Grant Agreement no. 643529

Empowering patients and strengthening self-management in cancer diseases

Research and Innovation Action

PHC-26-2014: Self management of health and disease: citizen engagement and mHealth

D5.3
Extended decision support and patient guidance services

Contractual Due Date: 31 July 2017
Actual Submission Date: 11 August 2017

Lead partner for deliverable: PHILIPS-NL

Dissemination Level: Public

Revision: v1.0

COVER AND CONTROL PAGE OF DOCUMENT	
Project Acronym:	***iManageCancer***
Project Full Name:	Empowering patients and strengthening self-management in cancer diseases
Project Duration	1 February 2015 - 31 July 2018
Deliverable No.:	D5.3
Deliverable Name:	Extended decision support and patient guidance services
Nature (R, DEM)[1]	DEM
Dissemination Level (PU, CO)[2]	PU
Version:	1.0
Actual Submission Date:	
Editor: Institution: E-Mail:	PHILIPS-NL
Contributors (Institution)	Stephan Kiefer (FRAU), Fatima Schera (FRAU), Michael Schäfer (FRAU), Eric Herve Ngantchjon (PHILIPS-NL), Anca Bucur (PHILIPS-NL), Haridimos Kondylakis (FORTH), Lefteris Koumakis (FORTH), Kostas Marias (FORTH)
Reviewers (Institution)	Feng Dong (BED)

Copyright

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 643529.

[1] **R** = Document, report (excluding the periodic and final reports), **DEM** = Demonstrator, pilot, prototype, plan designs

[2] **PU** = Public, fully open, e.g. web, **CO** = Confidential, restricted under conditions set out in Model Grant Agreement

Document History

Issue Date	Version	Changes Made / Reason for this Issue
01.08.17	0.7	First version for internal review.
10.08.17	1.0	First official version after review process.

Table of Contents

1 Executive summary

This deliverable describes the extended decision support and patient guidance services integrated in iManageCancer Platform that have been accomplished in work package 5.

This demonstrator comprises.

- a decision support framework consisting of the Care Flow Engine, a Care Flow Designer, a model repository service and client apps for patients and doctors.
- a Personal Medical Information Recommender service embedded in the iPHR of the iManageCancer Platform.
- new predictive model, OncotypeDX, for the prediction of the recurrence of women breast cancer at early stage.
- an alert workflow for early breast cancer based on OncotypeDX and St Gallen models, it uses BPMN 2.0 technologies
- a drug self-management service as part of the app iManageMyHealth that enables the patient to compose his medication plan and set reminders for drug intakes while the plan is checked for drug-drug interaction with the help of external internet based drug databases.
- a personal Health Information Recommender, which is targeted at improving the opportunities that patients have to inform themselves in the internet about their disease and possible treatments, and providing to them personalized information and recommendations.
- a decision aid to support patients' participation in consultations, the implementation is still in progress but the document presents use cases and an example for prostate cancer patients.

2 Introduction – Decision support and patient guidance services of iManageCancer Platform

Cancer research has led to more cancer patients being cured, and very many more enabled to live with their cancer.

As such, cancer is now considered as a chronic disease, patients and their families face the need to take an active role in their own care and in some cases in their treatment.

To this direction the iManageCancer project aims to provide a cancer specific self-management platform designed according to the needs of patient groups while focusing, in parallel, on the wellbeing of the cancer patient with special emphasis on avoidance, early detection and management of adverse events of cancer therapy but also, importantly, on psycho-emotional evaluation and self-motivated goals.

WP5 delivers for the iManageCancer Platform the central knowledge based system for guidance and support of the decision making of the patient as well as a set of interrelated tools that focus on specific aspects of this decision process.

The objectives of WP5 are:

a) to develop formal knowledge models for the management and self-management of side effects of cancer therapy, medication and long-term follow-up,

b) to develop a predictive model on adverse events for chemotherapy monitoring,

c) to execute these models in a Care Flow Engine and associated components of the iManageCancer Platform,

d) to develop an advanced personalized drug self-management tool,

e) to provide a Personal Medical Information Recommender as a decision aid to the patient,

f) to develop an alert workflow for adverse events based on the predictive model for chemotherapy monitoring,

g) to develop a specific decision aid for the consultation process that support patients' participation in clinical decision making and

h) to integrate these tools with the overall iManagerCancer Platform.

Use case for decision aid for the consultation process (g) is presented but the implementation is not part of the demonstrator and is subject to future versions.

3 System concept for decision support and patient guidance

The high-level architecture of the iManageCancer platform is shown in Figure 1. In orange are depicted the tools that are being reported in this deliverable and that will be analysed in the sequel.

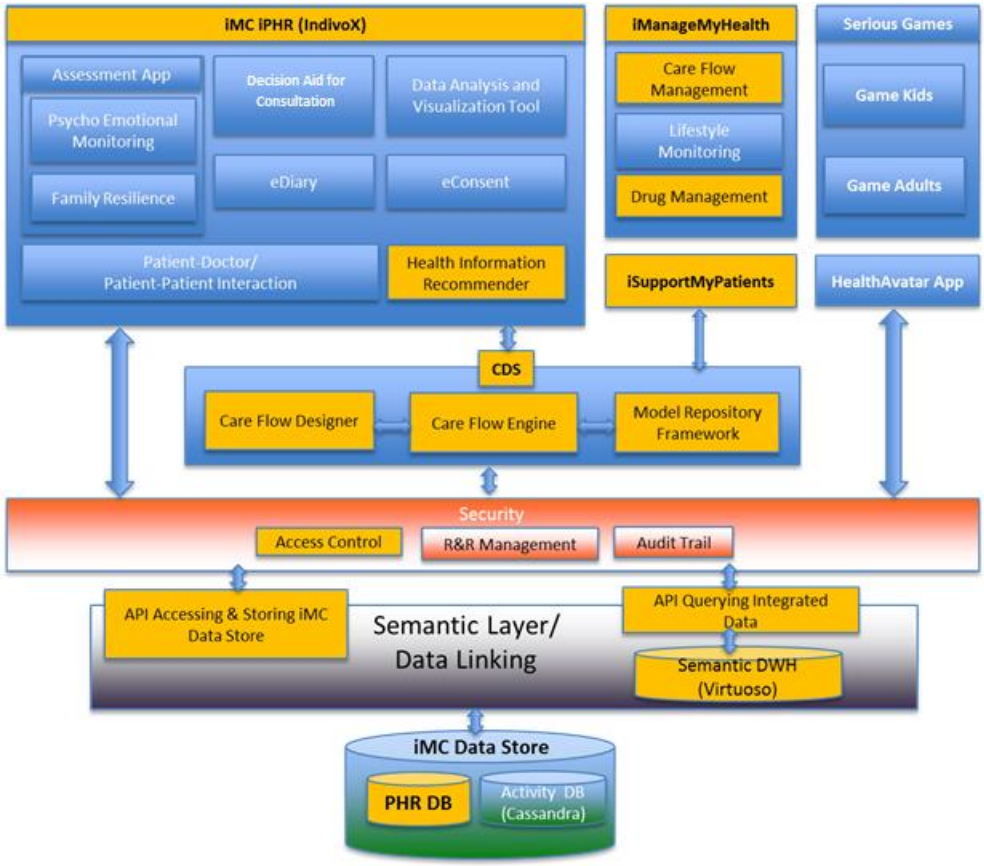

Figure 1: High level functional view on iManageCancer platform. The demonstrator that is described in this deliverable comprises the functional blocks marked in orange.

The decision support and patient guidance services of WP5 are provided by a central decision support system (CDS) embedded in the backbone of the iManageCancer Platform in combination with some end user tools in the apps iManageMyHealth, iSupportMyPatients and the PHR Portal. The CDS itself is formed by the Care Flow Engine and the Model Repository Framework to execute personalised and workflow oriented Care Flow Plans that can also leverage the results of predictive models at decision points. The Care Flow Designer shall allow clinical experts to design the corresponding Care Flow Diagrams. Two prerequisites for useful Care Flow Diagrams from which patients can benefit are a high degree of automation in the iManageCancer Platform that allows us to incorporate the services of the different backbone components in such Care Flow Diagrams. Secondly, apps must exist that can leverage the Care Flow Engine with its available Care Flow Diagrams and can incorporate this in its user interface concept. With iManageMyHealth and iManageMyPatients two apps have been developed in the context of WP5 (and WP6) that incorporate a client for the services of the Care Flow Engine. While iManageMyPatients has no further features than to use of or to contribute to specific Care Flows by physicians for their patients, iManageMyHealth offers also patients to manage their drugs independently of the Care Flow Engine. The original idea to develop Care Flow Diagrams that propose adjustments of drug doses was abandoned due to concerns on patient safety (see last Periodic Technical Report). An important aspect of Care Flows is to educate patients in the management of specific aspects of the disease. This is supported by offering the patients access to corresponding high quality web content provided by the Personal Health Information Recommender. This tool can be incorporated in Care Flow Diagrams but also be used by patients as a service in the iManageCancer iPHR. The decision support and patient guidance services are completed by the Decision Aid for Consultations. This questionnaire based instrument is still under construction.

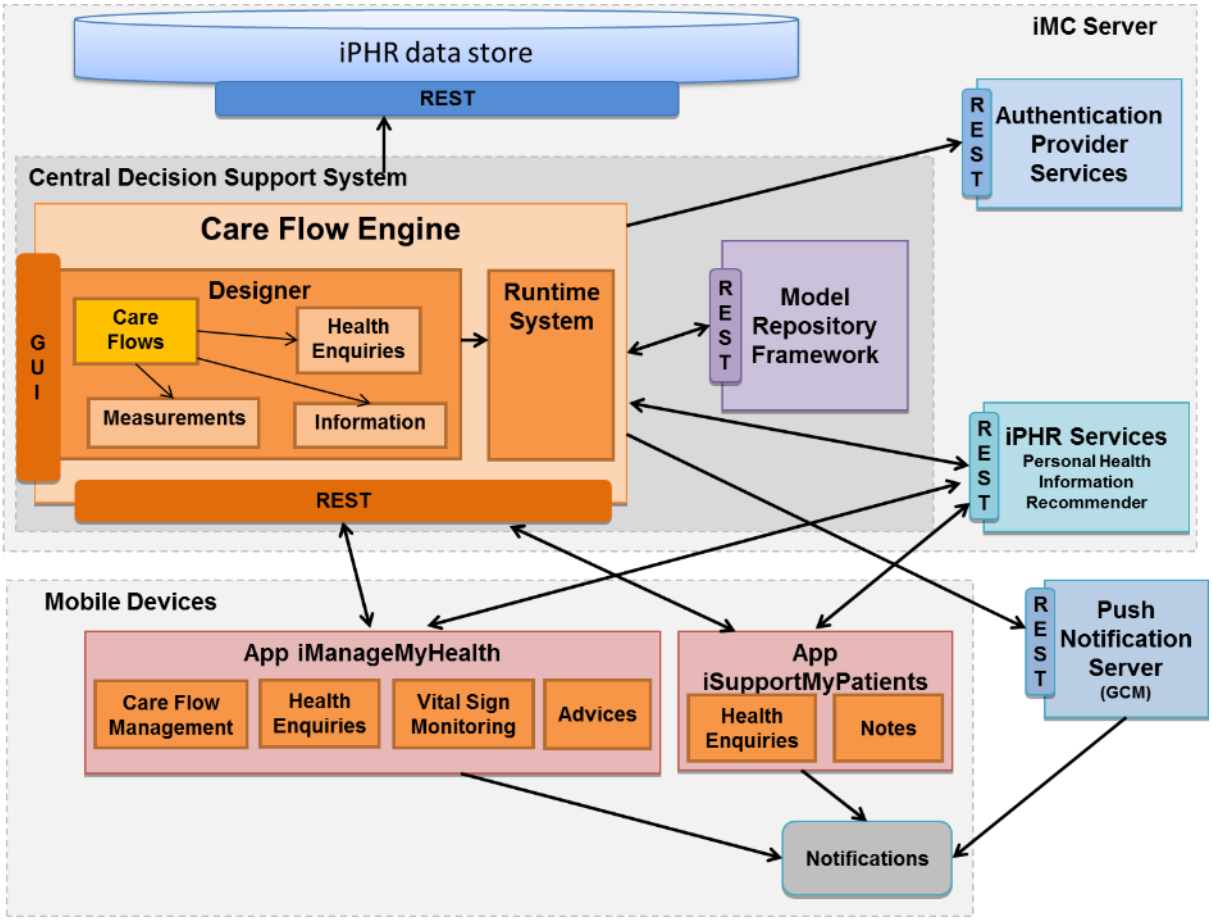

Figure 2: Care flow driven central decision support system of iManageCancer as the control instance of the applications of the users.

Figure 2 presents an architectural view from the perspective of the central decision support system with its Care Flow Engine and the model repository. Communication with other components of the platform is based on REST interfaces.

The Care Flow Engine represents a workflow-driven and business process management based approach to decision support that guides patients and doctors with their end devices and corresponding applications through the care pathway and supports them in decision making and management with corresponding precise information provision. The approach is presented in detail in D5.1.

The main idea behind this approach is that domain experts design formal definitions of the care flow based on clinical guidelines, knowledge on care pathways and an organisational model for integrated care with the patient as the co-manager of his health in the centre of it. These Care Flow Diagrams are personalised for a specific patient in individual Care Flow Plans and executed by the Central Decision Support Unit of the iManageCancer Platform, the so-called Care Flow Engine. Based on these Care Flow Plans the Care Flow Engine guides the patient but optionally also the healthcare team through the management of his disease and related co-morbidities by issuing tasks and recommendations to the patient and his healthcare team and by controlling the execution of the Care Flow Plan based on the results of tasks and monitored health status of the patient. In the design phase of the Care Flow Diagram, further knowledge is modelled as a set of clinical rules that control execution of the plan.

The iManageMyHealth app of the patient download and process these tasks for the patient issued by the Care Flow Engine. Tasks for patient are typically health enquiries and measurement tasks, but also information tasks with the recommendations obtained from the Care Flow Engine. Results are sent back to the Care Flow Engine for further assessment and control. Similarly, iSupportMyPatients, the app for physicians, downloads and process tasks for the patient's physician. The apps receive also notifications from the Care Flow Engine about tasks assigned to health professionals or patients.

An initial set of Care Flows was modelled in collaboration with our oncologists for different aspects of the management of cancer with a focus on those aspects of the disease that can be managed by the patient him/herself. Proper integration with the e-diary and PHR of patients was achieved by implementing client and server functionality in these systems in order to access the iMC data store or to download tasks, to process them and to send results of their execution back. A view on the high level architecture of Central Decision Support Unit of the iManageCancer Platform and its interfaces to the rest of the platform is presented in Figure 2.

Health enquiries represent a fundamental type of tasks in a Care Flow Diagram. As measurement tasks they represent a possibility to collect actual information on the patient's current condition by the patient himself or his doctors.

Several established predictive models in the cancer domain have been included in the model repository, such as St. Gallen and MASCC. They depend on data to be provided by the treating clinicians. Initial Care Flows have been constructed and integrated in the Care Flow Engine that leverage these models to demonstrate the power of the chosen approach.

4 Care Flow Engine

The following description is taken from deliverable D5.2. No changes have been made since then except some novel or modified care flow models that are presented at the end of this section.

4.1 Summary of use case and requirements from deliverables D2.2 and D2.3

Deliverable 2.2 proposes five areas in which patients can be guided by mobile apps in the management of their health:

- *Pain management*: This scenario aims to support patient's freedom from pain. An effective pain therapy is fundamental for the quality of life of cancer patients. Not every cancer patient suffers from pain but those who do can be relieved from pain with modern therapies. The sensation of pain is subjective and electronic tools are expected to support the self-management of pain.
- *Nutrition planning and drinking protocol:* This scenario aims to control patient's nutrition and drinking behaviour, especially for the early detection of malnutrition (i.e. primary anorexia cachexia syndrome due to tumour, or secondary anorexia cachexia syndrome due to the treatment).
- *Fatigue management:* This scenario aims to reach the best treatment and handling of chronic fatigue in cancer ("tumour fatigue" caused by cancer disease, or accessory symptom caused by e.g. drug side effect or associated cancer symptoms like anaemia, mental stress). This shall support the patients to budget their own energy and to avoid that they are overwhelmed as well as not enough challenged. Health professionals can work with patients with cancer to develop an activity/rest program based on an assessment of the patient's fatigue patterns that allows the best use of the individual's energy. Any changes in daily routine require additional

energy expenditure. Individuals with cancer are advised about setting priorities and maintaining a reasonable schedule.

- *Motion and exercise planning:* This scenario aims to provide the best motion and exercise plan to improve the well-being and efficiency of the patients.
- *Treatment and follow-up guidance:* This scenario aims to support the patient in managing his treatment and follow-up schedule. It may comprise smart scheduling of diagnosis and therapy examinations, reminders to necessary appointments and recognizing treatment gaps, assistance in preparing appointments (e.g. creation of checklists), and supporting the treatment of side effects and symptoms.

Deliverable D5.1 further elaborates initial knowledge models for pain management, fatigue management and Febrile Neutropenia (FN) monitoring on the basis of MASCC risk stratification. These scenarios drive the development of the decision support and guidance backbone in iManageCancer with the corresponding apps for patients and doctors.

Deliverable D2.3 contains the use case diagrams associated with pain management and fatigue management. Although they present the way forward, they are too complex to implement them in the care flow management oriented approach for decision support of in iManageCancer as they require a very tight integration of the different apps and tools in the platform which is not the case yet. Deliverable D5.2 presented initial implementation of this workflow management approach with the so-called Care Flow Engine, together with a graphical design tool for simple management models and some guiding Care Flow models for demonstration purposes. In the demonstrator described by deliverable D5.3 we included very simplistic models for the management of pain and fatigue with the focus on the monitoring of these side effects by the patient himself.

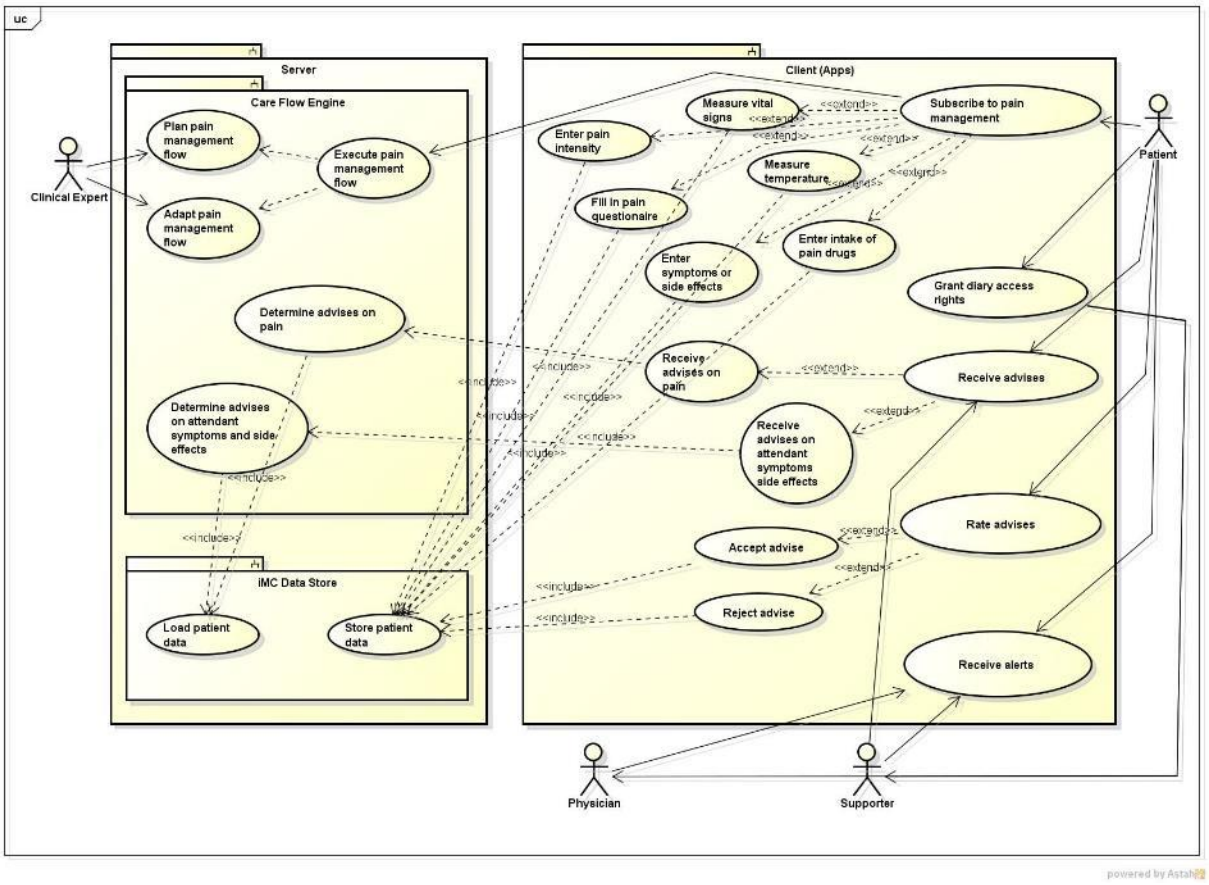

Figure 3: Use cases diagram for Pain Management as described in deliverable D2.2.

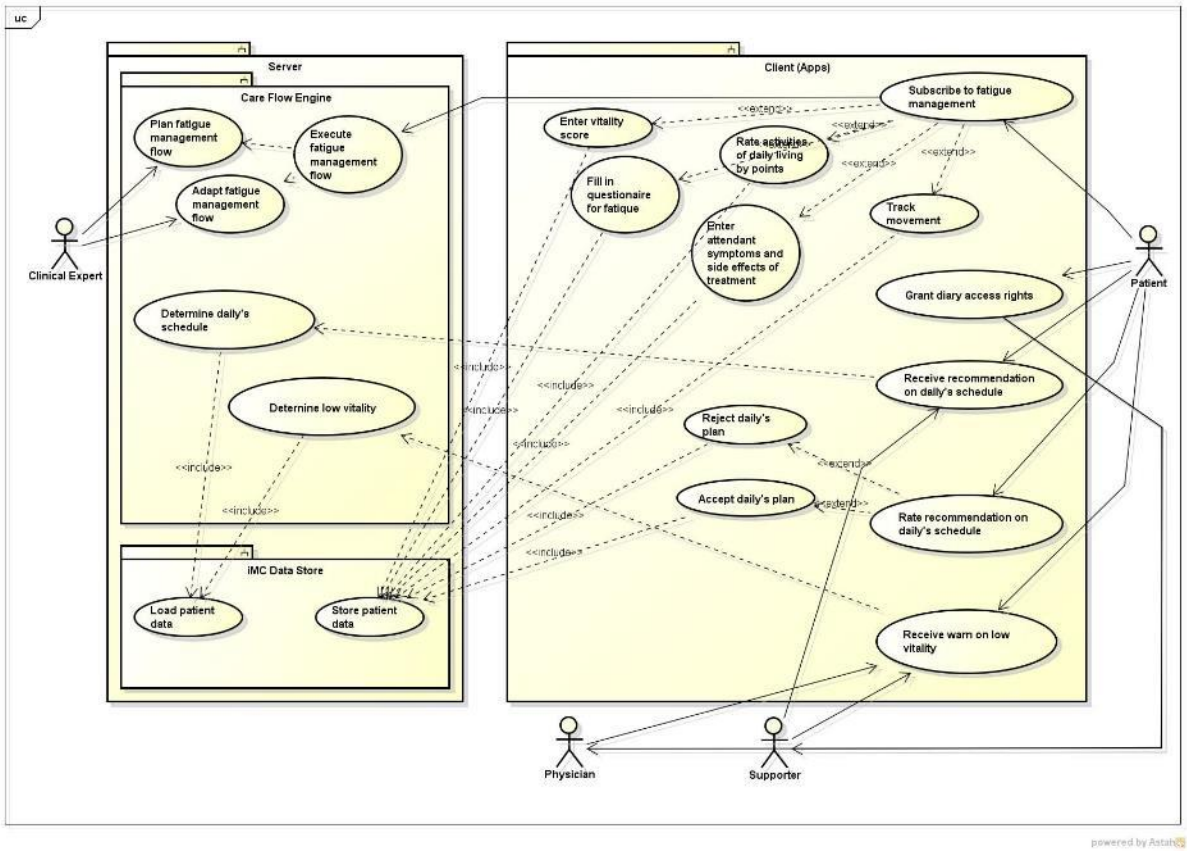

Figure 4: Use cases diagram for Fatigue Management as described in deliverable D2.2.

