReMINE*).

5.4.5.2. Patient state reporting

It implements the functionality described in *D7.2 – Pilots Test environment definition – 6.1 – 1* Rank coding of possible infected patients and notification to ward managers: STEP 3 and STEP 4 – 5 (page 105).

These functionalities are implemented in the following three processes:

5.4.5.2.1. DailyReports

This process ensures selection of patients with an infection risk code and then for each of these patients it starts up an instance of the *DailyReportsUpdateHandler* process.

5.4.5.2.2. DailyReportsCreate

It can be started up from the *Intalio Server's Workflow* interface's *Processes* tab via the *Daily report – add* process by filling in a patient's ID. This process starts an instance of the *DailyReportUpdateHandle* process instance for the patient with the given ID.

5.4.5.2.3. DailyReportsUpdateHandler

Ward managers and the patient flow team are daily notified by ReMINE about the state of the patients in their wards and in the hospital. These reports can be changed by the infection control team. This process generates reports about a patient for the infection control team, the patient flow team and for the proper ward manager.

When infection control team changes a report, a new one is generated for the ward manager and the patient flow team with incremented suffix index. Ward managers and the patient flow team can only view the notifications, but can not change them. Infection control team can change the infection risk code of the patient, add free text comments to the patients and can remove patients from the report.

5.4.5.3. Monitoring of MRSA screening¹

This functionality is defined in the deliverable *D7.2 – Pilots Test environment definition – 6.1 – 3* Monitoring of MRSA screening (page 106). The monitoring of the screenings is integrated into the event handler process *TRFT04_PatientLabResult*. There are two alerts generated:

- Alert about long lasting screening **Errore. L'origine riferimento non è stata trovata.**
- Alerts about risk code change from green to red

5.4.5.4. Deep cleaning of cubicles

This functionality is defined in the *D7.2 – Pilots Test environment definition – 6.1 – 4* Creation of a contact list (page 106).

5.4.5.4.1. DeepCleaning

When a patient is discharged from the hospital or transferred to another ward, it is required to clean up his previous place. There are two different types of cleaning: internal (normal) and external (deep) cleaning.

When a patient leaves a ward, the ward's manager (WM) needs to control his data and report his leaving to the infection control team (ICT). ICT then decides whether cubicle needs normal or deep cleaning and sends cleaning request to terminal cleaning team (TCT) or to external cleaning team (ECT). After this they need to register the date and time of the cleaning. ICT receives an update after their reports.

During this process there are AMS alerts sent to:

1. WM receives AMS alert about the assigned task
2. ICT receives AMS alert about the available WM report
3. TCT receives AMS alerts about the required cleaning

¹ *D7.2 – Pilots Test environment definition – 6.1 – 3* Monitoring of MRSA screening (page 106)

5.4.5.4.2. CubicleStatusReport

Ward managers need to report the state of the cubicle to the infection control team daily. This process reminds them this duty and creates a task in the Intalio Workflow. When WM filled in the form, the ICT receives a notification about the ward's state.

5.4.5.5. ReportInfectedPatient

When an infection occurs, the ward manager can report this fact for the ICT via starting the process.

5.5. Business rules Engine (BRE) integration

Business rules component is an integral part of the REMINE platform and part of the technological advantages of the project. The use of business rules within REMINE enables complex decision logic on the BPMN diagrams to be abstracted and offered in a pluggable fashion using the service oriented architecture paradigm.

For REMINE project the use of BRE provides a mechanism to evaluate several patient data conditions in a separate module and thus provide an output which can be used to control the flow inside the process.

The benefits of using BRE in REMINE are:

- A central place to store and retrieve complex rules
- Possibility to reuse rules without having to increase complexity in the processes.
- More efficient rules management (add, edit, delete)
- Data abstraction with BR implementation details hidden

For the scope of BR in REMINE and for the needs of the first and second prototype, Intalio BRE has been chosen as the technology to implement the real time rules. As seen from the following architectural diagram each process modeled in Intalio that addresses a specific clinical scenario, interfaces with an additional Intalio process that holds the business rules.