Three general use cases associated with these scenarios are further described in deliverable D2.3 to drive tool development. We summarize them in the following table.

ID	Name	Description	Priority	Imple-mented?
UC.CFD.1	Design of care flows to be executable in the system	Self-Management workflows can be designed by clinical experts as care flows and deployed for an execution in the system.	required	Yes. Initial version available with limited capabilities regarding designable care flows
UC.CFD.2	Adaption of existing care flows.	In the system existing and deployed care flows can be changed and adapted	required	Yes. Very basic version available
UC.CFE.1	Execution of care flows.	Self-Management workflows can be selected and executed as care flows by patient, patient's supporters and attending physician.	required	Yes

D2.3 derives various functional and non-functional system requirements from these general use cases for the planned tools Care Flow Engine and Care Flow Designer. They are briefly listed in the following tables.

Requirements for Care Flow Designer:

ID	Name	Description	Imple-mented?
REQ.CFD.1	GUI for designing care flows which are executable in the system.	System should provide a GUI for creating, storing and adapting care flows for the usage in the system.	Yes. Initial version available with limited capabilities regarding designable care flows
REQ.CFD.2	Appropriate usability of care flow designer.	Care flow designer should be intuitive and easy to use for a clinical expert.	Yes
REQ.CFD.3	Secured access to care flow designer using iManageCancer user and roles management.	Care flow designer should be secured by a user authentication. User should have an authority as a clinical expert. This should be defined and provided by the iManageCancer system.	Yes
REQ.CFD.4	Appropriate functionality of care flow designer	Care flow designer must support the required functionality to implement the management workflows of the iManageCancer system.	Yes, partially.
REQ.CFD.5	Definition of enquiry tasks as part of a care flow	Care flow designer should support defining enquiry tasks as part of a care flow. These tasks should provide definition of health enquiries requirements (see below) and should be executed by proper apps.	Yes
REQ.CFD.6	Definition of measurement tasks as part of a care flow.	Care flow designer should support defining measurement tasks as part of a care flow. These tasks should be executed by proper apps which are able to measure or record the specified vital signs (blood pressure, pulse, body temperature) or body weight.	Yes
REQ.CFD.7	Definition of advice tasks as part of a care flow	Care flow designer should support defining advice tasks as part of a care flow. These tasks should be executed by proper apps which are able to display the specified information to an actor.	Yes
REQ.CFD.8	Definition of warning tasks as part of a care flow.	Care flow designer should support defining warning tasks as part of a care flow. These tasks should be executed by proper apps which are able to display the specified information to an actor.	No. (They represent special information tasks)
REQ.CFD.9	Definition of decision points as part of a care flow.	Care flow designer should support defining decision points as part of a care flow. This comprises the assignment of a specified logic and the definition of conditions for the outgoing branches.	Yes. However conditions are very

			basic terms.
REQ.CFD.10	Encoding of decision support logic and assignment to decision points of care flows.	It should be possible that available services for executing decision support (e.g. predictive models) can be selected and assigned to decision points of care flows.	No. Under preparation
REQ.CFD.11	Selection and assignment of decision support services to decision points of care flows	The drug self-management app should be functional without an internet connection for accessing the iManageCancer PHR and a drug database.	Yes, partially
REQ.CFD.12	Assigning entities and attributes of iManageCancer data model to variables in the care flow design.	It should be possible that entities and attributes of iManageCancer data model can be selected and assigned to variables of care flows (e.g. task properties, service parameter, logic variables). This should allow synchronising the collected data with the iManageCancer data storage during runtime	No. Under preparation.
REQ.CFD.13	Services for retrieving structured information about iManageCancer data model.	Services should be available for retrieving structured information about iManageCancer data model in order to use these for assigning to care flow variables.	Yes
REQ.CFD.14	Defining role of the tasks executors in the care flow design	The Care Flow Designer must allow selecting and assigning the executor role (e.g. patient, patient's supporter, and physician) to each task.	Yes
REQ.CFD.15	Services to access user and roles of the iManageCancer platform.	The iManageCancer system must provide services to access available users and roles	Yes
REQ.CFD.16	Configuration of Care Flow Designer.	Care flow designer should be configurable by a GUI form (e.g. components, services).	No

Requirements for Care Flow Engine:

ID	**Name**	**Description**	**Imple-mented?**
REQ.CFE.1	Query interface for available care flows.	CFE should provide service for querying available care flows. Query can be filtered by specific parameters (e.g. types of tasks).	Yes
REQ.CFE.2	Service interface for executing care flows.	CFE should provide service for executing care flows on demand.	Yes
REQ.CFE.3	Selection of care flows for execution.	Proper patient apps must be available which allow selecting and executing care flows.	Yes
REQ.CFE.4	Executing care flows at specified points of time.	It should be possible that care flows will be executed at a specified point of time automatically.	Yes
REQ.CFE.5	Executing care flows triggered by system events.	It should be possible that care flows will be triggered by a specified (system) event automatically.	No
REQ.CFE.6	Query interface for open tasks.	CFE should provide service for querying open tasks. Query can be filtered for specific task types (e.g. enquiry, measurement) and assignee (user id).	Yes
REQ.CFE.7	Service interface for confirming tasks.	CFE should provide service for confirming tasks. Method should return an error code if required data is missing of task is expired.	Yes

REQ.CFE.8	Requesting and executing care flow tasks by apps.	Apps installed on actor's smart device must be available in order to request, execute and confirm tasks of a running care flow.	Yes
REQ.CFE.9	Executing logic assigned to decision points of a care flow.	System must provide mechanisms (rule engine, interpreter) in order to execute the defined logic during care flows runtime.	Partially
REQ.CFE.10	Calling decision support services assigned to decision points of a care flow.	The Care Flow Engine can call services for requesting decision support scores during runtime. Decision support components must provide the required interfaces.	No
REQ.CFE.11	Loading required data from patient's PHR, eCRFs and eDiary for the execution of care flows.	Required data from patient's PHR, eCRF and eDiary can be loaded during the runtime of care flows. The iManageCancer data storage must provide the required interfaces.	Yes, partially
REQ.CFE.12	Storing collected data in patient's PHR during runtime of care flows.	Collected data can be stored in patient's PHR during runtime of care flows. The iManageCancer data store must provide the required interfaces.	Yes
REQ.CFE.13	Storing collected data as an entry or attachment of an entry in patient's eDiary during runtime of care flows.	Collected data can be stored in patient's PHR during runtime of care flows. The iManageCancer data storage must provide the required interfaces.	Yes

4.2 Internal architecture of Care Flow Engine

The architecture of the Care Flow Engine with its interfaces as shown in Figure 2 is explained in the previous chapter. When a new Care Flow Diagram is deployed in the system it is converted in the mentioned Activiti BPMN object model for its execution by the Care Flow Engine. During the execution of the care flow for an individual patient client apps are able to query for user tasks. The different types of tasks are rendered and presented to the user in the app.

The internal architecture is depicted in the following picture. The kernel is provided by the open source BPMN Engine Activiti[3].

[3] https://www.activiti.org/

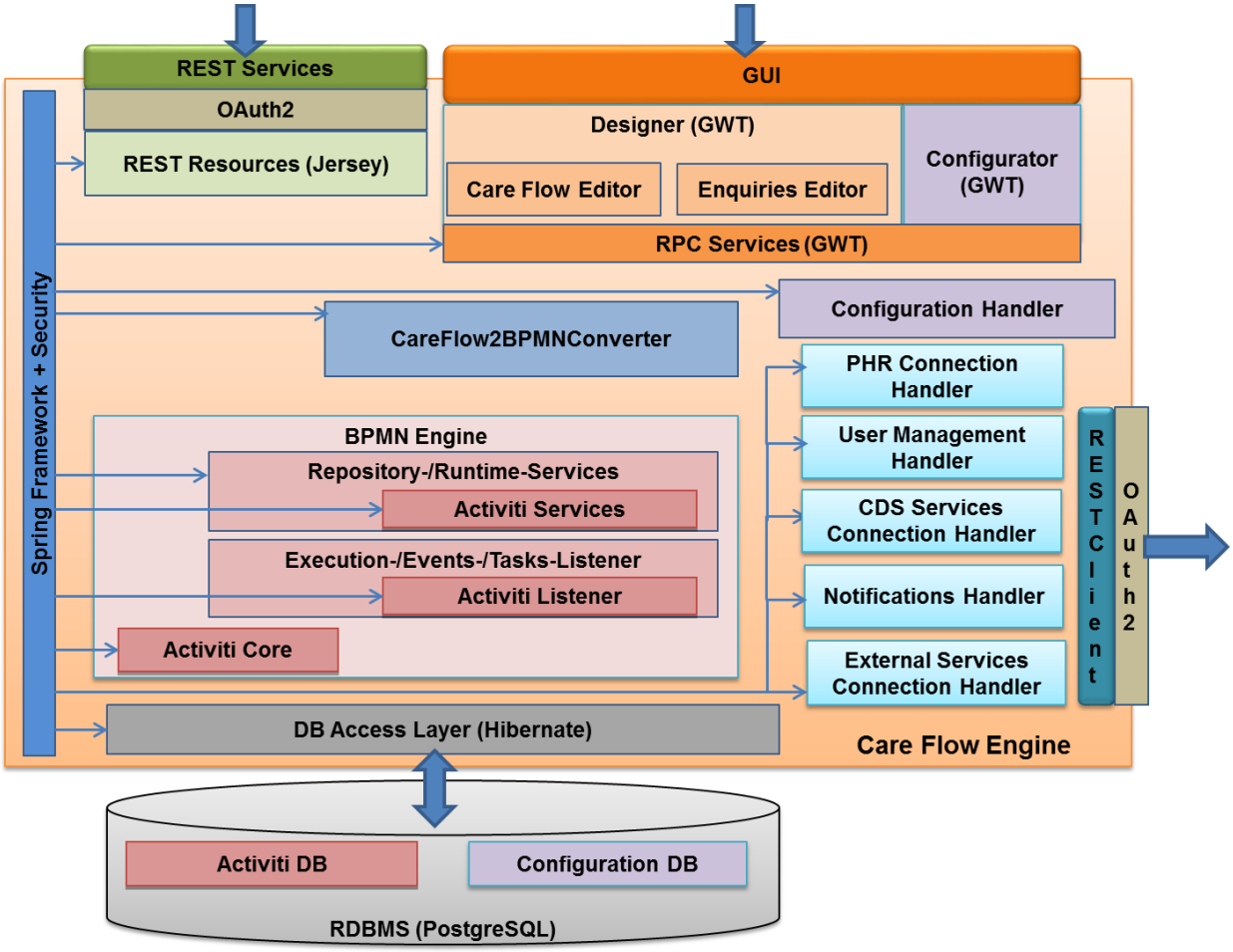

Figure 5: Internal architecture of the Care Flow Engine.

It provides the following functionalities:

- Deploying and running Activiti BPMN workflows.
- Providing repository services.
- Providing runtime services.
- Querying tasks (with properties)
- Integration of external components via listener (events, gateways and tasks).
- Persistence storage of process definitions and processes.

In the following table features are listed which we developed to adapt an Activiti process definition to a Care Flow and vice versa.

Feature	Implementation in Activiti
Assigning patient to a Care Flow process	Process variable "PATIENT_ID": • Configurable as form property of a start event • Parameter for RuntimeService.startCareFlowProcess UserManagement services will be used to get additional user data (e.g. assigned physicians and companions)
Assigning admin(s) to a Care Flow process	Process variable "adminIdList": • Configurable as form property of a start event • Parameter for RuntimeService.startCareFlowProcess UserManagement services will be used to get user data

Timer triggered process start	Process variable "TIMERS": • Configurable as form property (variable "TIMERS_AS_JSON" in JSON) of a start event (e.g. `[{"id":"timer1","times":["08:00"],"dayCycle":1}, {"id":"timer2","times":["08:00","18:00"],"dayCycle":1}]`) • Parameter for RuntimeService.startCareFlowProcess Each timer consists of • "id" • "times" (of a day) • "dayCycle" (e.g. "2" for every second day). Process variable "ACTIVE_TIMER_ID" determine the currently used timer. Process variable "START_TIME" will be calculated from the active timer (in a Listener of the start event) and set in an intermediate TimerCatchingEvent at process begin. At the end of the process the next start time will be calculated (in a listener of an inclusive gateway). If the startTime is null the process will be finished otherwise the process starts again.
On-demand process start	If no timer and no activeTimerId are defined a process will start immediately.
Event triggered process start	Process variable "START_MESSAGE" and MessageStartEvent? (remark: planned)
Language	Process variable "LANGUAGE": • Configurable as form property of a start event • Parameter for RuntimeService.startCareFlowProcess Process variable will be used in tasks to resolve multi-language labels.
Enabling sending notifications during a process	Process variable "NOTIFICATIONS_ENABLED": • Configurable as form property of a start event • Parameter for RuntimeService.startCareFlowProcess
Definition of an initial flow	Process variable "PROCESS_LOOPS" serves as counter and can be used to identify the first loop (PROCESS_LOOPS == 1). Process starts with a parallel gateway which comprises a conditional branch for the first loop.
Identifying user who starts process	Using process variable "initiatorId".
Assigning task assigned to user(s)	Using task property "assignee" (for single user, i.e. patient) or "candidateUsers" (for list of users, e.g. physicians or companions).
Delegating tasks to a Web App	Using specific form property "name", i.e. name = `{"valueType":"ForwardLink"}`
Health Enquiry	• Set task property "category" to "HealthEnquiryTask" • Using form properties to specify the question items
Measurement	• Set task property "category" to "MeasurementTask" • Using form properties to specify the measurement items
Information	• Set task property "category" to "InformationTask" • Using form properties to specify the informationText and confirmQuestion
Multi-lingual labels	Using JSON format encoding in • "documentation" property of process definition and tasks • form properties of user tasks e.g. `{"labels":[{"label":"Body Temperature Measurement","language":"en"}, {"label":"Körpertemperaturmessung","language":"de"}, {"label":"Temperature misurazione","language":"it"}], "descriptions":[{"label":"Request to measure patient's body temperature", "language":"en"}, {"label":"Anfrage zur Messung der Körpertemperatur des Patienten","language":"de"}, {"label":"Richiesta per misurare la temperatura corporea del paziente","language":"it"}]}`

Customised datatypes, e.g. "smilies" score (1 - 10)	• Implement class extended from AbstractFormType (e.g. "ScoreFormType") • Configure property "customFormTypes" in the SpringProcessEngineConfiguration (applicationContext.xml)
Calling PHR to store data	Using Activiti listener on • Task completed • Gateway completed Listener field "phrService" to define (in JSON format) • Service resource • Input values (especially variables) E.g. `[{"serviceCode":"iMC_IndivoX","key":"Body_Temperature", "operation":"CREATE", "inputValues":[{"name":"Field", "alt":"value", "variable":"bodyTemperatureMeasurement"}]}]`
Calling PHR to load data	Using Activiti listener on • Task created • Gateway started Listener field "phrService" to define (in JSON format) • Service resource • Input and output values (especially variables) E.g. `[{"serviceCode":"iMC_IndivoX","key":"Body_Temperature", "operation":"READ", "inputValues":[{"name":"Field", "alt":"value", "variable":"bodyTemperatureMeasurement"}]}]`
Calling CDS services	Using Activiti listener on • Task created • Gateway started Listener field "cdsService" to define (in JSON format) • Service resource • Input and output values (especially variables) E.g. `[{"serviceCode":"MRF_ModelRunner", "key":"MRF ModelRunner Execute", "operation":"CREATE", "inputValues":[{"name":"modelInputValue", "alt":"BURDEN OF ILLNESS", "variable":"mASCCRiskIndexFactors_burdenOfIllness"}, {"name":"modelInputValue", "alt":"HYPOTENSION", "variable":"mASCCRiskIndexFactors_hypotension"}, {"name":"modelInputValue", "alt":"PULMONARY DISEASE", "variable":"mASCCRiskIndexFactors pulmonaryDisease"}, {"name":"modelInputValue", "alt":"SOLID_TUMOR_OR_NO_FUNGAL_INFECTION", "variable":"mASCCRiskIndexFactors_solidTumorOrNoFungialInfection" }, {"name":"modelInputValue", "alt":"OUTPATIENT", "variable":"mASCCRiskIndexFactors outpatient"}, {"name":"modelInputValue", "alt":"DEHYDATRATION", "variable":"mASCCRiskIndexFactors_dehydration"}, {"name":"modelInputValue", "alt":"AGE", "variable":"mASCCRiskIndexFactors_age"}], "outputValues":[{"name":"modelInputValue", "variable":"masccModelResult"}]}]`
Send notifications if tasks are available	Using Activiti listener on Task created
Decision points with conditional branches	• Using ExclusiveGateway • Define outgoing branches • Define default flow
Parallel branches	• Using ParallelGateway • Define outgoing branches
Conditional branches	• Define condition (boolean expression) for branch • Define listener with expression to set process variables
Exception handling	• Define ErrorEndEvent with an errorCode • Connect ErrorEndEvent with a conditional branch to a Gateway / Task • Define EventSubProcess with a ErrorStartEvent with same errorCode
Task expired handling	• Define TimerBoundaryEvent with property "Time duration"

	• Set task property “Due date” with same value

The Activiti DB is explained in
http://activiti.org/userguide/index.html#database.tables.explained

The following internal services provide access to the backend functionality of the Care Flow Engine based on the Activiti API.

Interface	Implementation	Injected Services	Methods
IRepositoryService	ActivitiRepository Service	RepositoryService, IdentityService	• queryCareFlows(String userId, String patientId, String id, String key, String nameLike): List<CareFlowDTO> • getCareFlow(String userId, String key): CareFlowDTO • getBpmnDiagram(String processDefinitionId): String • getCareFlowDiagram(String deploymentId): String • deployBpmnDiagram(String bpmnResource, InputStream inputStream): CareFlowDTO
IRuntimeService	ActivitiRuntime Service	RuntimeService, IdentityService, HistoryService, FormService	• queryProcesses(String userId, String careFlowKey, String initiatorIdFilter, String patientIdFilter, boolean onlyActive): List<ProcessDTO> • getProcess(String userId, String processId): ProcessDTO • startProcess(String userId, String processDefinitionId, ProcessVariablesDTO processVariables):ProcessDTO • stopProcess(String userId, String processId, String reason): ProcessDTO
ITaskService	ActivitiTask Service	TaskService, FormService, IdentityService, HistoryService, IPHRConnection Service, IPHRSettingDAO	• queryTasks(String userId, String assignee, String careFlowKey, String processId, String category, Integer priority, Boolean active): List<TaskDTO> • getTask(String userId, String taskId): TaskDTO • completeTask(String userId, String taskId, List<FormPropertyDTO> formProperties):TaskDTO

The following table summarizes the used Activiti Listener classes which allow handling specific events in a running Care Flow.

Implementation	Activiti Interface	Injected Services	CFE Functionalities
Activiti-StartEventListener	ExecutionListener	-	• Set timers if they are configured • Set startTime to now if it is not set yet.
Activiti-SubProcess-	ExecutionListener	-	• Increase loop variable

StartEventListener			
Activiti-TaskCreateListener	TaskListener	INotificationsHandler, ICDSServiceConnectionHandler, IPHRConnectionHandler, IConfigurationHandler	• Calling Notification services (e.g. GCM) if enabled. • Calling CDS services according to field "cdsServices". • Read data from PHR according to field "phrServices"
Activiti-TaskCompleteListener	TaskListener	IPHRConnectionHandler	• Store data in PHR according to field "phrServices"
Activiti-GatewayStartListener	ExecutionListener	ICDSServiceConnectionHandler	• Calling CDS services according to field "cdsServices"
Activiti-GatewayEndListener	ExecutionListener	IPHRConnectionHandler	• Store data in PHR according to field "phrServices"
ActivitiTimerGateway Listener	TaskListener		• Determine next startTime

Datatypes
Supported types are "string", "long", "double", "boolean", "enum", and "date".
Customized types are "score" (->ScoreFormType).

Customized datatypes can be defined as described in http://stackoverflow.com/questions/15331195/how-to-make-activiti-show-textarea-field-instaded-of-text-input.

4.3 User interface functionality of Care Flow Engine

The Care Flow Engine is a web application. Its user interface, which is shown in the following figures, contains three components:

- A designer to construct Care Flow Diagrams in a customized BPMN as well as an editor to create enquiries,
- a configurator to configure the Care Flow Engine, and
- a runtime dashboard to manage the deployed Care Flows Diagrams and to monitor their execution.

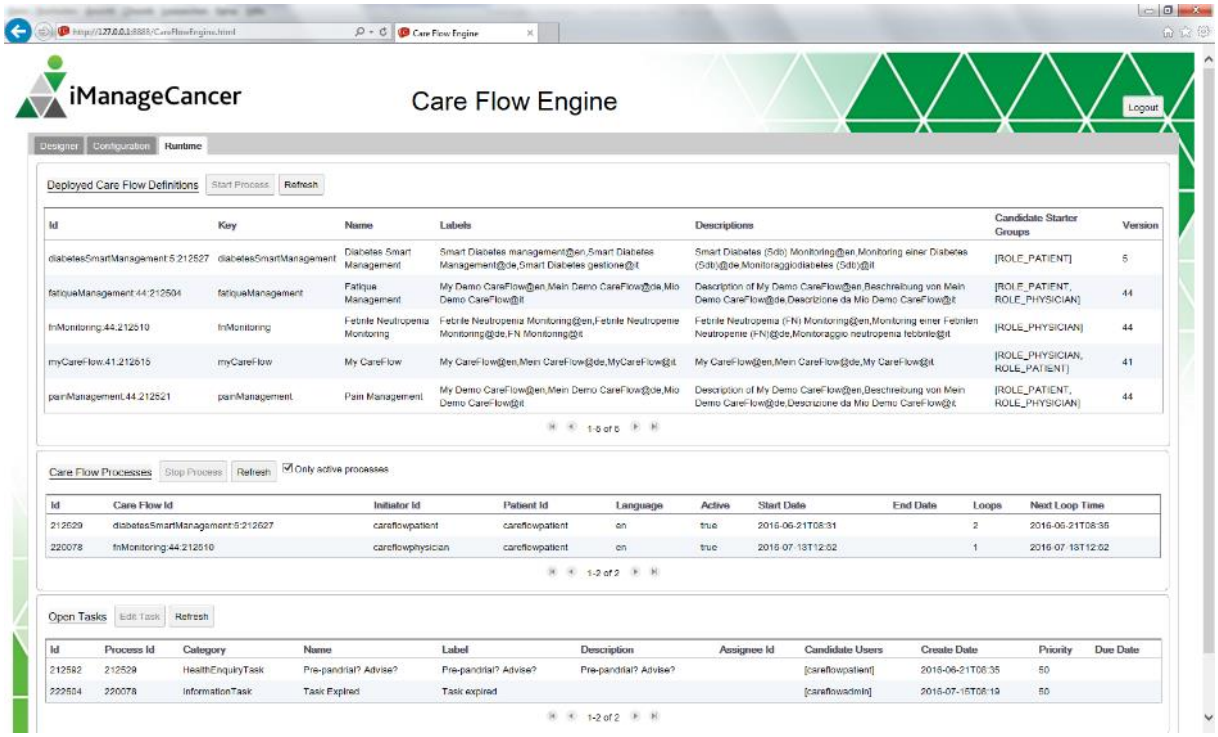

Figure 6: Main user interface of Care Flow Engine with three tabs 'Designer', 'Configuration' and 'Runtime'. This screenshot shows the 'Runtime'.

The GUI of the Runtime provides the following functionality:

- List of Care Flow Diagrams deployed in the system ('Care Flow Definitions'),
- Monitoring of the processes of the executed Care Flow Processes for individual patients,
- Monitoring of the open tasks of these processes,
- Possibility to simulate executing processes and tasks in the runtime environment.

In the following screenshot, a Care Flow Process is started for a specific patient. This feature is made available for test purposes. Usually, the Care Flow Plans are started by the users themselves with their apps by calling the corresponding REST service of the Care Flow Engine.

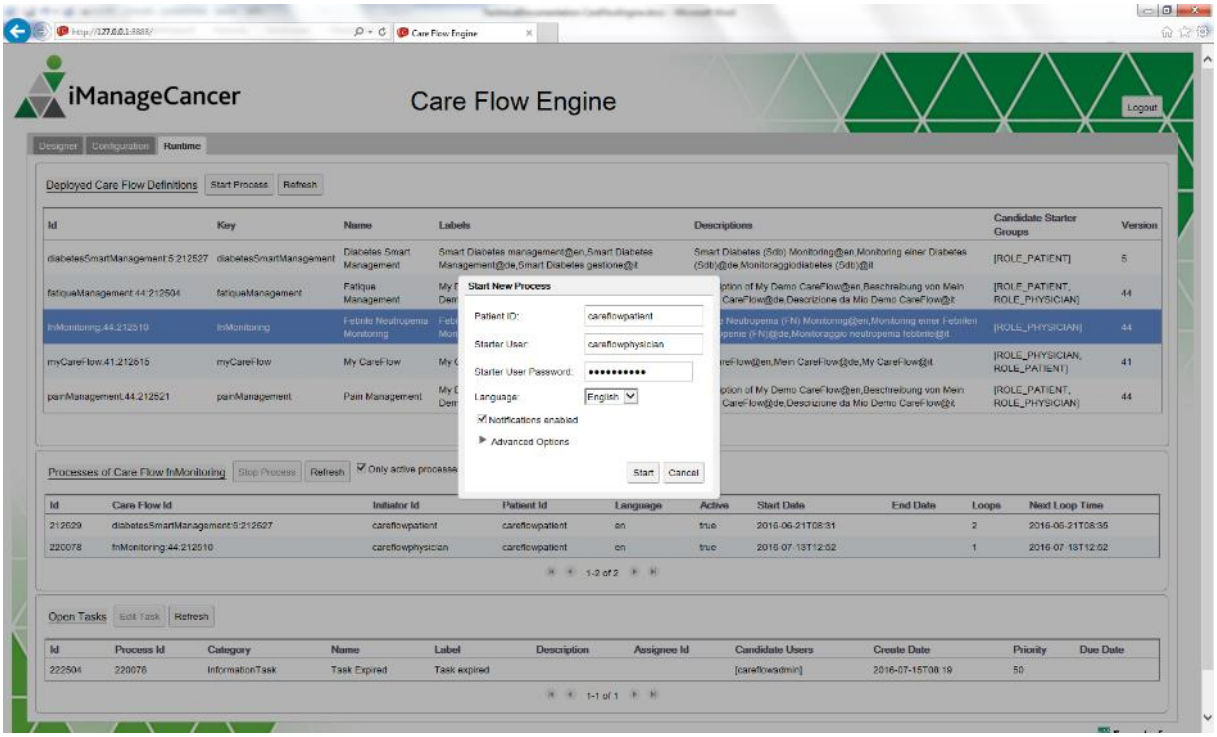

Figure 7: Dialogue 'Start New Process' in the Runtime dashboard.

In the following screenshot, an open enquiry task for a specific user was edited. Answers can be given in this dialogue for testing purposes. Usually open tasks are presented to the corresponding user in his app. The app of the user gathers the answer from him and sends it back to the Care Flow Engine through the corresponding REST service of the Care Flow Engine.

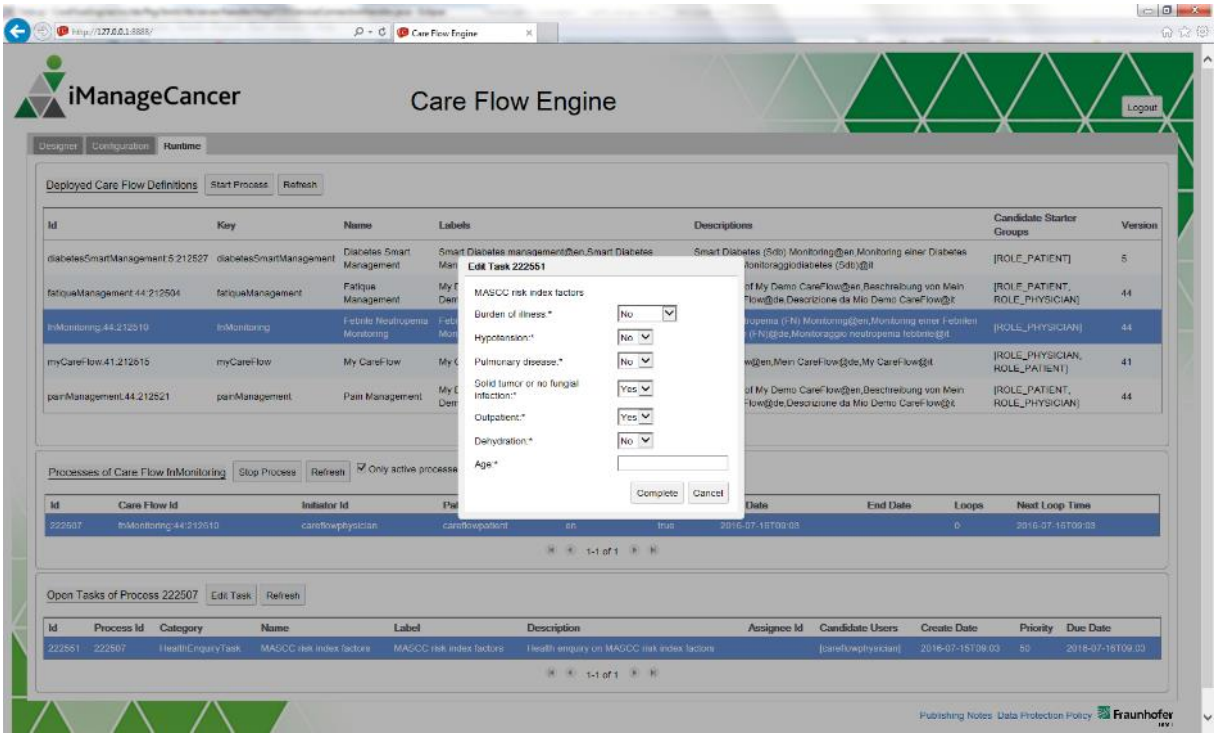

Figure 8: Dialogue 'Edit task' in the Runtime dashboard.

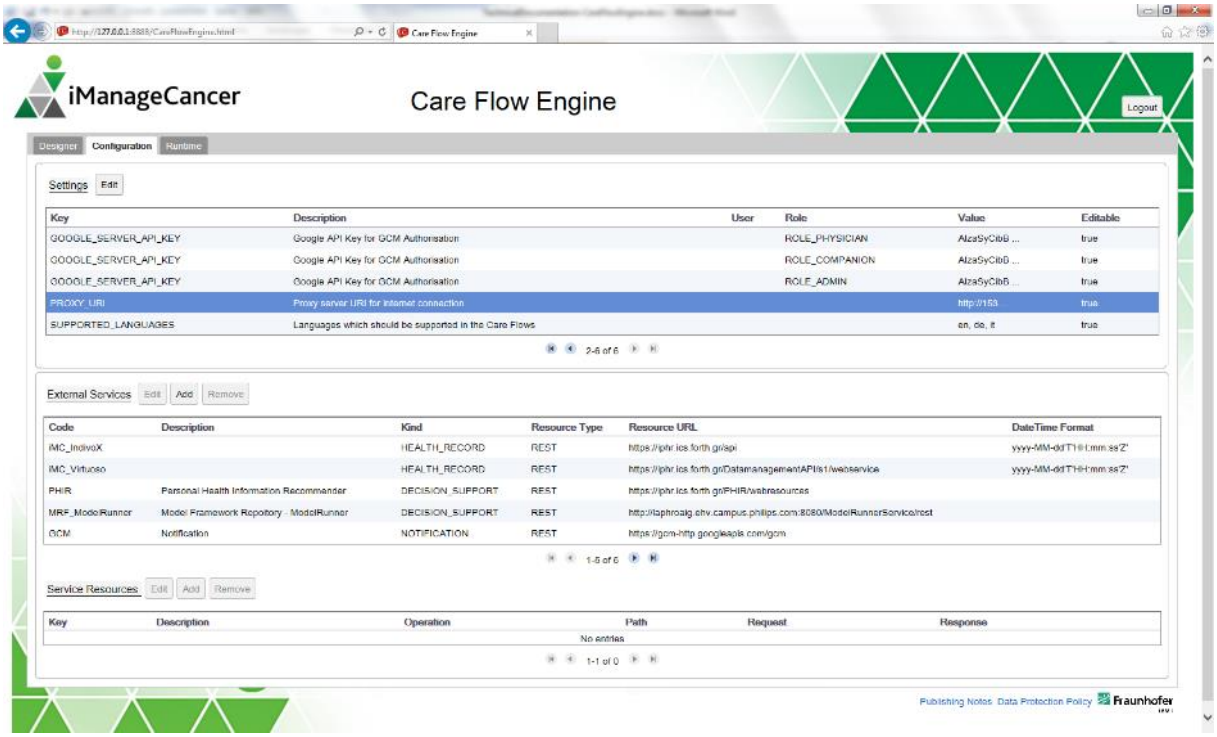

Figure 9: Main user interface of Care Flow Engine with three tabs 'Designer', 'Configuration' and 'Runtime'. This screenshot shows the 'Configuration'.

Figure 9 shows the configurator. The GUI offers

1. to specify external services with their keys as resources that can be used in care flows. The following services are actually available to the Care Flow Engine:
    - iManageCancer Health Record
    - iManageCancer Model Repository
    - Notification service Google Cloud Messaging (GCM)
    - Central iManageCancer User Management
2. configurations
    - Languages which should be supported in the care flows
    - Proxy server URI for internet connection
    - Google API Key for GCM authorisation
    - GCM tokens for addressing users in the push notifications. They are entered by apps of users through the corresponding REST service of Care Flow Engine.

4.4 Creating Care Flows with Care Flow Designer

The graphical user interface of the designer for care flows is presented in the following figure with an example of a Care Flow Diagram composed of different types of tasks.

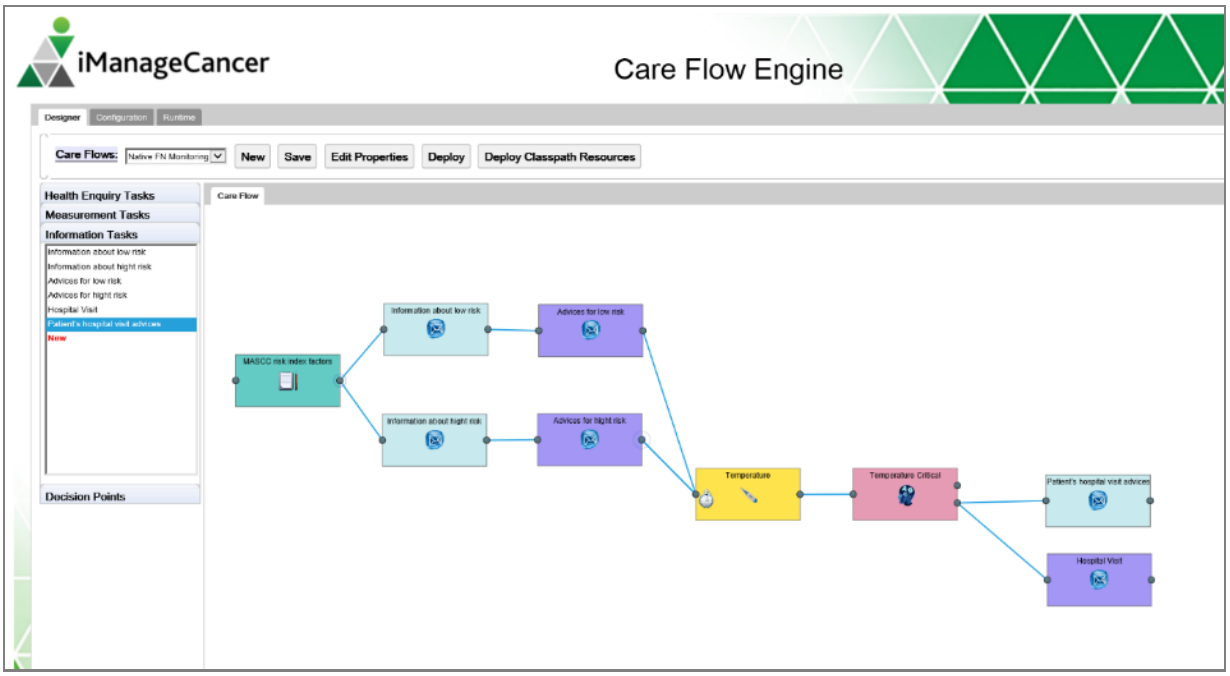

Figure 10: Canvas of the designer with an exemplary Care Flow Diagram.

The GUI of the Designer has the following functional areas:

- Upper toolbar for the management of Care Flow Diagrams with the following functions:
  - **Select menu 'Care Flows':** Serves to select a care flow from the internal database.
  - **New**: Create a new Care Flow Diagram.
  - **Save**: Store a care flow in the internal database.
  - **Edit Properties**: Opens the Edit-dialogue for the properties of the care flow.
  - **Deploy:** Deploys a care flow from the database in the runtime environment of the Care Flow Engine.
  - **Delete:** Delete a workflow from the database and the BPMN repository
  - **Deploy Classpath Resources:** Deploys Care Flow Diagrams stored in XML notation in the runtime environment of the Care Flow Engine. This is an alternative method for software developers to deploy Care Flow Diagrams that have been developed with external tools with a software update of the Care Flow Engine.

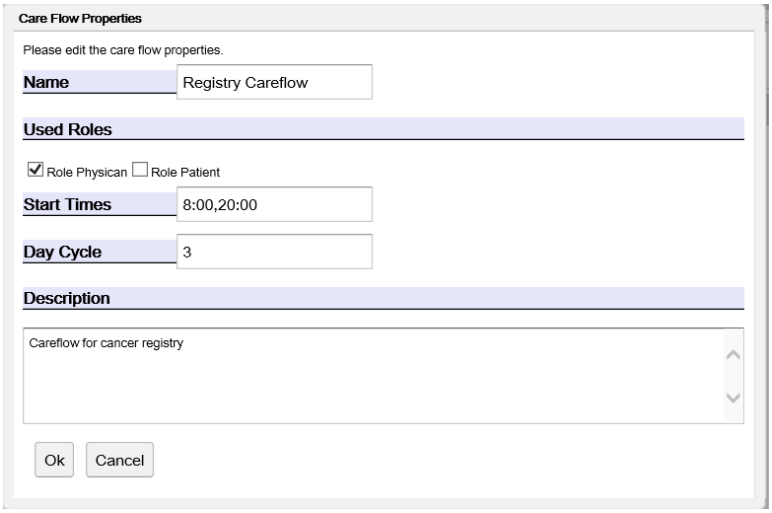

Figure 11: Editor for properties of the care flow.

- Left toolbar serves to select an element to edit it and add it to a Care Flow Diagram in the canvas. The following type of elements can be chosen:
    - health enquiry task
    - measurement tasks
    - information tasks
    - decision points (under preparation)

  A specific editor is available for health enquiries that allows the user to design questionnaires with the following type of answers
    - Text
    - Date
    - Yes/No
    - Enumerations
    - Numbers, optionally with a unit
    - Score

  The individual values can be linked with an external service. This mechanism allows, for example, storing the answers in the PHR. Questionnaires are stored in JSON format in the database.

  The tool also contains an editor for information tasks which allows the user to design simple information notifications.

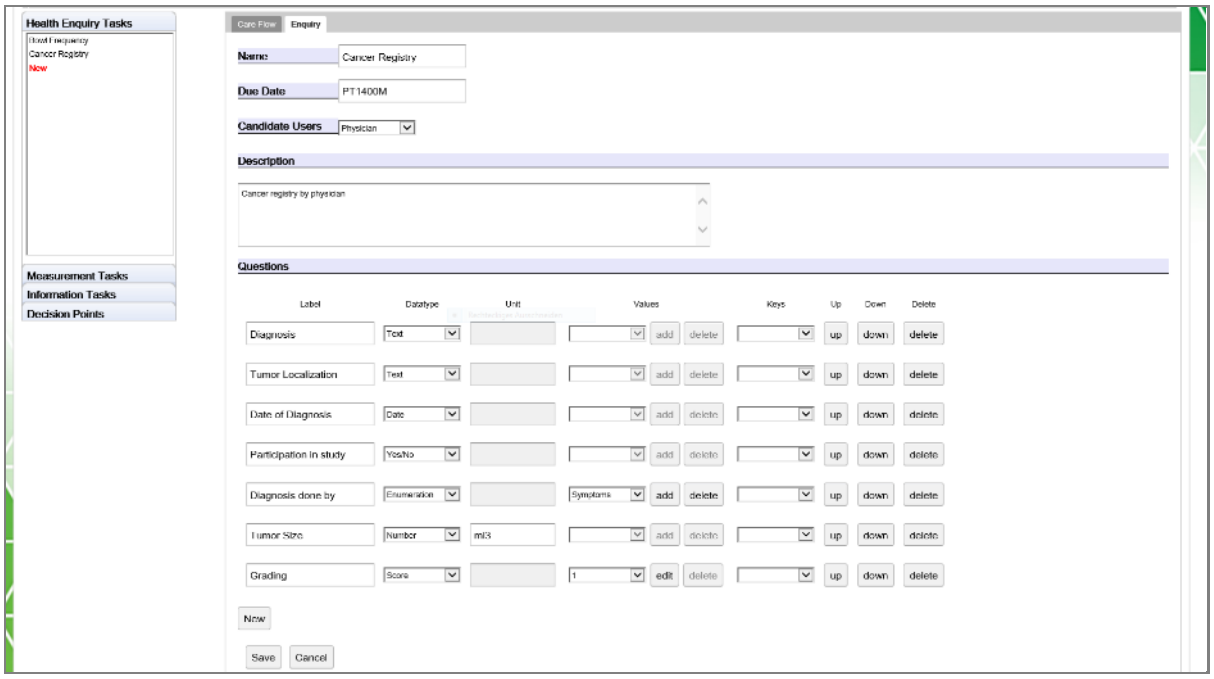

Figure 12: Designing health enquiries. A cancer enquiry is shown in this screenshot that utilises all supported types of question items.

- A canvas to design a Care Flow Diagram by placing graphical symptoms of different tasks on the dashboard and connecting them in the desired order. The designed Care Flow Diagram is stored in JSON format in the database too. It also contains references to the definitions of tasks in the database that are used in the Care Flow Diagram.

  The user can attach one or several conditions to the connection of two subsequent tasks. The expression which combines these conditions must be fulfilled to allow the Care Flow Plan to proceed to the next task. For this purpose, the user clicks on the connection and an editor will open to enter conditions. For each condition, a pre-existing variable of this Care Flow Plan will be compared with another variable or with a value. The logical operators '=', '<', '>', '<=' and '>=' can be used for this. The different conditions can be connected with the Boolean operators 'and', 'or', 'nor' and 'nand'.

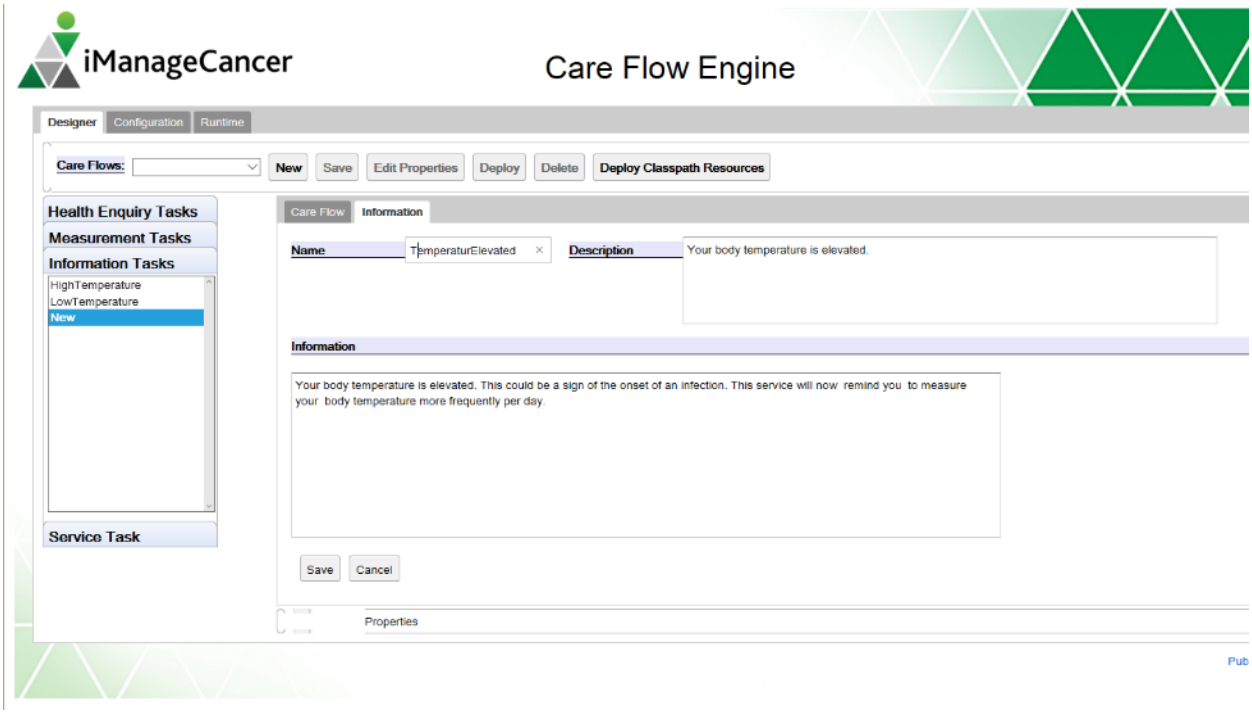

Figure 13: Designing information tasks.