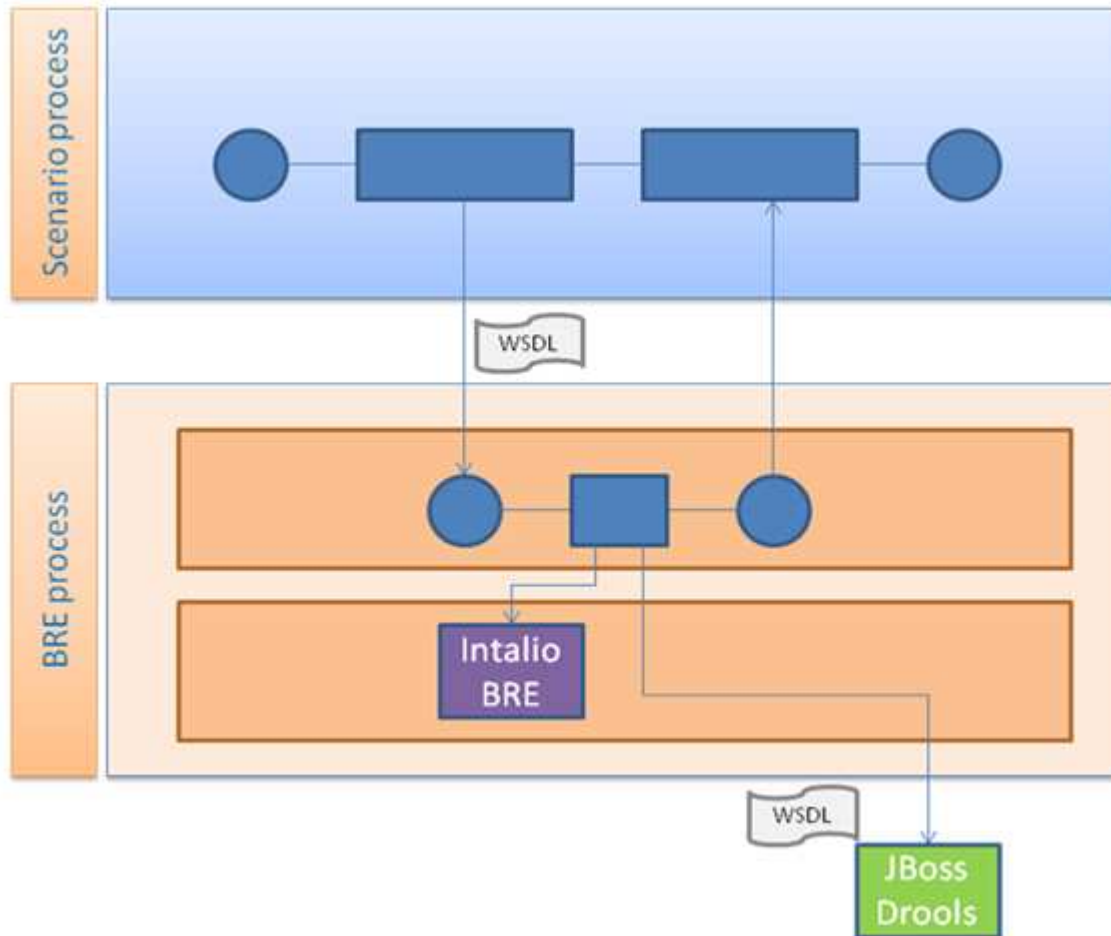


Figure 60 – Architectural diagram of BRE integration in BPEL processes

The integration is achieved by taking advantage of the supported Web Services framework from Intalio platform using an agreed interface between the involved technical partners. Every time the clinical process requires accessing a business rule the corresponding BRE process that holds the rules for each pilot will be consequently called and an output will be returned which is going to determine the course of an action. The clear separation of between control flow (clinical processes) and data logic (business rules) is handled in such a way so that changes in the implementation details on either modules does not affect the operation of the other. In addition, the BRE component can be extended in functionality or support a different technology such as JBoss Drools which is a more advanced and complete Business Rules engine.

In the following screenshot the actual Intalio BRE process is shown where it is deployed as an independent process. In this model the rules (conditions and values) that govern the Niguarda scenario are implemented using the Intalio built-in forms to create the table.