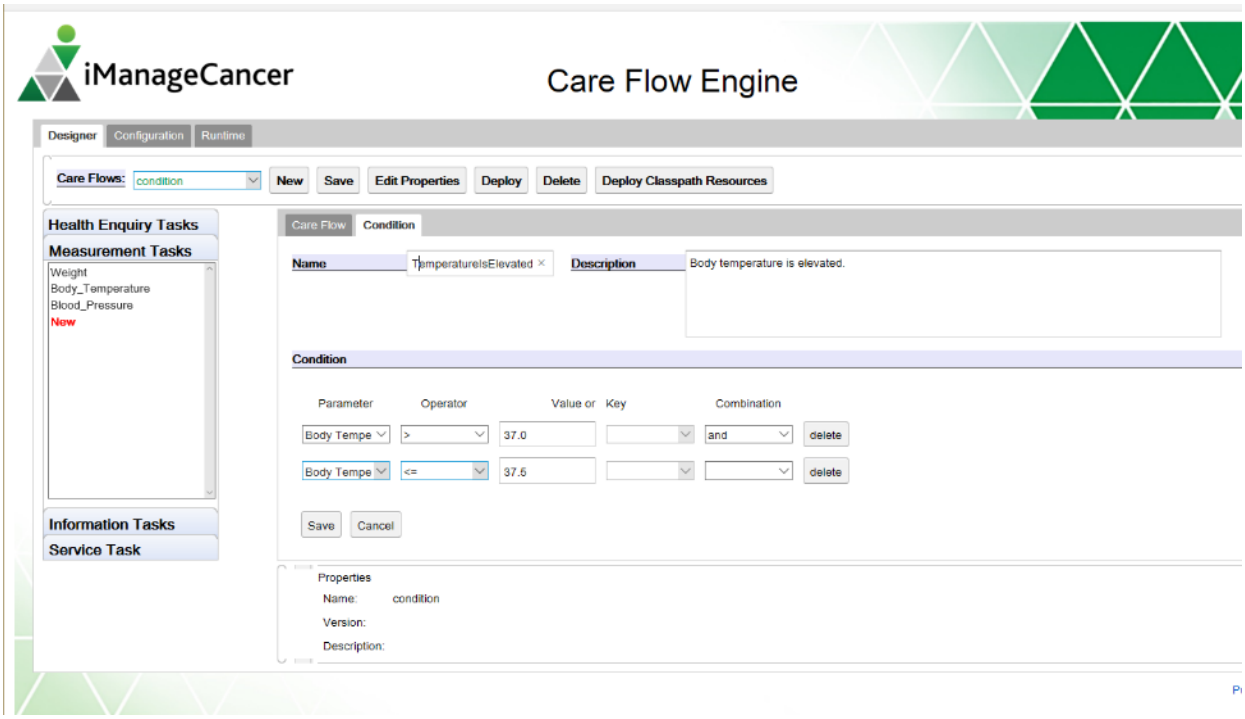

Figure 14: Defining conditions.

Technical implementation

A generator will transform the Care Flow Diagrams that have been drafted in the designer tool into a deployable and executable Activiti BPMN object model. The graphical care flow is encoded in JSON.

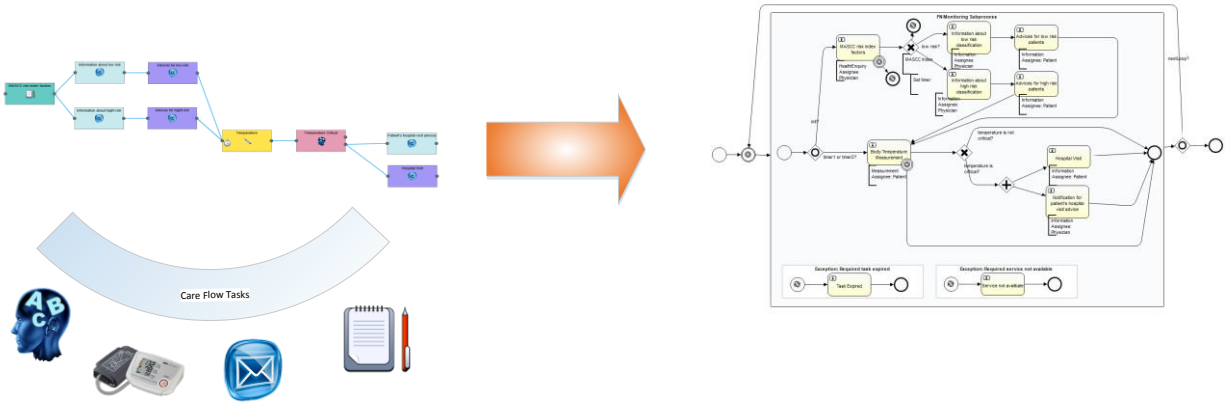

Figure 15: Transformation of designed care flow in Activiti BPMN object model.

In a first step of this process the relevant data of this graphical care flow is extracted from the JSON representation of the care flow. Referenced tasks and connections (with condition) in this care flow are loaded from the database and an Activity BPMN object is created from this task. Then a template is created for an Activity BPMN object model and the task and connection objects with conditions are added to this model. The finalised object model can now be deployed as a new care flow in the Care Flow Engine.

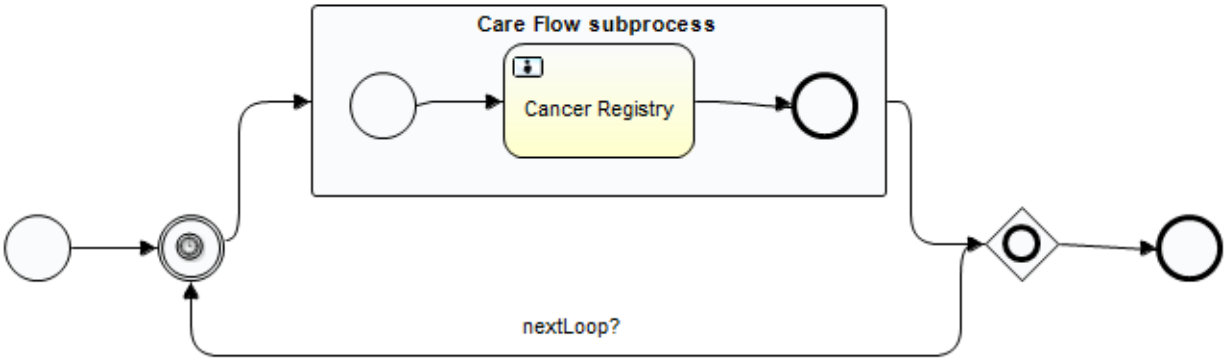

Figure 16: Activiti BPMN Object Model.

This version of the developed Care Flow Designer supports health enquiry, measurement and information tasks and connections between them. The connections can have conditions and we can create now simple decision points in these BPMN models.

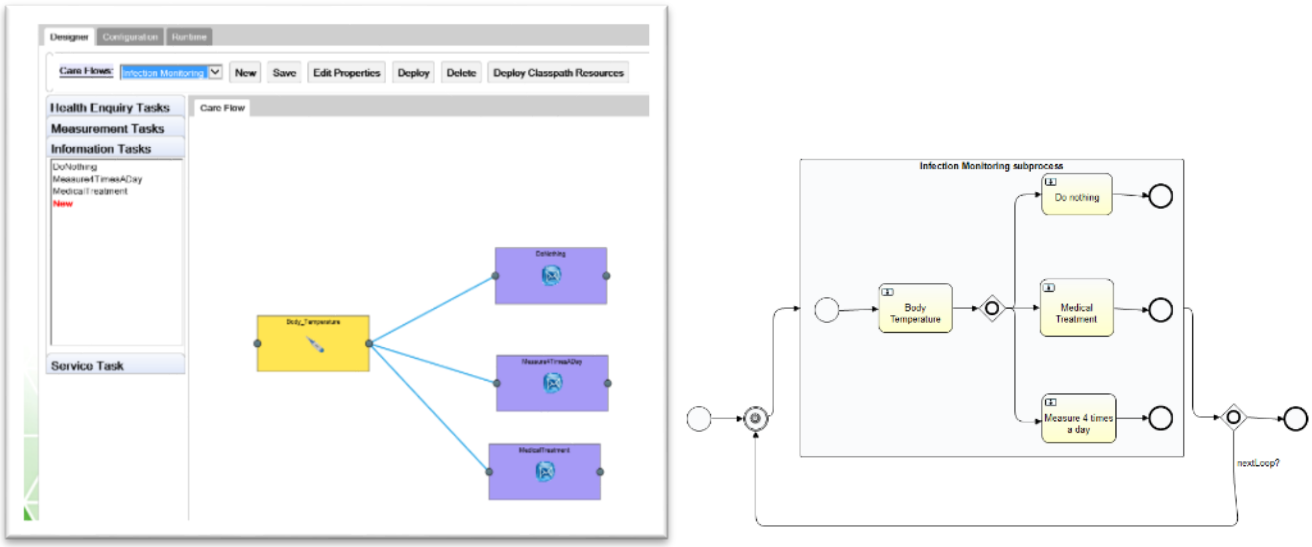

Figure 17: Example of a Care Flow "Infection Monitoring" in Care Flow Designer and resulting BPMN Diagram.

4.5 Creating and deploying Care Flows embedded in the source code

Care Flow Diagrams can also be composed with the Activiti BPMN Designer that needs to be installed as a plug-in in the integrated development environment Eclipse, which we use for the development of Care Flow Engine. The created Care Flows are stored as BPMN files (in XML notation) in the project of the Care Flow Engine (folder src/resources/bpmn). With an update of the Care Flow Engine the new Care Flow will be transferred to the war folder of the web application server. After a restart of the application server the new Care Flow can be added to the BPMN repository of the Care Flow Engine by pushing the button 'Deploy Classpath Resources'. It can now be executed. Alternatively, the BPMN file can be stored in the web space of the web server.

In this way more types of Care Flow Diagrams can be designed and data objects from iPHR can be included in conditions at decision points. The latter is not possible yet in the current version of the Care Flow Designer.

4.6 Provided provisional management services

In this first version of the decision support framework we put the focus on the development of this complex framework with the corresponding clients in the apps and on the integration with the iManageCancer platform. In consequence, we used the management services as described in D5.1 and developed light weighted versions of the proposed Care Flows to drive these developments. In the following, we summarize the corresponding Care Flows.

4.6.1 Infection monitoring

This Care Flow Diagram was designed with the Care Flow Designer for demonstration and development. The main idea is to monitor body temperate at home during chemotherapy cycles. Depending on the measured body temperature a stepwise intervention and monitoring regime is implemented in this workflow. The Care Flow Diagram is shown in Figure 17.

4.6.2 Pain management

A simplified Care Flow for pain management is included in the CDS that asks the patient every day to specify the level of his pain on a scale from 1 to 5 and provides some feedback to the patient depending on the entered level of pain:

- Pain free: “Stay positive and enjoy your life.”
- Mild pain: “Stay active, but prioritize and take rests”
- Moderate pain: “If your pain is new please discuss this with your treating physician. For persistent pain set an action plan or adapt your action plan with your doctor.”
- Severe pain: "Please urgently visit your treating physician”
- Worst pain possible: “Please urgently visit your treating physician or call the ambulance”

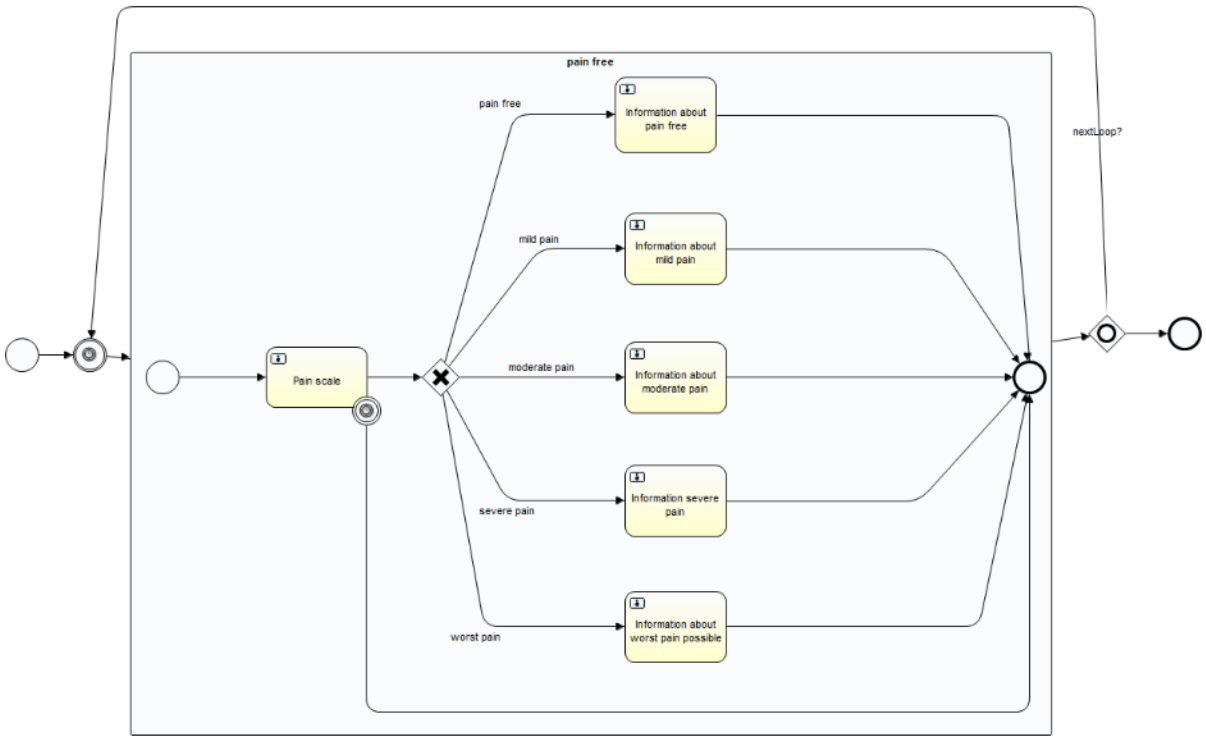

Figure 18: Example of a Care Flow for the management of pain.

In the future this service can be extended to store the level of pain in the iPHR of the patient and show trend curves to patients and doctor, inform treating physicians on pain, or involve more complex rules based on the trend to propose interventions or further literature and resources in the internet.

4.6.3 Fatigue management

The main idea of the Care Flow for fatigue management is to support the patient in budgeting his resources by adapting his plan of daily activities respectively. In consequence, he will be asked in the morning how much energy he has for today, while in the evening the system will ask him how much exhausted he is and whether he feels overburdened or not enough challenged. In the case he feels overburdened and very exhausted the system will give him recommendations how to better budget his resources.

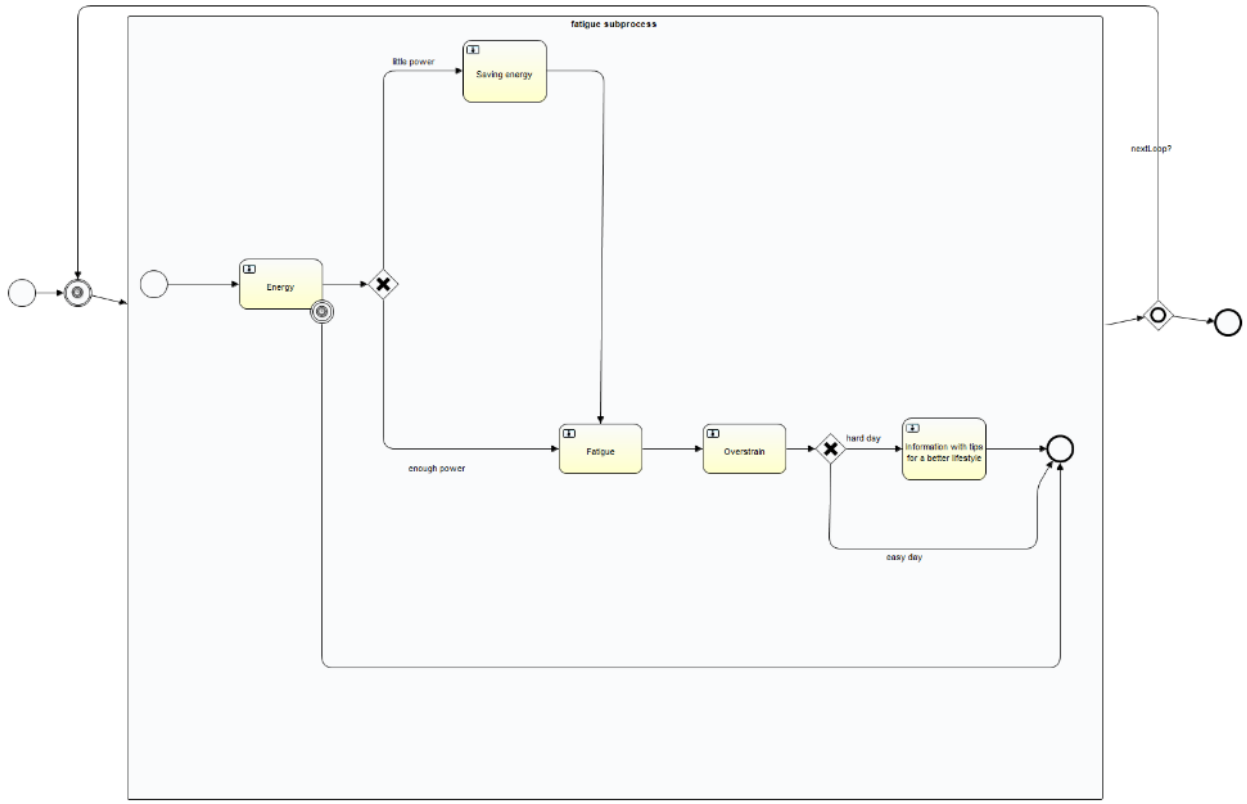

Figure 19: Example of a Care Flow for the management of fatigue.

4.6.4 Febrile Neutropenia risk management

This flow is an example for a specific implementation of a treatment monitoring that involves the Model Repository. It aims to exploit and integrate the "MASCC Risk Index, a scoring system for identifying low-risk cancer patients with febrile neutropenia. The Risk Index could accurately identify patients at low risk for complications. It can be used to select patients for testing more convenient or cost-effective therapies. Details of MASCC Risk-Index Score are presented in deliverable D5.1.
The care flow is as follows:

- Patient or physician signs in to the "FN Monitoring" management flow via the apps iManageMyHealth or iSupportMyPatients.
- Initial flow:
  - Patient's attending physician will be asked for entering patient's MASCC factors with a health enquiry using physicians app iSupportMyPatients:
    - "BURDEN_OF_ILLNESS": [NO,MODERATE, SEVERE]
    - "HYPOTENSION": [NO,YES]
    - "PULMONARY_DISEASE": [NO,YES]
    - "SOLID_TUMOR_OR_NO_FUNGAL_INFECTION": [YES,NO]
    - "OUTPATIENT": [YES,NO]
    - "DEHYDATRATION": [NO,YES]

- "AGE": years (integer)

- The model repository is called with these data to execute the MASCC predictive model and to calculate a risk score. Patient is classified as low risk or high risk patient for developing a febrile neutropenia. This classification result is stored in patient's PHR.
  - Patient receives information material (e.g. for infection prophylaxis) according to the risk classification by the app. Appropriate web documents or links are received by calling the iMC Personal Health Information Recommender.

- Regular measurement tasks are triggered by the CDS (e.g. once a day for low risk and twice for high risk patients):
  - Patient will be notified to measure her/his body temperature via the app iManageCancer. The measurement can be recorded by a Bluetooth Low Energy thermometer or it can be entered manually. The value is be stored in patient's PHR.
  - If the temperature is increased (e.g. >38.5°C or lasting >38.0°C) the patient receives an advice for attending his hospital immediately via the app. Patient's attending physician are also be informed about this advice via the app.

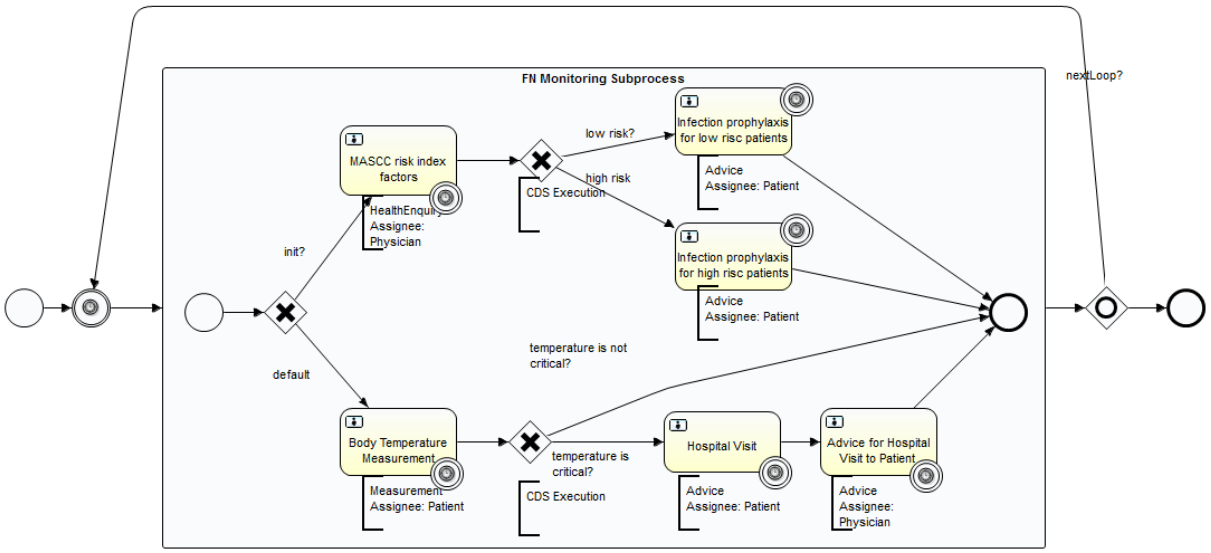

Figure 20: Example of a Care Flow for the management of the risk of Febrile Neutropenia based on the MASCC model.

4.6.5 Care Flows for the management of breast cancer

The model repository was extended with a set of models that can assist in selecting the best therapy to patients with early breast cancer (see 5.5). In consequence a novel Care Flow called StGallen-OncotypeDX was implemented for health professionals that leverage these models respectively. The orginal BPMN2 diagram is shown in Figure 21 . The corresponding implemented Care Flow does not distinguish between different health professional roles as the iManageCancer platform does not support different roles. Some questionnaires that health professionals need to fill to feed the models and receive a suggestion of the system is shown Figure 44.

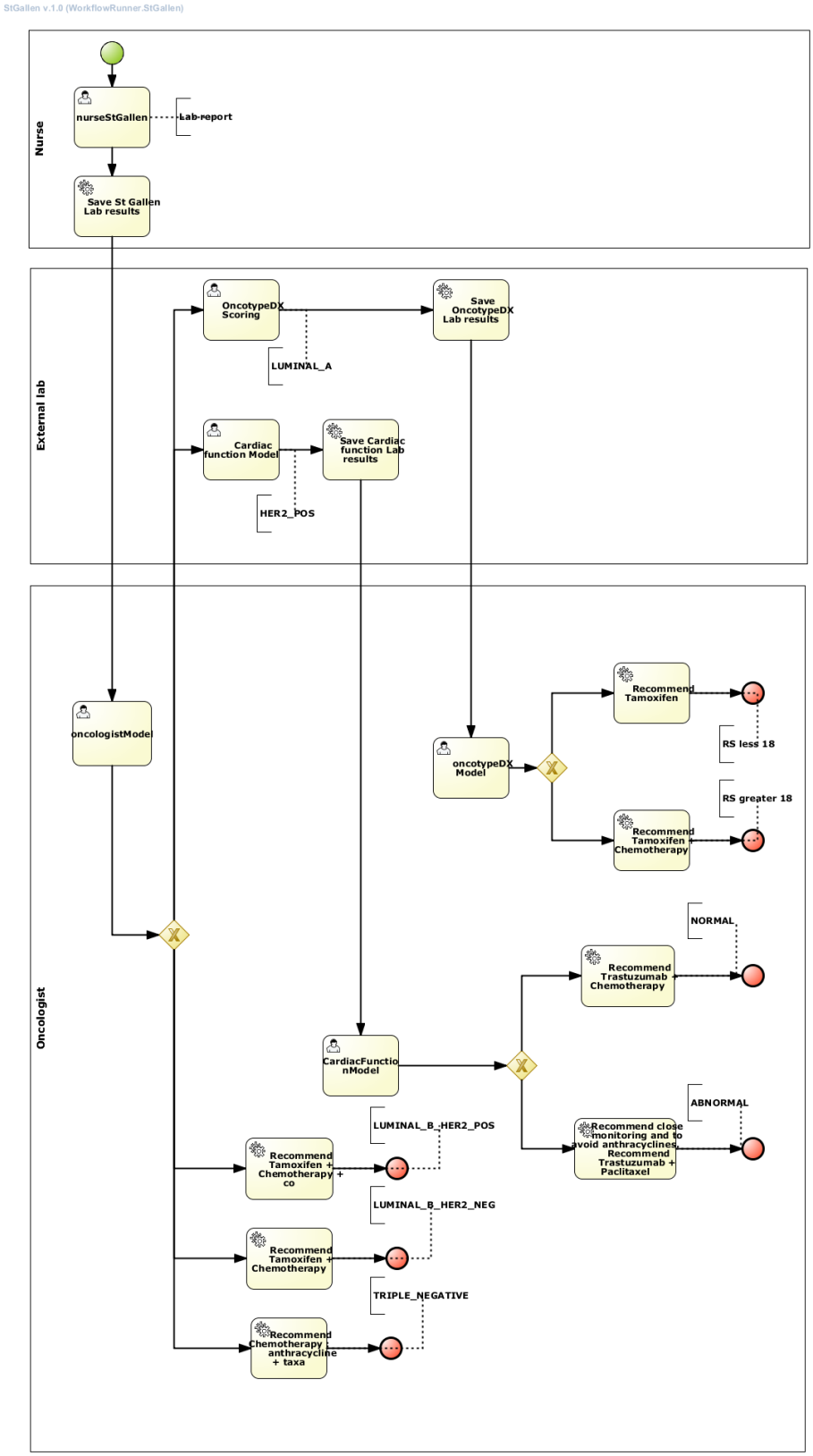

Figure 21: St. Gallen-OncotypeDX workflow to assist in selecting the best therapy for early breast cancer patients.

4.7 REST API

The details of the API of the Care Flow Engine are presented in the appendix of deliverable D5.2.

To this moment, the capabilities of the Care Flow Designer regarding designable Care Flow Diagrams are still very limited. The information editor does only allow to provide text messages to the patient. However, they cannot yet include the content of variables nor links to internet information resources. The Care Flow Designer does not yet integrate with the Personal Health Information Recommender, cannot include parameters from iPHR Data Store or models from the model repository in the Care Flow Diagrams. Future versions shall not only provide these features but also the possibility to include measurement tasks, set timers in Care Flows dynamically, offer more possibilities with decision points and implement exception handling.

5 Model repository

The Model Repository objective is to store predictive models. It is part of the CDS (Clinical Decision Support) along with the Care Flow Engine.

Models added are the validated knowledge ones from the research, and they can also be used by the CDS framework for their continuous validation on new data.

5.1 Internal architecture

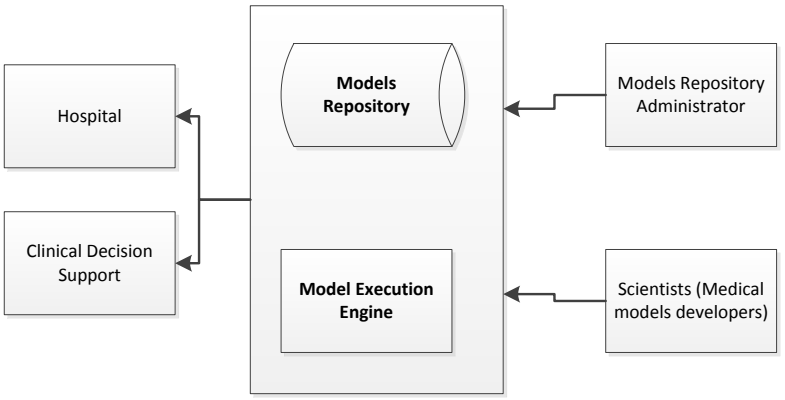

Figure 22: Model repository and model execution engine.

Hospitals and CDS (Clinical Decision Support) users can use the models framework for 2 purposes: query and run models.
Therefore, there are 2 services available to users:

- *Model repository*, service used to manage (create, read, update, delete) models.
- *Model execution*, service used to run a model with patient data.

The different kinds of users are the following:

- *Patients*, they use the Models Repository through the mobiles applications that monitor their health.
- *Hospitals staff*, they query the Models Repository database for clinical models.
- *Scientists and doctors*, they create medical models and upload them to Models Repository.
- *Administrator*, they manages the models and the Execution Engines (e.g. Jess engine).

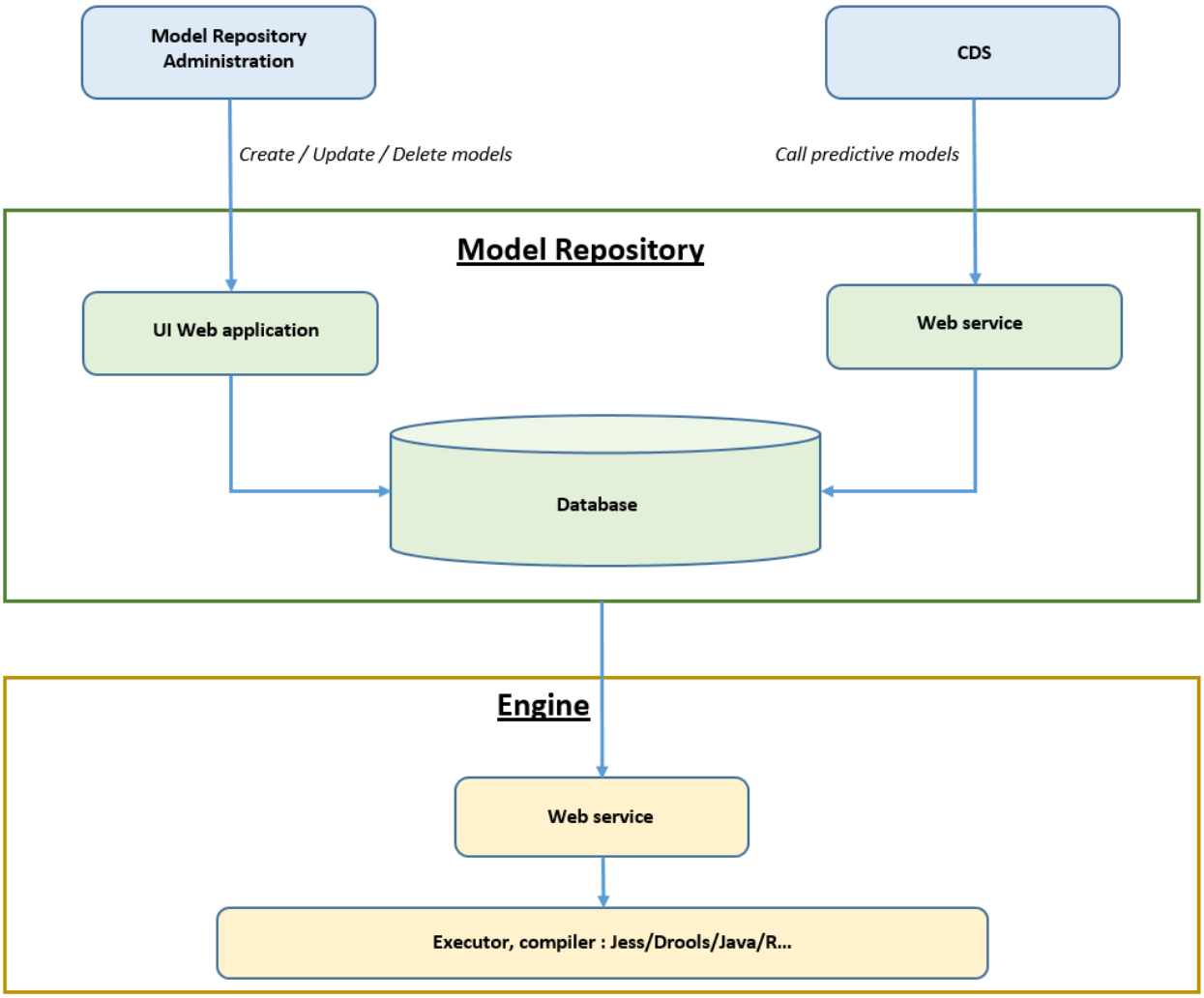

Figure 23: Models repository and engine architecture.

The Model Execution Engine represents the part of the system that executes the clinical model based on the input received from the caller.
There are many engines server depending on the framework used to build the predictive model: Jess, Drools, Java, R; the description regarding the servers (name, url, …) are available in the database.

5.2 Use cases

The Figure 24 describes the use case for the executing of a model.

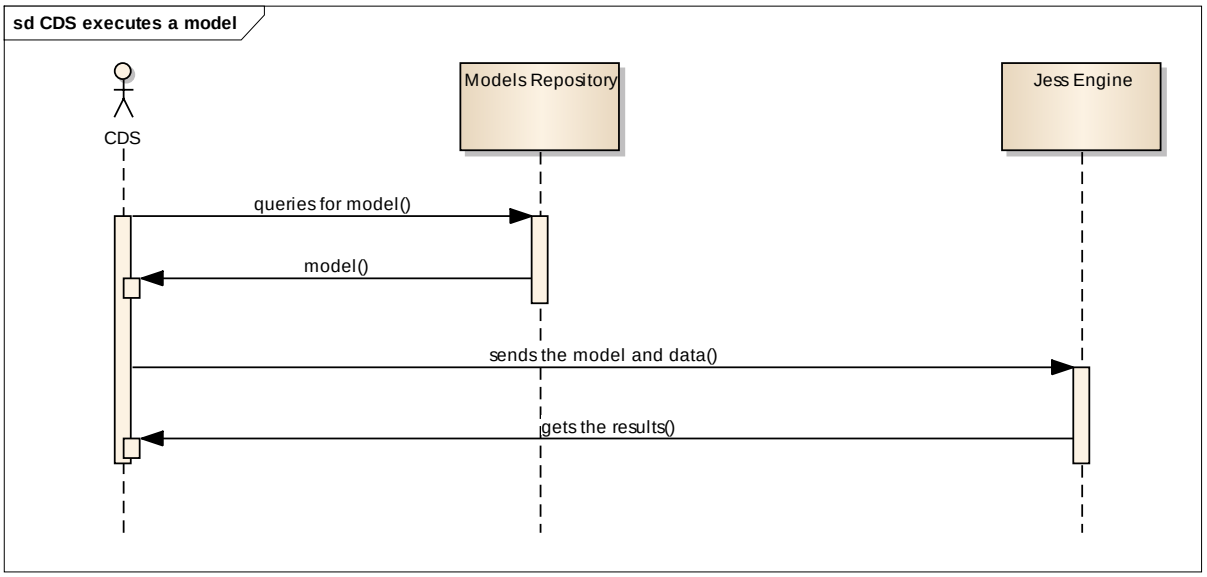

Figure 24: User from CDS executes a model.

The Figure 25 describes the use case for the update and validation of a model.

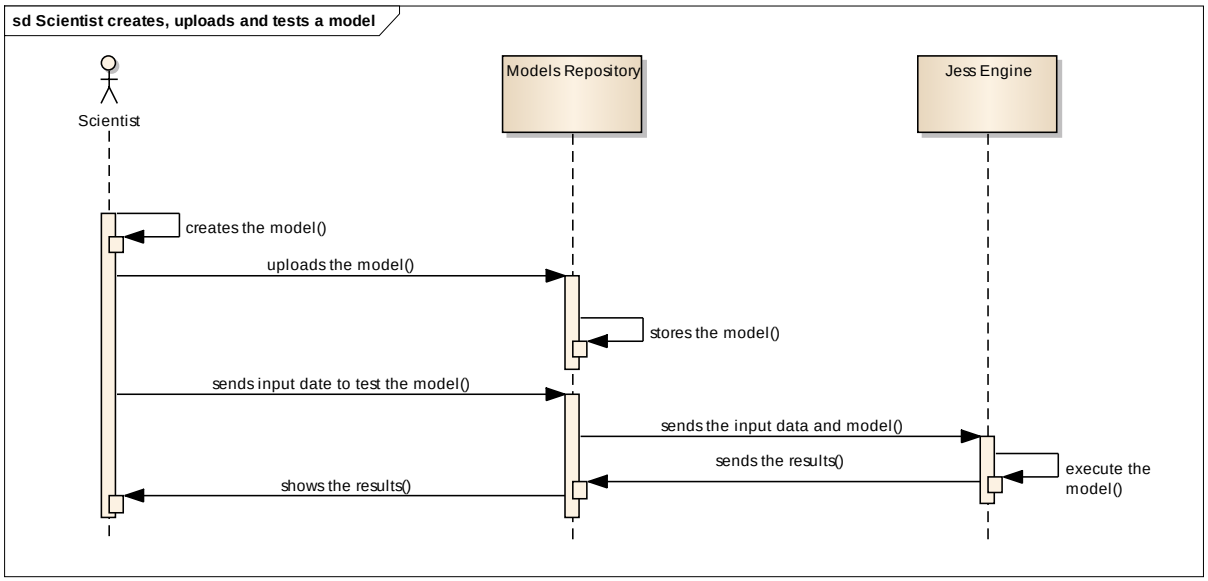

Figure 25: Users update and validate models.

The Figure 26 describes the use case for the management of the model repository.

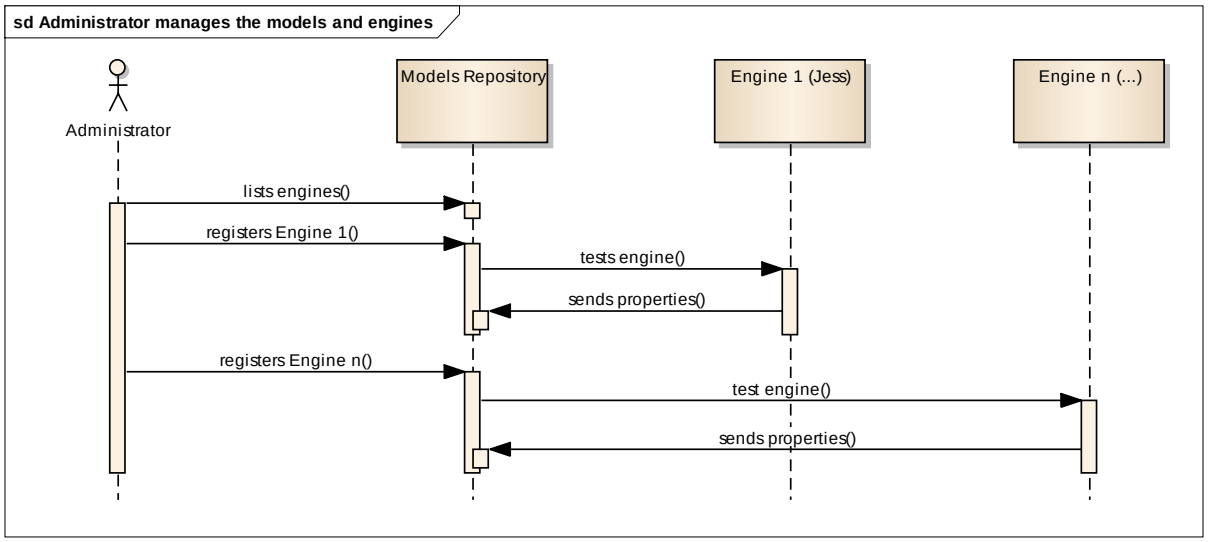

Figure 26: Administrator manages the model repository.

Data design

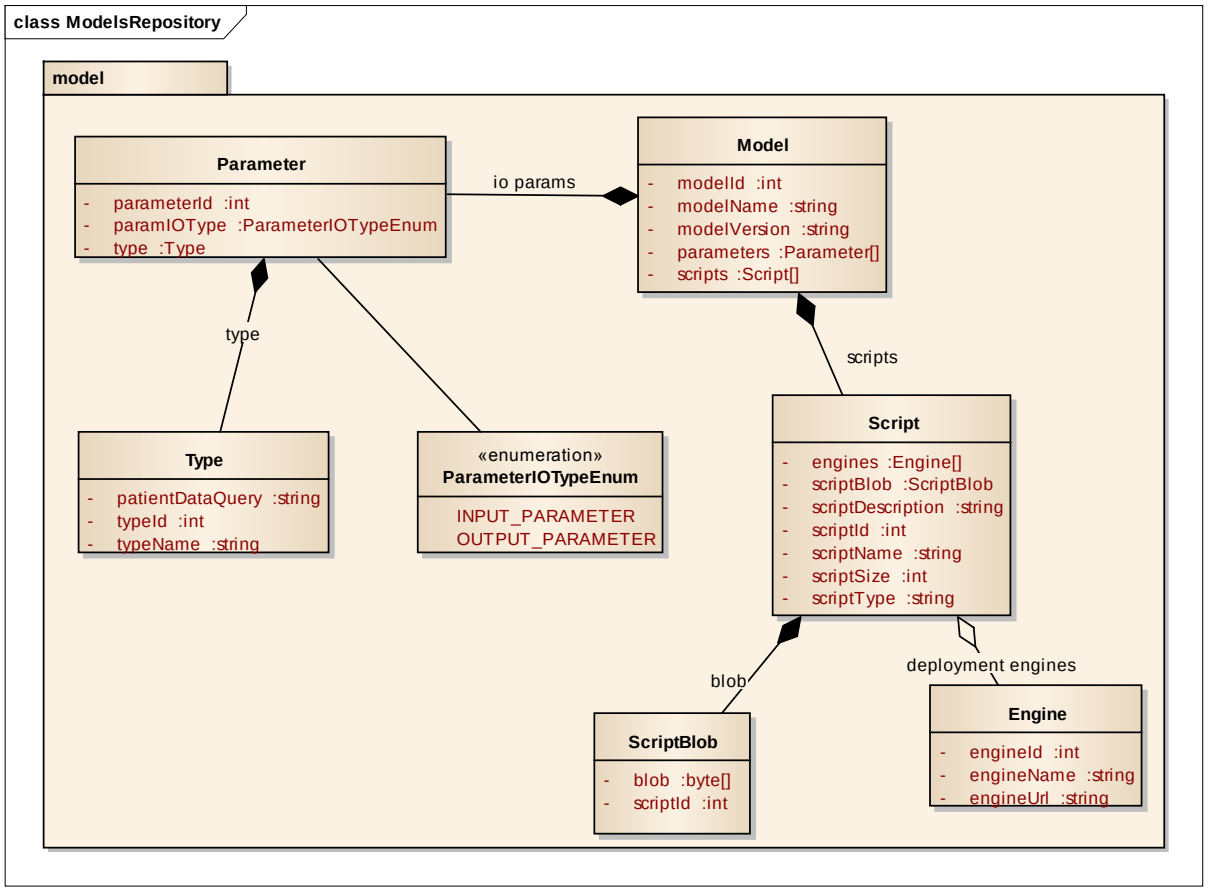

Figure 27: Models repository class diagram.

The central element in the data model is the Model element. It has a list of Parameter (which could be input and output) as well as a list of Script elements. Each script is bound to one or more Engine elements, depending on the execution requirements of the script (Jess, Drools, Java, R…).

5.3 User interface functionality

The Model Repository is accessed through web service as described below.

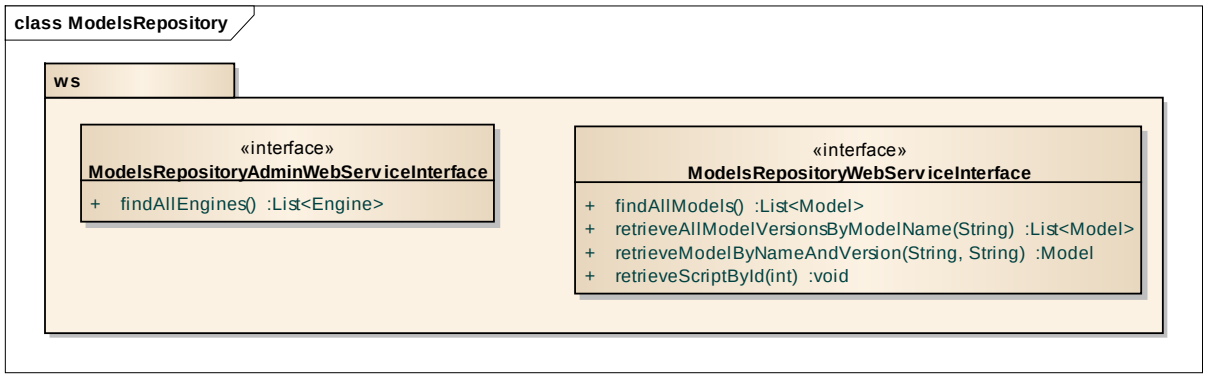

Figure 28: Models repository interface.

ModelsRepositoryAdminWebServiceInterface

- *findAllEngines*, returns a list with all engines register to the Models Repository

ModelsRepositoryAdminWebServiceInterface

- *findAllModels*, returns a list with all models stored by the Models Repository.
- *retrieveAllModelVersionsByModelName*, returns a list with all versions of a model specified by name.
- *retrieveModelByNameAndVersion*, returns a model with a given name and version.
- *retrieveStriptById*, returns the script (blob) of the model.

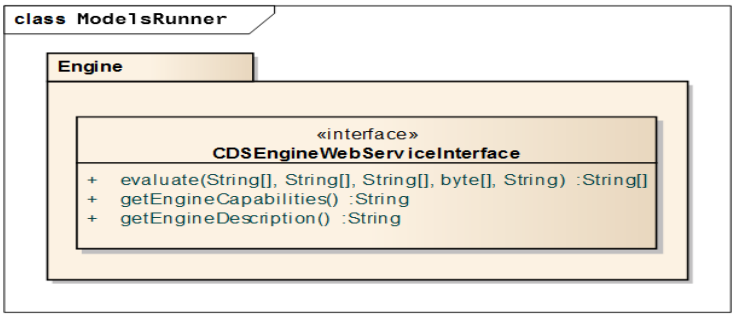

Figure 29: ModelsRunner interface.

ModelsRunnerServiceInterface

- *evaluate*, evaluates a given model and returns the output parameters.
  - inputParameterNames: String[] – the name of the input parameters
  - inputParameterValues:String[] – the values of the input parameters (same order as their names)
  - outputParameterNames: String[] – the name of the expected output parameters
  - script: byte[] – the Jess script that will be sent to Jess
  - type/action:String - type or purpose of the script
  - returns the output parameter values (in the order the names are given as input)
- *getEngineCapabilities*, returns the engine capabilities (e.g. "jess" which means the engine can run Jess code).
- *getEngineDesciption*, returns a short description of the engine.

5.4 Engines

The engines are the servers where are hosted the applications executing the predictive models.

The predictive models can be built in any language: Jess, Java, Python, R...

The Figure 30 shows the sequence of interactions between CDS and the engine (here a Jess engine).