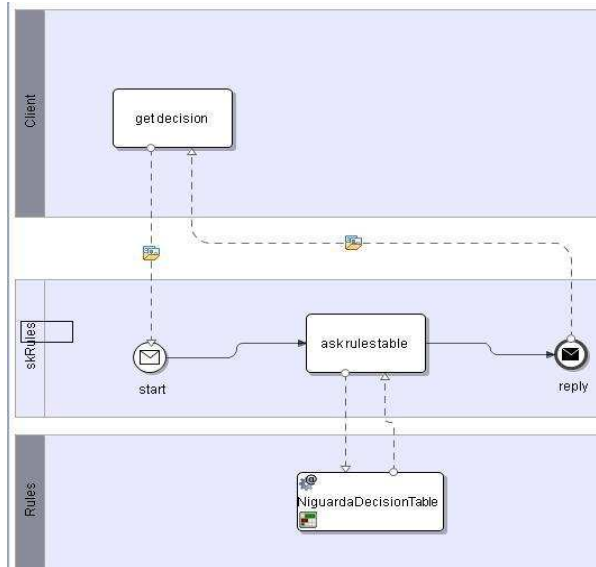


Figure 61 – BRE process

The following screenshots depicts parts of the actual NiguardaDecisionTable

ACTIONS		
Niguarda.aeTransferStrokeUnit	RemAction.message = {value}	RemAction.time = {
*	"Neurology assessment has not been completed within 30 minutes from triag"	"PT30M"
*	"Exams have not been requested within 30 minutes from triage"	"PT30M"
*	"Cerebral CT scan has not been requested within 30 minutes from triage"	"PT30M"
*	"Test tubes have not been delivered within 15 minutes from the exam request"	"PT15M"
*	"Test tubes have been checked-in by the Lab staff"	
*	"Exam results are available"	
*	"Exam results have not been delivered within 45 minutes from the triage"	"PT45M"
*	"Contact the Laboratory for a second evaluation."	
*	"Cerebral CT image is available."	
*	"Cerebral CT scan hasn't been assessed within 60 minutes from triage."	"PT60M"
*	"Reperfusion treatment hasn't been administered within 90 minutes from triage"	"PT90M"
true	"Transfer of the patient to the Stroke Unit hasn't been done within 70 minutes from triage. Reperfusion treatment should be administered in the A&E within 20 minutes."	"PT70M"
false	"Reperfusion treatment has to be administered within 20 minutes. An eventual transfer of the patient to the Stroke Unit has to be done now."	"PT70M"

CONDITIONS							
Niguarda.priorityCode	Niguarda.patientExit	Niguarda.neuroAssessment	Niguarda.examsComplete	Niguarda.cerebralCTScanCompleted	Niguarda.testTubesDeliveryToLab	Niguarda.testTubesCheckIn	
"YELLOW STROKE"	false	false	*	*	*	*	
		*	false	*	*	*	
		*	*	false	*	*	
		*	*	*	false	*	
		true	*	*	*	*	true
		*	*	*	*	*	*
		*	*	*	*	*	*
		*	*	*	*	*	*
		*	*	*	*	*	*
		*	*	*	*	*	*
		*	*	*	*	*	*
		*	*	*	*	*	*
		*	*	*	*	*	*

Figure 62 – Parts of a decision table (NiguardaDecisionTable)

The process map shows all steps and the activities together with inputs and outputs. The scope of the process mapping is to provide a unifying vision of business processes, so that the organization and the individuals have a common understanding. The “Process Mapper” serves two main functions the mapping of hospital’s processes and the association of those with data and Data Events. First the processes are modeled in BPMN format, then are transformed in BPEL format for being executable and be able to trigger the appropriate web services and finally are associated to data and Data Events and are enriched with KPIs (Key Performance Indicators). This module is dependent from the ReMINE sub-module “RAPS Management System” where the execution of the processes (BPEL transformation) is implemented.

6. Conclusion

Deliverable D.3.4 – Third Revision Data & Process Model system framework addresses both data management and process mapping components within the last prototype of the ReMINE project. The present deliverable outlines the metadatabase tools used in the development of WP3 services and the enhancements to the process mapper for the working version installed at all the pilots.

The process map shows all steps and the activities together with inputs and outputs. The scope of the process mapping is to provide a unifying vision of business processes, so that the organization and the individuals have a common understanding. The “Process Mapper” serves two main functions the mapping of hospital’s processes and the association of those with data and Data Events. First the processes are modeled in BPMN format, then are transformed in BPEL format for being executable and be able to trigger the appropriate web services and finally are associated to data and Data Events and are enriched with KPIs (Key Performance Indicators). This module is dependent from the ReMINE sub-module “RAPS Management System” where the execution of the processes (BPEL transformation) is implemented

7. Glossary

A&E-Accident and Emergency

API – Application Programming Interface

ASCII- American Standard Code for Information Interchange

BPD-Business Process Diagram

BPEL-Business Process Execution Language

BPM-Business Process Management

BPMN-Business Process Modeling Notation

CDA – Clinical Document Architecture

CT-Computed tomography

CRUD – Create, Read, Update, Delete

DEMS - Data Event Management System

ECG-Electrocardiogram

EFM- Electronic Fetal Monitoring

KPI - Key Process Indicator

HL7 – Health level seven

LDAP – Lightweight Directory Access Protocol

LOINC – Logical Observation Identifiers Names and Codes

RAPS - Risks against Patient Safety

RFID - Radio Frequency Identification

RIM - Reference Information Model

RLUS - Retrieve Locate Update Service

SAML – Security assertion markup language

SQL -Structured Query Language

TOS – Taxonomy and ontology service

UML - Unified Modeling Language

XACML - eXtensible Access Control Markup Language

XML – eXtensible markup language