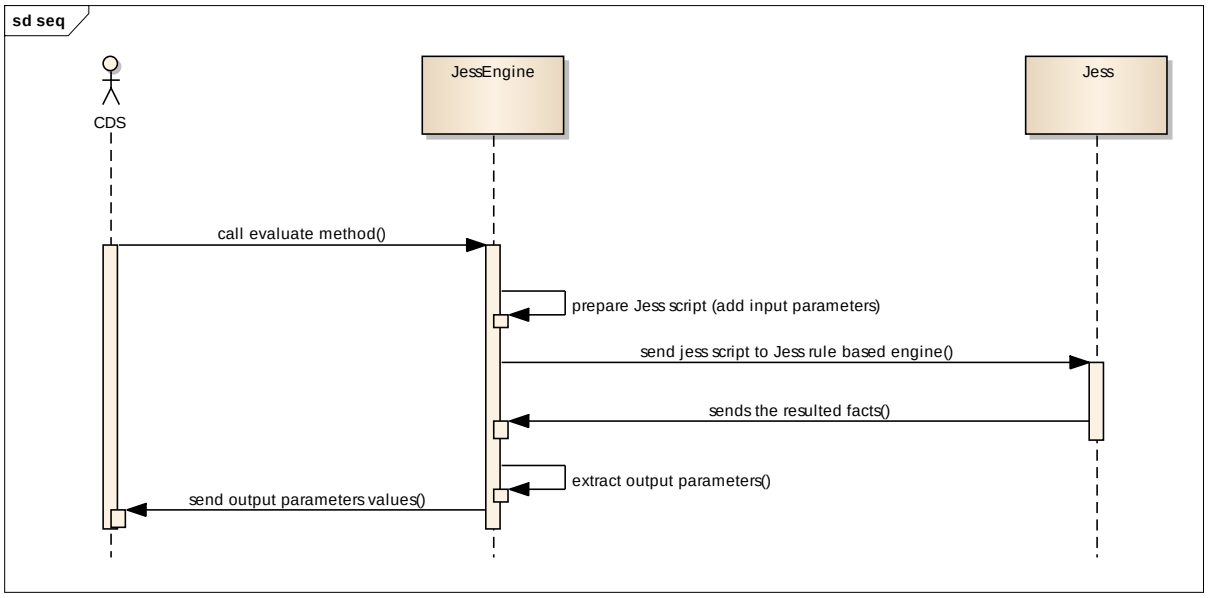

Figure 30: Sequence diagram – interaction between CDS and Jess engine.

5.5 Available predictive models

Some predictive models have been added to the model repository:

- ***St Gallen***, it proposes treatment recommendation for patients suffering from early breast cancer. It uses as input the following parameters: *ER (Estrogen receptor), HER2 (Human epidermal growth factor receptor 2), Ki-67 (protein encoded by the MKI67 gene), PgR (Progesterone receptor)* [17].
- ***Body Mass Index***, calculates the body mass index. Weight is in kg, height is in cm [19].
- ***MASCC*** (Multinational Association of Supportive Care in Cancer), it predicts the risk of a patient for developing a febrile neutropenia. It takes as input the following parameters: *burden of illness, hypotension, pulmonary disease, solid tumor or fungal infection, outpatient, dehydration and age*; and it returns a risk score. The MASCC Risk Index can accurately identify patients with febrile neutropenia that have a low risk for complications. It can be used to select patients for more convenient or cost-effective therapies [19].
- ***Wilms Tumour***, predicts the phenotype that can determine if a patient is characterized, based on his/her miRNA expression data, to a Wilms tumour patient or to a healthy person [19].
- ***BRC* (Breast cancer): Intermittent Bevacizumab Treatment Prediction**, predicts the effect of a user-specified intermittent bevacizumab monotherapy scheme to a specific breast tumour [19].

- ***BRC* (Breast cancer): Bevacizumab Comparison**, compares the treatment outcomes while applying fractionated versions (total amount of drug spread out over total treatment period) of an original bevacizumab monotherapy scheme to a specific breast tumour [19].
- ***Vincristine-Actinomycin***, predicts the effect of a user-specified combination treatment scheme consisted of Actinomycin and Vincristine on a specific Wilm's tumour [19].
- ***OncotypeDX***, it is a test that predicts the recurrence of women breast cancer at early stage. It is based on 21 genes in order to estimate the probability of the benefit of chemotherapy; the scoring is a value between 0 and 100 and we use the following threshold :
    - High (scoring > 18)
    - Low (scoring < 18)
- ***Cardiac function***, it predicts the cardiac function of a patient: *normal or abnormal*.

5.6 REST API

The use of the Model Repository in the CDS (Clinical Decision Support) is through the execution of a model for a specific patient (*For more details, please refer to deliverable iManageCancer_D5.2, section 4.3.4 REST API*).

5.7 New workflow

An extension of the models is to use them within workflows (a workflow can contain many models).

Besides, a workflow will contain different types of tasks:

- ***Laboratory tests***, the results of these tests will be filled in a form by a nurse.
- ***Model validation***, based on laboratory tests the physician can execute a predictive model for the patient.

5.7.1 St Gallen – OncotypeDX workflow

5.7.1.1 Models

The workflow 'St Gallen-OncotypeDX' contains 3 models:

- St Gallen model,
- OncotypeDX model,
- Cardiac function model.

5.7.1.2 Workflow description

There are 3 roles/users (you can choose to merge some roles and have only 2 or 1 depending on your environment):

- *Nurse*, she/he will provide the results of the laboratory results: St Gallen.
- *Laboratory*, it is an external laboratory that will provide some results: OncotypeDX and Cardiac function.
- *Physician/oncologist*, with the laboratory results she/he will execute the models (St Gallen, OncotypeDX, Cardiac function).

The workflow is the following:

1. The nurse provides laboratory results for St Gallen (the results are saved in a database).
2. The oncologist receives a notification making him aware that there are patient results she/he needs to validate, then she/he executes the model with patient data (results of model execution are saved in database).
3. Depending on the results of the St Gallen model execution:
    a. LUMINAL_A, the physician orders OncotypeDX laboratory test and depending on the results
        i. Less than 18, **"Recommend Tamoxifen"**
        ii. Greater than 18, **"Recommend Tamoxifen + Chemotherapy"**
    b. HER2_POS, the physician orders Cardiac function laboratory test and depending on the results
        i. NORMAL, **"Recommend Trastuzumab + Chemotherapy"**
        ii. ABNORMAL, **"Recommend close monitoring and to avoid anthracyclines. Recommend Trastuzumab + Paclitaxel"**
    c. LUMINAL_B_HER2_POS, **"Recommend Tamoxifen + Chemotherapy + co"**
    d. LUMINAL_B_HER2_NEG, **"Recommend Tamoxifen + Chemotherapy"**
    e. TRIPLE_NEGATIVE, **"Recommend Chemotherapy : anthracycline + taxa"**

5.7.1.3 Implementation with BPMN 2.0

The **Fehler! Verweisquelle konnte nicht gefunden werden.** shows the BPMN 2.0 implementation of the *"St Gallen-OncotypeDX"* workflow.

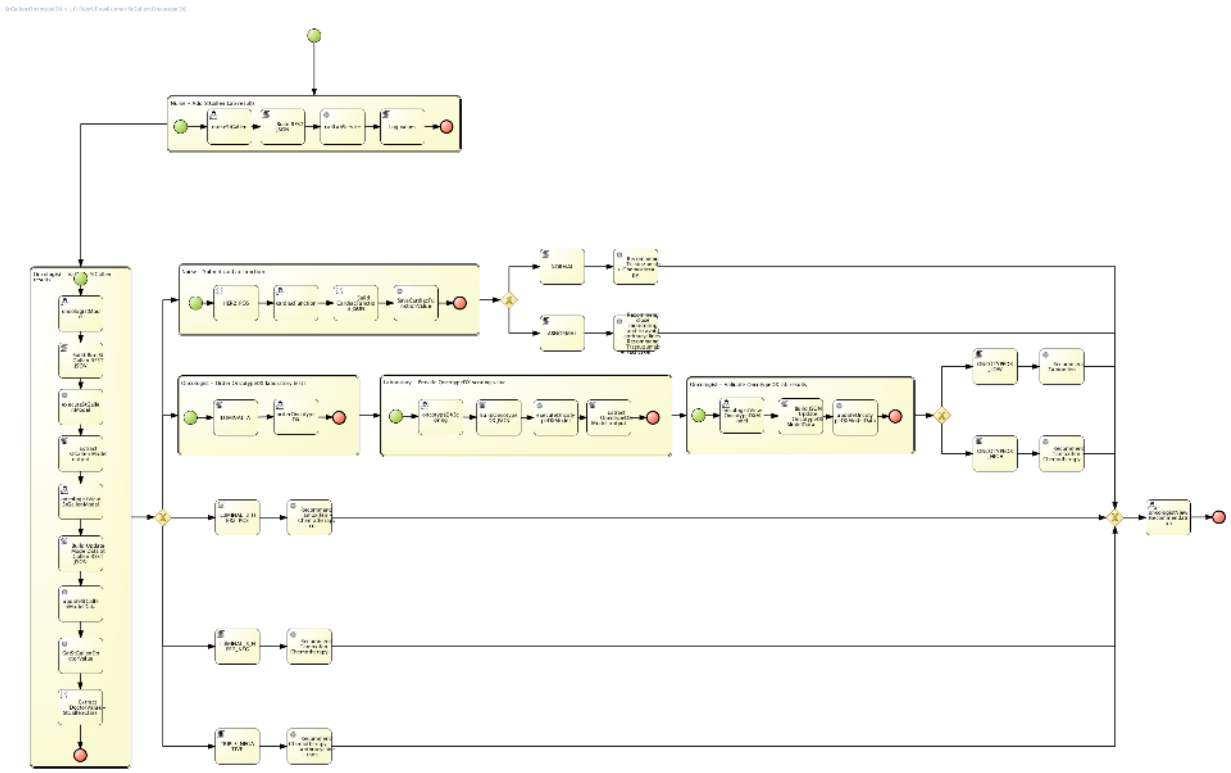

Figure 31: StGallen-OncotypeDX workflow.

6 Client apps iManageMyHealth and iSupportMyPatients

6.1 Internal architecture of apps

The internal structure of the iManageMyHealth app is shown on the picture below. The app contains various components, e.g., for building the user interface, for the management of the app modules (e.g. *My Drugs*, *Health Docs*, *Measurements*), for the communication with external services, for managing the patient's data in the local database, for notification functionalities for drug intakes as well as to interact with the Care Flow Engine. This part has not been changed from deliverable D5.2.

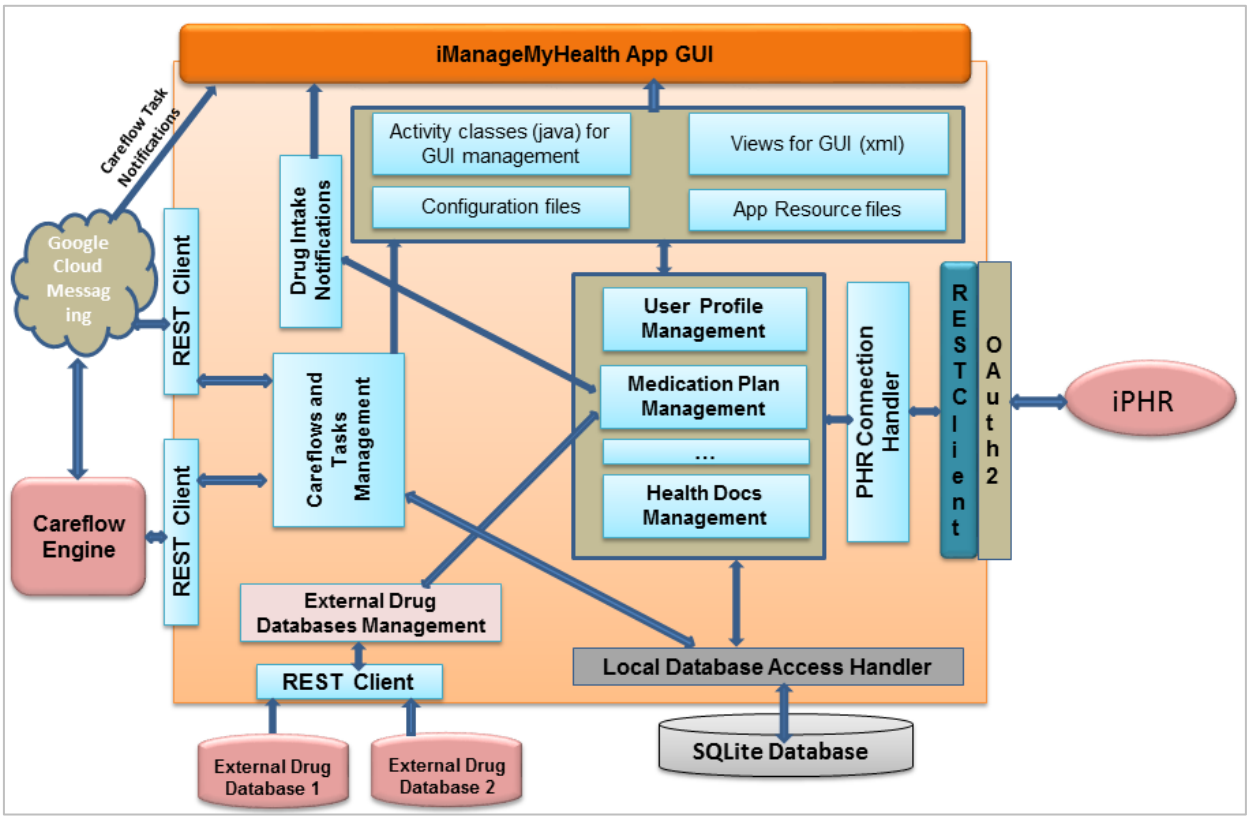

Figure 32 : Internal architecture of app iManageMyHealth.

The app comprises several REST services for the communication with different external services:

- For the communication with the iManageCancer iPHR, e.g. for synchronization of medication plan and health documents as well as retrieving explanation from the Personal Health Information Recommender for selected text in the health documents and sending audit data about user interactions to the iManageCancer iPHR.
- For the communication with the external drug databases for retrieving information about prescribed drugs (incl. drug-drug interactions).
- For the communication with the Care Flow Engine for the management and execution of the Care Flows and their user tasks.
- For the communication with the Google Cloud Messaging services for receiving notifications about novel tasks required by a Care Flow.

6.1.1 Using REST services API of the iManageCancer platform

i. Getting an authentication token from iManageCancer platform

Since the iManageMyHealth app is included in the security framework of the iManageCancer platform, in order to synchronize patient's data (e.g. patient's profile, medication plan, health documents, measurements) with his health record on iManageCancer iPHR as well as to communicate with the Care Flow Engine, the app requests the security framework of the iManageCancer platform for the security token for the patient using the REST method with the URL https://iphr.ics.forth.gr:443/oauth2/access_token (POST). The header of the call should be "Content-Type: x-www-form-urlencoder (or application/x-www-form-urlencoder)".

We use the JSON and XML open standard formats to transmit data objects between the app and the iManageCancer platform.

ii. Synchronizing local patient's account data with his record on iManageCancer iPHR

In order to use advantages of the iManageCancer platform, the patient can enter his iManageCancer account data in the app (user name, password and e-mail entered previously while creating an account on the iManageCancer portal iPHR). As a response for successfully entered user credentials, the patient's personal data stored on the portal is automatically entered in the local database of the app and can be viewed in the *Settings* of the app.

For retrieving the patient's personal data from the iManageCancer platform, the following REST methods are used:

GET *https://iphr.ics.forth.gr/api/accounts/{ACCOUNT_EMAIL}/records* - for getting the patient's record, which is used in the next REST call.

GET *https://iphr.ics.forth.gr/api/records/{RECORD_ID}/demographics* - for getting the patient's demographics.

The user also has a possibility to create a new account on iManageCancer iPHR using a web link to achieve the corresponding iPHR page.

iii. Synchronizing patient's medication plan with iManageCancer iPHR

For synchronizing the medication plan with the iManageCancer iPHR, the following REST methods are used:

POST *https://iphr.ics.forth.gr/api/records/{RECORD_ID}/documents/* - for storing drugs in the medication plan. The details about the drug are transmitted in the body of the REST call in XML format.

POST *https://iphr.ics.forth.gr/api/records/{RECORD_ID}/documents/{document_id}/replace* - for replacing drugs in the medication plan. The details about the drug are transmitted in the body of the REST call in XML format.

GET *https://iphr.ics.forth.gr/api/records/{RECORD_ID}/reports/Medication/* - for retrieving drugs in the medication plan

iv. Synchronizing health documents with the iManageCancer iPHR

For handling health documents on the iManageCancer iPHR, the following REST methods are used:

POST *https://iphr.ics.forth.gr/api/records/Documents/documentsUpload/{record id}* - for storing a health document. The details about the health document will be transmitted as header parameters. The health document will be transmitted in binary format

GET *https://iphr.ics.forth.gr/api/records/{RECORD_ID}/reports/Documents/{document id}* - for retrieving a health document.

v. Uploading measurement values to the iManageCancer iPHR

For storing measurement values of the patient's vital signs (e.g., weight, body temperature, blood pressure), the following REST method is used:

POST *https://iphr.ics.forth.gr/api/records/{RECORD_ID}/documents/*

The measurement values are transmitted in the body of the REST call in XML format.

vi. Retrieving information from Personal Health Information Recommender (PHIR)

For retrieving explanation about selected text in the patient's health documents, the following REST method is used:

GET https://iphr.ics.forth.gr/PHIR/webresources/search/primary/{patient's e-mail} /{language}/{search text}

vii. Sending audit information to the iManageCancer platform

For sending audit information about patient's interactions in the app to the iManageCancer platform for later analysis, the following REST method is used:

POST https://iphr.ics.forth.gr/api/records/{RECORD_ID}/documents/

The auditing information will be transmitted in the body of the REST call in XML format.

viii. Sending assessment and rating information about the app modules to the iManageCancer platform

The patient has a possibility to rate different modules of the app and to answer question about his experience with the app as well to make suggestions for improving the app. For this functionality the following REST method is used:

POST http://139.91.210.63:8084/DataManagementAPI/s1/webservice/patient/{patient' id}/{date}/{questionnaire number}/{app name}/{app module}/{rating}

The questionnaire answers will be transmitted in the body of the REST call in JSON format.

These methods above are described in more details in deliverable D3.2.

6.1.2 Using of REST services API of Care Flow Engine

For handling health enquiries as well as measurement and information tasks, the Care Flow Engine provides REST services. The API is described in detail in the appendix of deliverable D3.2***Fehler! Verweisquelle konnte nicht gefunden werden.***. We use the JSON open standard format to transmit data objects between the apps and the Care Flow Engine.

i. Using an authentication token from iManageCancer iPHR

Since the Care Flow Engine is included in the security framework of the iManageCancer platform, in order to use the REST services of the Care Flow Engine, the app requests the security framework of the iManageCancer platform for the security token for an app user (patient or physician) which is used for calling the REST services of the Care Flow Engine in the header of the REST call: "Headers: Authorization: bearer {IMC_TOKEN}" (i).

ii. Getting an authentication token from iManageCancer iPHR

Since the Care Flow Engine and the apps are included in the security framework of the iManageCancer platform, in order to use the REST services of the Care Flow Engine, the app requests the security framework of the iManageCancer platform for the security token for the user of an app (patient or physician). The token is used for calling the REST services of the Care Flow Engine in the header of the REST call: "Headers: Authorization: bearer {IMC_TOKEN}".

The token from the iManageCancer platform can be retrieved using the REST method with the URL https://iphr.ics.forth.gr:8080/oauth2/access_token (POST). The header of the call should be "*Content-Type: x-www-form-urlencoder (*or *application/x-www-form-urlencoder*)".

iii. Subscribing/unsubscribing for care flows

In order to subscribe and unsubscribe to Care Flows the following REST methods of the Care Flow Engine API are used:

POST careflows/{careFlowKey}/start) – a new Care Flow for the user is started on the Care Flow Engine.

PUT processes/{processId}/stop – a running Care Flow of a patient is stopped and all pending tasks related to this process are deleted on the Care Flow Engine

A particular Care Flow can be requested from the Care Flow Engine in the case that a user of the app has received a notification for a specific task of this Care Flow and the corresponding Care Flow is not yet stored in the user's local data base. In this case the app is using the following REST calls:

- for getting a task for which a notification is received: *GET tasks/{taskId}*
- for getting the related care flow process: GET processes/{processId}
- for getting the related care flow*: GET careflows/{careFlowKey}*

iv. Getting Care Flows, processes and tasks from the Care Flow Engine

For retrieving Care Flow for a user (patient or physician) from the Care Flow Engine the REST call *GET careflows/query* is used. When the Care Flow is started for a particular patient the processes and the pending tasks for this patient related to the Care Flow are also retrieved from the Care Flow Engine using the following REST calls:

- for the patients' Care Flow processes: *GET processes/query*
- for the users (patients and physicians) pending tasks: *GET tasks/query*

Specific REST methods are provided to get a Care Flow of a particular kind:

- for getting only health enquiry tasks: *GET tasks/enquiries/query*
- for getting only information tasks: *GET tasks/information/query*
- for getting only measurement tasks: *GET tasks/measurements/query*

v. Executing tasks

The tasks received from the Care Flow Engine shall be presented to the user, who has to provide specific health related information in most of the tasks. Such results shall be sent back to the Care Flow Engine in order to complete the task and allow the Care Flow to proceed to the next task:

- for health enquiries: the questions need to be answered
- the measurements shall be performed
- reading of any text message (information task) should be confirmed

For completing a task, the app is using a REST call *PUT tasks/{task-id}/complete*. The body of the REST call contains the results (values) of the task execution with related question identifiers.

6.1.3 Notification services

For supporting notifications using Google Cloud Messaging services we have created corresponding projects and server keys for the apps iManageMyHealth and iSupportMyPatients on the Google platform using the link https://console.developers.google.com:

The project numbers need to be entered to the '*res/values/strings.xml*' files of the both apps, e.g.

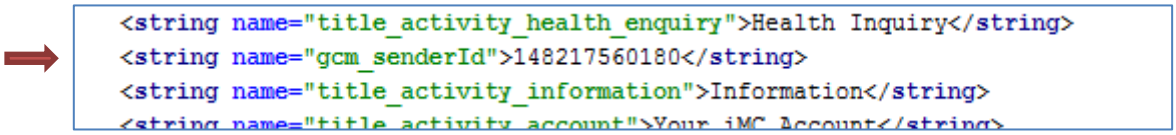

Figure 33: res/values/strings.xml' file of the app with project number.

The corresponding server keys need to be entered in the configuration of the Care Flow Engine.

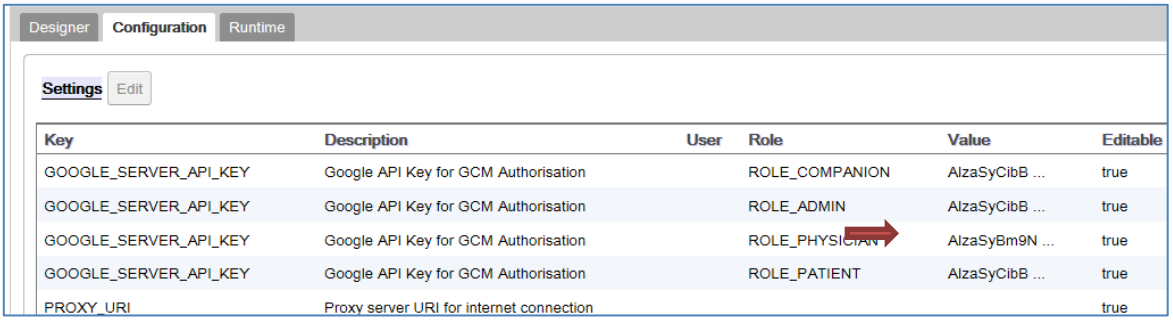

Figure 34: Configuration in the Care Flow Engine where Google API keys are entered by apps.

For creating the server keys for the iManageMyHealth and for the iSupportMyPatients apps we have selected "Activate APIs and manage" and then "Access data" in the Google project view. We have selected the option "API Key" in the drop down menu "Select credentials" and then selected the option "Server key" in the shown dialog "Create a new key". In the next view, we have entered a key name (any text) and pressed the "Create" button.

For activating the keys we have to select the "Overview" option in the navigation menu and select the "Google Cloud Messaging" option in the "Mobile Apps" section.

The diagram below shows how the notification service is working for the Care Flow Engine and for the apps.

The communication is implemented as REST clients for the Google Cloud Messaging (GCM) services. When an app is configured correctly, the app sends a sender_id value to the GCM and gets a token (a text sequence) back that needs to be sent to the Care Flow Engine using the REST call of the Care Flow Engine API: *POST notifications/register/gcm*. The Care Flow Engine stores the tokens for the apps together with the user (patients and physicians) iManageCancer identifiers (e-mail addresses) and can send notifications to the mobile devices of the users.

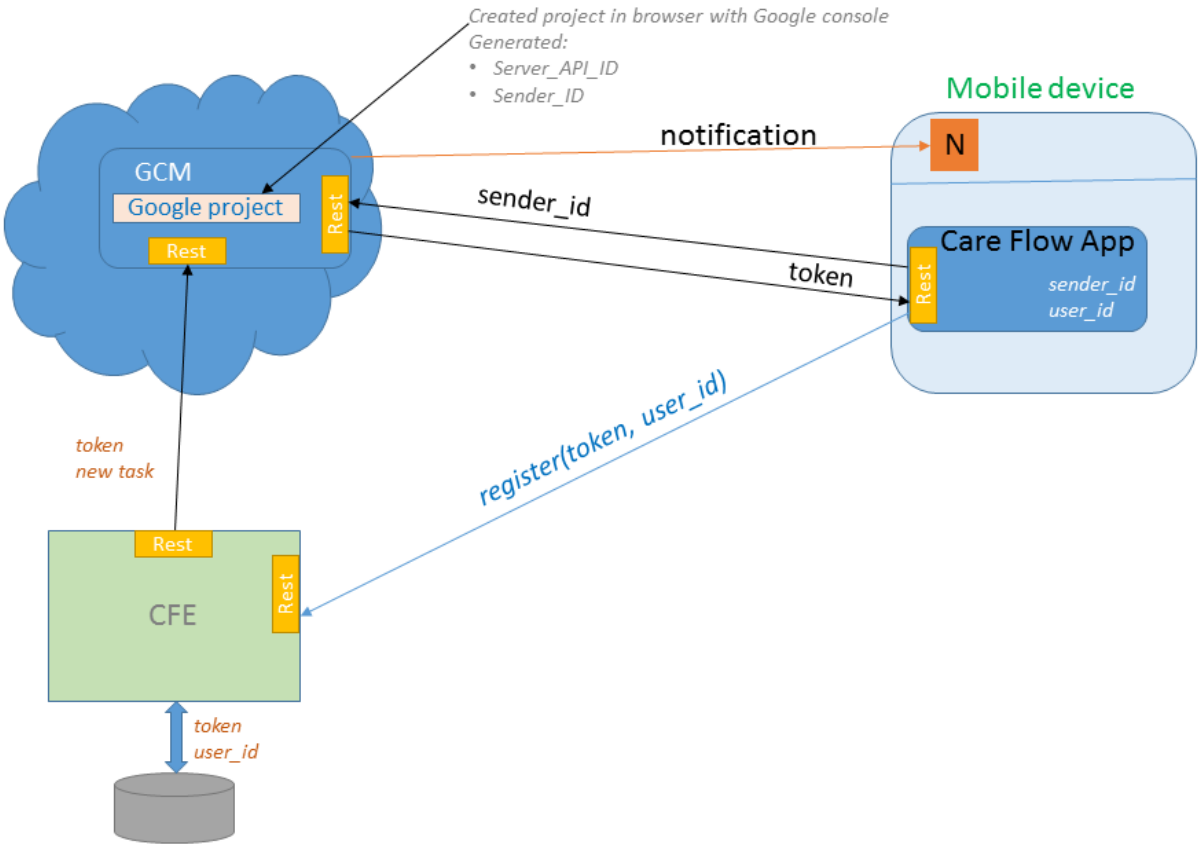

Figure 35: Use of Google Cloud Messaging in Care Flow Engine and corresponding apps.

6.1.4 SQLite database

The database tables are described in deliverable D5.2.

6.2 User interface

Patients and their physicians registered on the iManageCancer platform can utilize the management services provided by the Care Flow Engine by using the iManageMyHealth and iSupportMyPatients mobile applications (apps). Below the start pages of the both apps are shown.

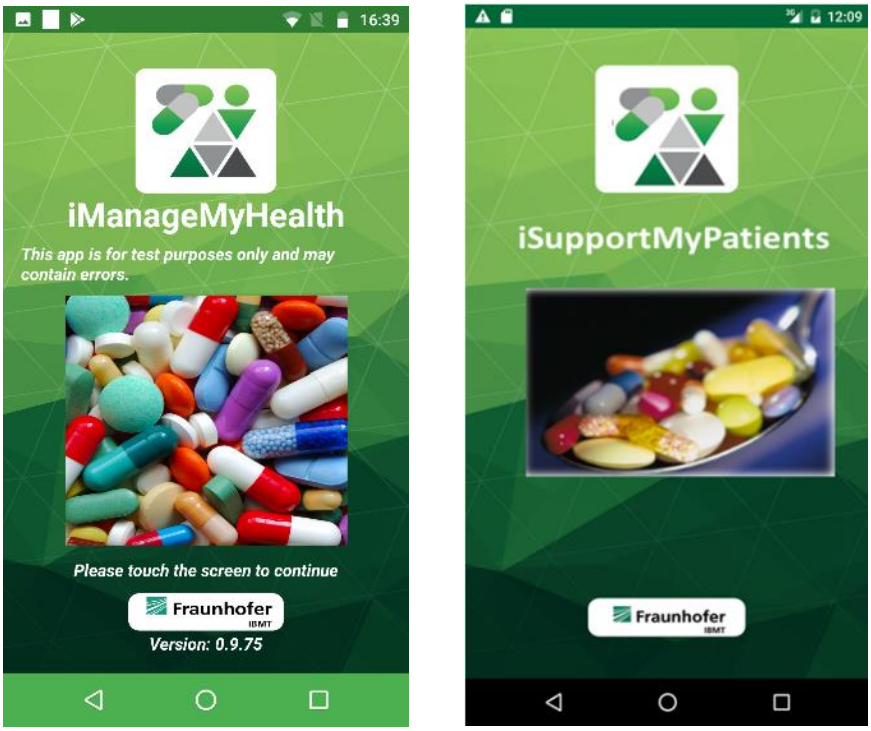

Figure 36: Start pages of the two iManageCancer apps that make use of the Care Flow Engine.

For using the services provided by the Care Flow Engine users (patients and their physicians) need to be registered on the iManageCancer platform. After creating accounts, they can enter their account data in the app which is stored locally in the SQLite database available on the app.

Patients using the iManageMyHealth app are forwarded to the main menu page in the app where they can achieve the Health Manager module in order to view and to manage the care flows and tasks received from the Care Flow Engine. There are different roles (e.g. ROLE_PATIENT and ROLE_PHYSICIAN) to define in the design phase of a Care Flow Diagram who is allowed to start a particular care flow. It is also possible to allow starting a care flow for multiple roles. Only the care flows assigned to the role ROLE_PATIENT are shown in the iManageMyHealth app to allow a patient to subscribe or unsubscribe for the care flow.

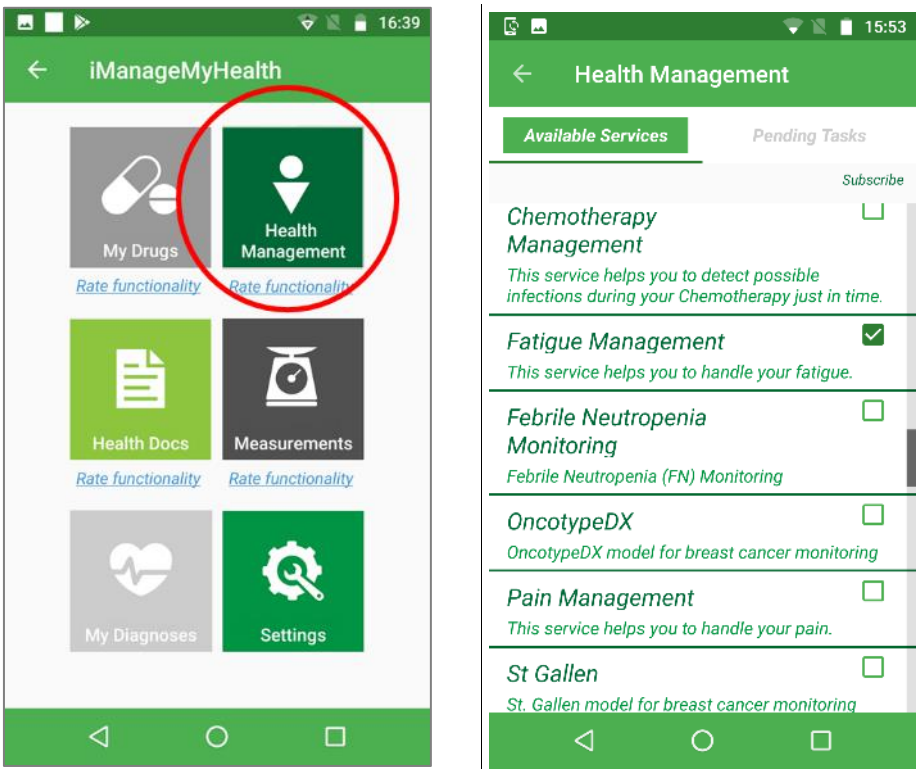

Figure 37: Care Flow Diagrams in the iManageMyHealth app. The patient can subscribe to these services or can unsubscribe from a Care Flow.

Achieving the Care Flows and tasks on the iSupportMyPatients app for physicians is a little bit different: the app receives the care flows and tasks from the Care Flow Engine for all 'own' patients registered on the iManageCancer platform. The 'own' patients means the patients who have decided to share their data with the particular physician. The physician can see a list of the 'own' patients and number of currently pending tasks for the patient, which should be executed by the physician. After selecting a patient or his pending tasks, the physician will be forwarded to the similar page (like for patients) for management of the care flows and tasks. Only the care flows assigned to the role ROLE_PHYSICIAN will be shown to a physician on the iSupportMyPatients app.

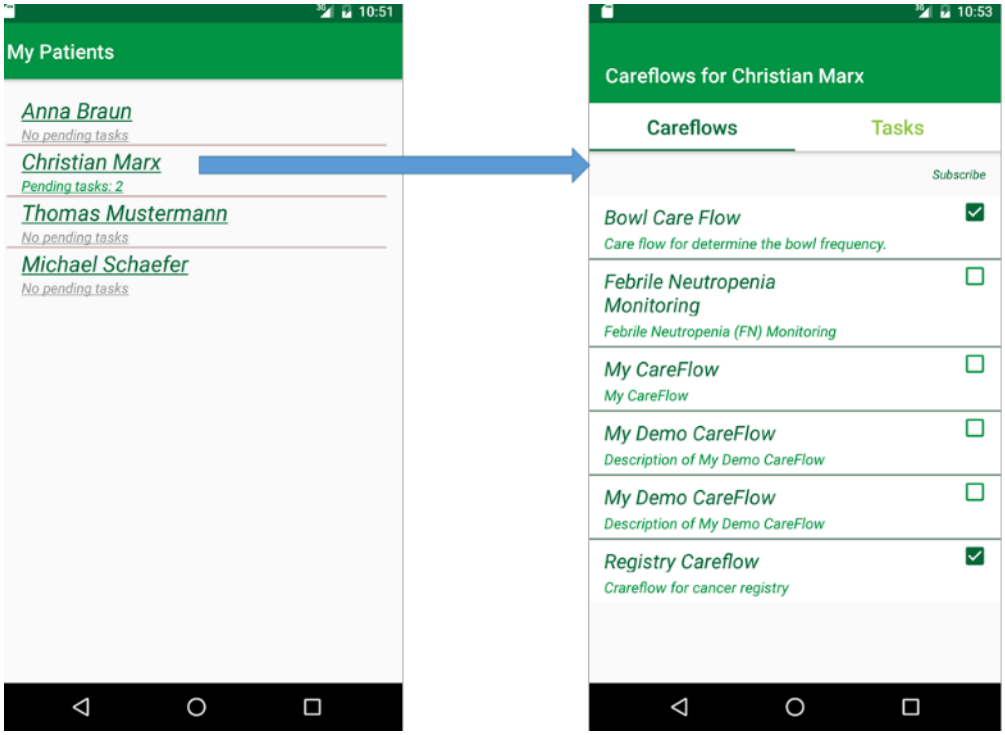

Figure 38: Care Flow Diagrams in the iSupportMyPatients app. The physician can subscribe to these services for a specific patient or he can unsubscribe from a care flow for this patient.

The page for managing care flows and pending tasks is similar for both apps and contains two fragments (tabs), one for the care flows and the other for the pending tasks. The user (patient or physician) can switch between them.
When the physician selects the name of the patient in the list of patients he will be forwarded to the list of available care flows the care flows shown on the picture above. The user (patient or physician) can see on this view a name of the care flow and its description. The user can subscribe for available care flows or unsubscribe from running care flows using a check box next to the care flow name on the view.

When the physician selects the pending tasks in the list of patients he will be forwarded to the second fragment for the pending tasks shown below.

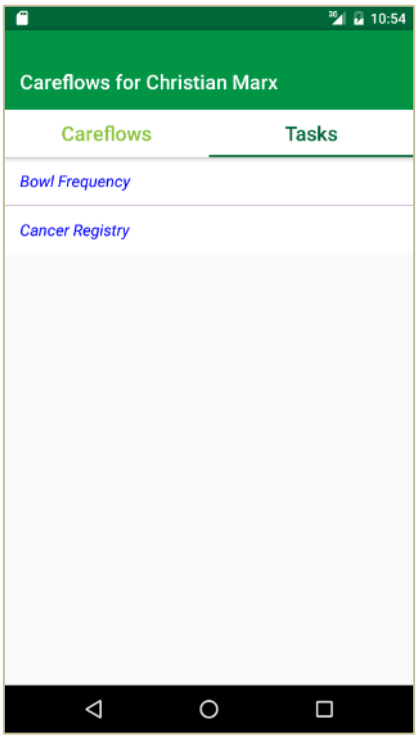

Figure 39: Pending tasks for the physician for care flows of his patient 'Christian Marx' in the iSupportMyPatients app.

The user (patient or physician) can select a pending task for displaying and executing the task. The task will be retrieved from the Care Flow Engine and a view with the content of the task will be generated dynamically for displaying it to the user.

Using the iManageMyHealth app, patients get typically the following tasks from a care flow:

- health enquiry tasks: health assessment questionnaires, psycho-emotional and cognitive tests,
- measurement tasks: a prompt to perform a measurement (e.g. blood pressure, weight, body temperature) or enter laboratory values,
- information tasks: information about helpful sources relevant to patient's health condition.

Using the iSupportMyPatients app, physicians get the following tasks from a care flow:

- health enquiry tasks: health assessment questionnaires about their patients
- information tasks: information about patient's health condition

Below is an example for a received health enquiry that shows how different types of generic questions are supported and rendered: text, date, number, enumeration (for selecting one or more value(s) from multiple answer possibilities), scores, and yes/no values.

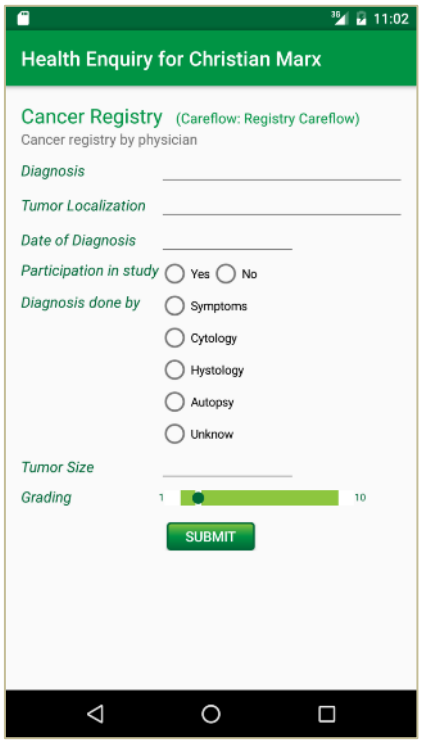

Figure 40: Enquiry task for a user that demonstrates how the different types of questions are presented in the user interface.

We provide different comfortable possibilities for entering values using soft keyboards for entering text or only number as well as a view for selecting a date.

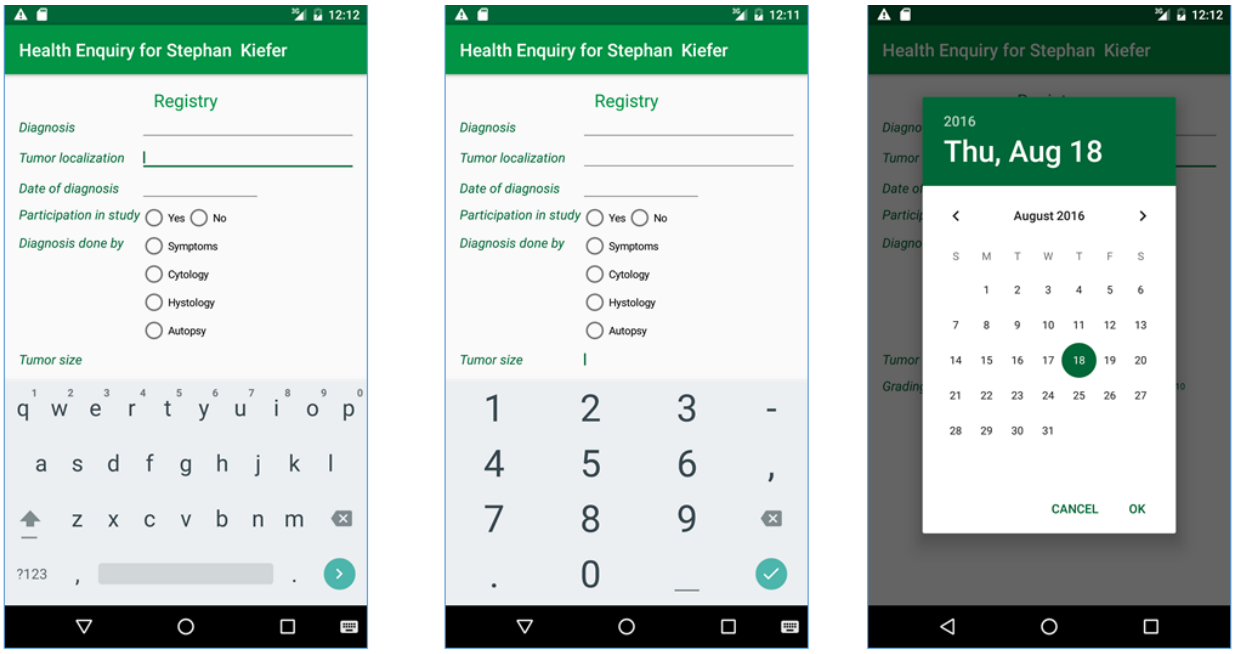

Figure 41: Enquiry task for user and means how to enter the answer to different types of questions.

The information tasks display to the users information which should be confirmed in order to complete a task on the Care Flow Engine. The displayed information for patients can include web links to important sources related to their condition recommended by the Personal Health Information Recommender tool included in the iManageCancer portal.

Measurement tasks retrieved from the Care Flow Engine are analysed on the app and the patients are forwarded to a corresponding view for entering measurement data, e.g. to the views for

entering blood pressure, weight or body temperature values. If medical devices are connected to the app the patient is asked to carry out a measurement. The results are received from the device, stored in the local database and sent back to the Care Flow Engine.

Since the Care Flow Engine and the apps are configured for using notification services, the Care Flow Engine sends notifications about new tasks to the apps and the users of the apps can initiate a process for retrieving the tasks from the Care Flow Engine after receiving a message about the new tasks.

The screenshot below shows a notification about a new health enquiry task. After clicking on the notification, the task will be retrieved from the Care Flow Engine and user will be forwarded the corresponding view generated for handling the retrieved task.

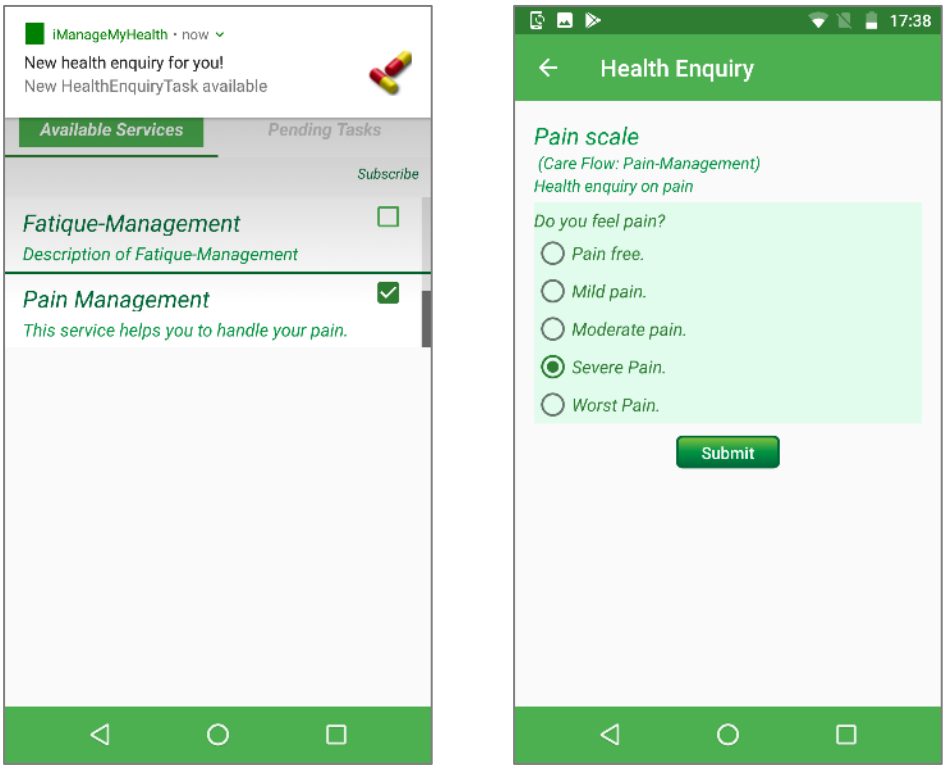

Figure 42: A message about a pending task is received on the smartphone. If the user touches the message he is forwarded to the corresponding page in his app.

The following screenshots illustrates the behaviour of the app iManageMyHealth when the user subscribes to the health service *Fatigue Management*.

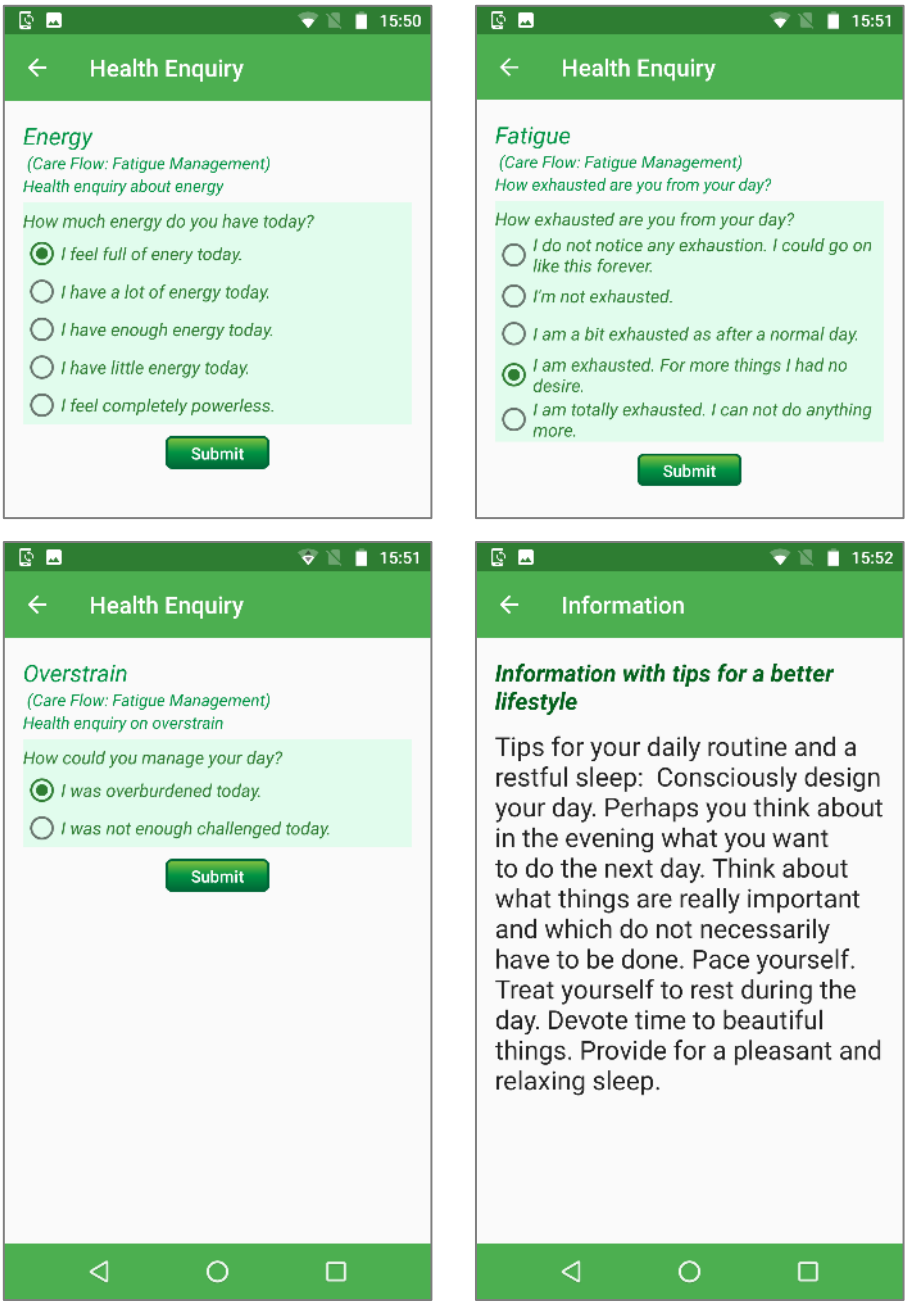

Figure 43: Fatigue Management: Health enquiry about energy in the morning and exhaustion in the evening.

Several services provided by the Care Flow Engine and the model repository are exclusive decision support services for health professionals, The following screenshots illustrates the behaviour of the app iSupportMyPatients when the user subscribes to the health service *St Gallen-Oncotype DX*.

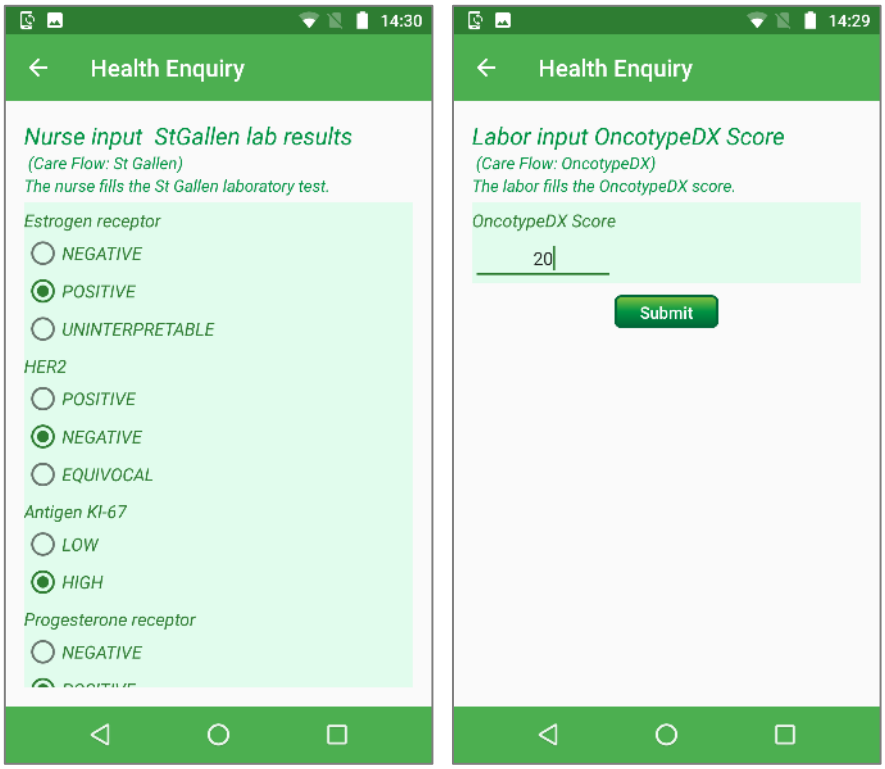

Figure 44: Health enquiries of the Care Flow 'St.Gallen-OncotypeDX' to suggest suitable treatments for breast cancer patients based on the subtype of their tumour following the St Gallen consensus and an Oncotype DX laboratory test.

7 Drug Self-Management with app iManageMyHealth

The app iManageMyHealth does not only allow the patient to subscribe to self-management services provided by the central decision support system, but it supports also the management of drugs. Patients can easily compose a medication plan with reminders to drug intakes. They are warned for drug-drug-interactions by leveraging two external internet based drug repository services, the Canadian DrugBank of OMx Personal Health Analytics Inc. and the German SCHOLZ Datenbank of ePrax AG and a built-in list of Italian drugs.

7.1 Summary of use case and requirements from deliverables D2.2 and D2.3

Patients' ability to manage medications known as medication management capacity (MMC) is one of the main drawbacks to effective care and disease control. It includes the capability to identify medications correctly and to describe how they should be taken. A mobile application for drug self-management, which is a part of this demonstrator, enables patients to successfully manage an increasing number of prescriptions and avoid dangerous drug interactions and side effects. Drug self-management through a mobile app shall improve patient's medication management capacity and shall include the following characteristics:

- *Patient's Profile* – contains patient's data (e.g. user name in iManageCancer, password (encrypted), date of birth, weight, gender and other options like pregnancy and smoking)
- *Medication Plan* – includes information about prescribed drugs
- *Drug Taking History* - includes information about all drugs which patient has already taken. This option is not yet implemented in the app.

- *Drug-Drug Interaction Checker* – checks if substances of drugs affect their activity when they are administered together. This action can be synergistic (when the drug's effect is increased) or antagonistic (when the drug's effect is decreased) or a new effect can be produced that neither produces on its own. This functionality requires that such information is available in public or industrial drug databases.
- *Drug Contraindications Checker* – checks if the patient should not use a drug because of his disease and comorbidities. Some treatments may cause unwanted or dangerous reactions in people with allergies, high blood pressure, or pregnancy. For example, Isotretinoin, a drug used to treat acne is absolutely contraindicated in pregnancy due to the risk of birth defects. Certain decongestants are contraindicated in people with high blood pressure and should be avoided. This option is not yet implemented in the iManageMyHealth app as. It requires that corresponding structured information is available in a public or industrial drug database.
- *Drug Information Receiver* – provides structured information about a drug (like instruction leaflet) in the patient's mother tongue.
- *Drug Taking Reminder* – reminds the patient about taking a drug

The corresponding scenario description for medication management support comprises the following functionality:

1. The patient can create and manage his medication plan
    i. The patient can add a new drug to the *Medication Plan*
    ii. The patient can edit data of drugs already included in the *Medication Plan*. He can manage a drug doses depending on symptoms and results of health assessment tools.
    iii. The patient can delete a drug in the *Medication Plan*. The drug will be not really deleted, but marked as "deactivated". In this case, it will be possible to use the available drug description for deciding to use or not to use it later according to a reason given for the "deleting" of the drug from the list.
2. The patient receives a reminder message when it is time to take a drug and confirms it after taking the drug.
3. Taking a drug will be documented in the *Drug Taking History*
4. The patient can access information about a drug (drug instruction leaflet), check for drug-drug interaction and drug contraindications in relation to his disease and comorbidities.

The resulting use cases that belong to this scenario are presented in this overview.

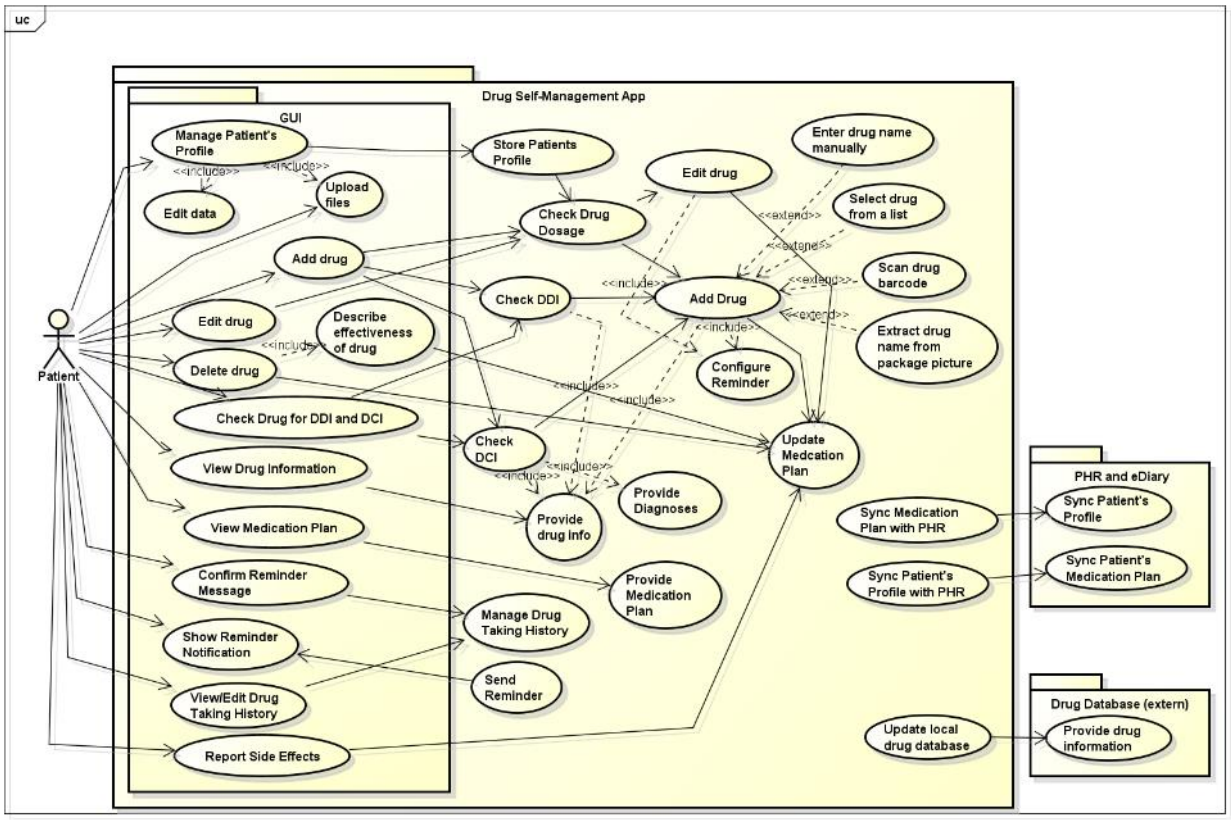

Figure 45: Use cases diagram for Drug Self-Management.

The following table lists the use cases presented in Figure 12 and gives an indication whether they have been implemented in this demonstrator. As one can see we do not check drugs for contra indications based on the patient's self-reported profile as this is not supported by the external drug databases that we use.

ID	Name	Description	Priority	Imple-mented?
UC.DSM.1	ADD DRUG	Patient can add a drug to the medication plan, which will be checked for drug-drug interactions and drug contraindications according to patients' diseases. The drug doses will be checked too. The functionality will be available in the offline mode as well: The added drug will be labelled as "not yet validated". As soon as the internet connection is available again, the checking will be performed.	required	Yes Checks for contraindicati ons not supported. Exception in offline mode not yet implemented
UC.DSM.2	EDIT DRUG	Patient can edit data for a drug in the medication plan.	required	Yes
UC.DSM.3	DELETE DRUG	Patient can delete a drug in the medication plan.	required	Yes
UC.DSM.4	ADD PREVIOUSLY DELETED DRUG	Patient can add a previously deleted drug to the medication plan, which will be checked again for drug-drug interactions and drug contraindications according to patients'	optional	No

		diseases. The drug doses will be checked again too.		
UC.DSM.5	VIEW DRUG INFORMATION	Patient can request drug information for the medication plan.	required	Yes
UC.DSM.6	CHECK FOR INTERACTIONS	Patient can check drugs for interactions	required	Yes
UC.DSM.7	CONFIRM REMINDER MESSAGE	Patient can receive a notification about a drug intake	required	Yes
UC.DSM.8	VIEW/EDIT DRUG TAKING HISTORY	Patient can view the history of drug intakes.	required	No
UC.DSM.9	EDIT PATIENT'S PROFILE	Patient can edit own profile in the app.	required	Yes partially
UC.DSM.10	REPORT SIDE EFFECTS	Patient can report side effects occurred when taking drugs.	optional	No

In consequence, D2.3 lists 16 functional and non-functional system requirements which were derived from these use cases. They are briefly listed in the following table.

ID	**Name**	**Description**	**Imple-mented?**
REQ.DSM.1	Medication plan – add a new drug	The app should provide a GUI for adding a new drug to the medication plan.	Yes
REQ.DSM.2	Medication plan – edit a drug	The app should provide a GUI for editing a drug tin the medication plan.	Yes
REQ.DSM.3	Medication plan – delete a drug	The app should provide a GUI for deleting a drug in the medication plan.	Yes
REQ.DSM.4	Medication plan – add previously deleted drug	The app should provide a GUI for reusing a previously deleted drug in the medication plan, which was not really deleted, but labelled as deleted.	No
REQ.DSM.5	View drug information	The app should provide a GUI for viewing information about a drug.	Yes
REQ.DSM.6	Check drug for interactions with other drugs	The app should provide a GUI for checking drug-drug interactions.	Yes
REQ.DSM.7	Drug intake notification handling	The app should provide a GUI for receiving and handling of notifications for drug intakes.	Yes
REQ.DSM.8	Management of drug taking history	The app should provide a GUI for management of the drug taking history	No
REQ.DSM.9	Management of patient's profile	The app should provide a GUI for management of the patient's profile.	Yes, partially
REQ.DSM.10	Appropriate usability of the drug self-management app.	The drug self-management app should be intuitive and easy to use for a patient.	In progress
REQ.DSM.11	Using of the drug self-management app without an internet connection	The drug self-management app should be functional without an internet connection for accessing the iManageCancer PHR and a drug database.	Yes, partially

REQ.DSM.12	Synchronizing of the patient's profile and the medication plan with iManageCancer PHR	The drug self-management app should synchronize the patient's profile and the medication plan with iManageCancer PHR as soon as the internet connection is available.	Yes, partially.
REQ.DSM.13	Secured access to the iManageCancer PHR	The drug self-management app should provide a secure access to the iManageCancer PHR by a user authentication. User should have an authority as a patient.	Yes
REQ.DSM.14	REST services for communication with iManageCancer PHR	The drug self-management app should communicate with the iManageCancer PHR via REST services for synchronizing a medication plan and a patient's profile.	Yes
REQ.DSM.15	REST services or HTTP connection for communication with external drug database	The drug self-management app should provide an access to an external drug database.	Yes
REQ.DSM.16	Medication plan – reporting of side effects	The drug self-management app should provide the possibility to edit, store and report side effects of drugs.	Yes

The table above shows that all functional and non-functional requirements have been realised in this demonstrator except of REQ.DSM.4 and REQ.DSM.8.

7.2 Internal architecture

7.2.1 SQLite database

The database tables are described in deliverable D5.2.

7.2.2 Notifications for drug intakes

The module *My Drugs* of the app allows the patients to define intake times for drugs in the medication plan. According to this definition, the patient will receive scheduled notifications about the drug intakes. This functionality can be activated (default setting) and deactivated in the GUI for app settings.

The notifications functionality essentially uses the alarm system service provided by Android to set alarms and notification system service for notifications. In order to avoid affecting user interface responsiveness we have implemented this functionality as a background service by using the *AlarmManager* class to perform time-based operations outside the lifetime of the app, e.g. scheduling to run any function in the future. E.g. dynamically create repeated notification alarms for the drugs in the medication plan. The service uses the *IntentService* class, which provides a straightforward structure for running an operation on a single background thread. This approach minimizes the app's resource requirements because we can schedule drug intake notifications without relying on timers or continuously running background services. Registered alarm also works if device is asleep but it does not work for reboot and turned off mobile. AlarmManager fires these events at set times and/or intervals. We are using this class in conjunction with broadcast receiver to start services and perform showing notifications for drug intakes to the patients. The alarms operate outside of the app, so they trigger notification events even when the app is not running, and even if the device itself is asleep.

The following classes have been implemented:

1. Activity class which uses the AlarmManager to set the alarm according to the drug intake times in the medication plan and send notification on alarm trigger. AlarmManager holds a wait lock on CPU as long as receiver's onReceive() method is executing. This feature is because AlarmManager has to ensure that phone will not sleep till it is broadcasting.
2. AlarmReceiver which is a WakefulBroadcasReceiver that receives the alarm trigger on set time. From here we create notifications when the alarm gets triggered. We initiate three type of notifications:
    a. show a message to user,
    b. play the alarm ringtone
    c. vibrate the mobile device

    The receiver will start the following IntentService to send a standard notification to the user.
3. Class for restarting the notification functionality: by default, all alarms are cancelled when a device shuts down. To prevent this from happening, we have designed the app to automatically restart a repeating alarm if the patient reboots the device. This ensures that the AlarmManager will continue doing its task without the patient needing to manually restart the alarm.

7.2.3 Supported external drug data bases and interfacing

This component leverages two external internet based drug database services and an integrated list of Italian drugs to select drugs from a list, to present information on drugs and to check contraindications. Patients are warned for drug-drug-interactions by leveraging two external internet based drug repository services, the Canadian DrugBank of OMx Personal Health Analytics[4] Inc. and the German SCHOLZ Datenbank of ePrax AG[5]. The Canadian DrugBank does not only contain approved drugs for the US and Canadian market but also includes drugs registered at EMA for the European market. However, the brand names of the different drugs are those of the UK market. The German Scholz Datenbank lists those drugs that are approved in Germany. In consequence, the app iManageMyHealth uses the online services of the Scholz Datenbank for German users. For Italian users we included in our app the official list of drugs published by the Italian Medicines Agency under http://www.aifa.gov.it/content/dati-sulle-liste-dei-farmaci-open-data. For smartphones with other language preferences the online services of the Canadian DrugBank are used.

It shall be noted that the Italian drug data that are used by iManageMyHealth does not include structure information to check interactions. In consequence this feature is disabled in the Italian version of the app. Further to this, the external drug databases have disclaimers and restrictions concerning the use of their database due to the fact that the drug information might be wrong, obsolete or incomplete. Similar restrictions in use and disclaimers are presented to the user in our app. Patients must be made aware that drug information might be wrong, obsolete or incomplete and that they shall always discuss eventual interactions of their drugs with their treating physician.

To this moment it is also not possible to check drugs also for contraindications. The external drug databases used cannot provide such information in a format which could be processed.

7.3 User interface functionality

In the following, we explain the features provided by the user interface of the module for drug self-management in the app iManageMyHealth.

[4] https://omx.io/

[5] http://scholz-datenbank.de/

Select ‘My Drugs’ option in the main menu. A list of drugs entered in the patient’s medication plan is opened. When this module is started for the first time the list is empty.

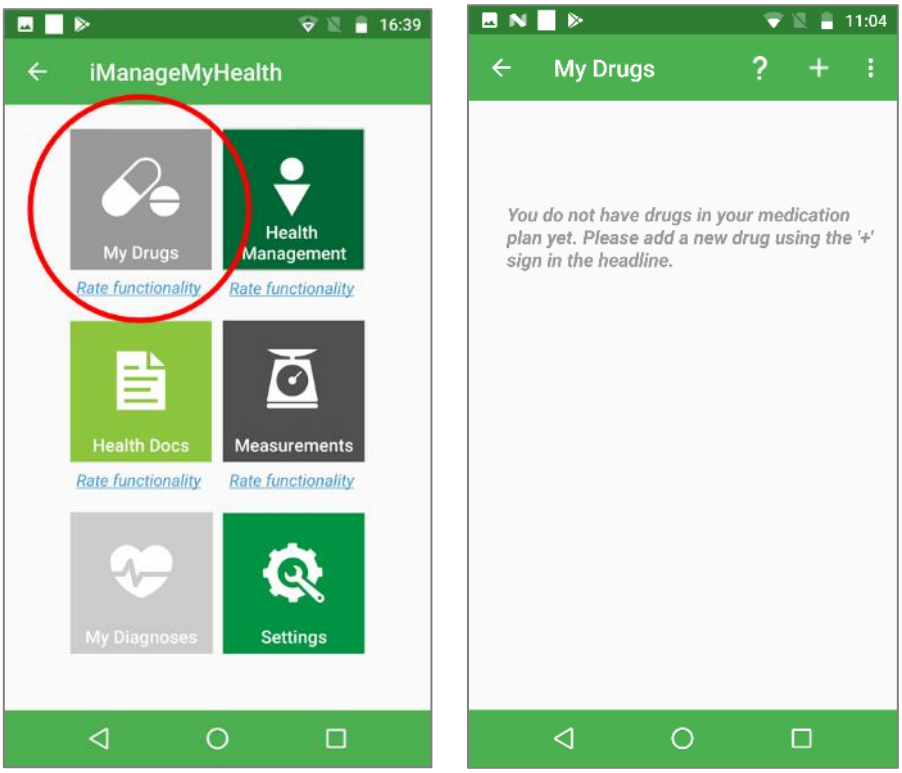

Figure 46: Empty medication plan after first start of the app.

Add a new drug

The patient can add a new drug to the medication plan using the ‘+’ sign on the title row of the app. The page for entering drug information is shown.

Figure 47: Entering new drug data.

The patient can take pictures of his drug package and for the drug itself when clicking on the grey rectangles. He can start to enter the name of the drug into the field for the drug name. After four characters an autocompleting text with a list of drugs starting with the entered characters is shown. This list is received from an external website based drug data repository or in case of Italian language from integrated drug information. The patient can scroll through the list and find his drug in the list. The smartphone must be connected to the Internet to use services from the external drug data repositories.

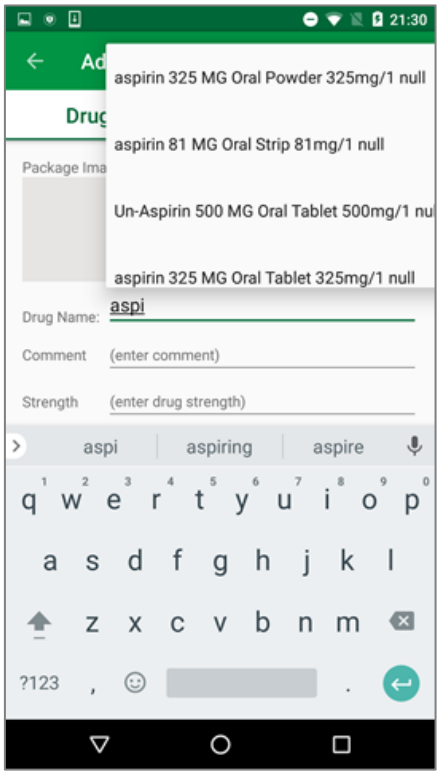

Figure 48: Autocompleting list for selecting a drug name.

The drug data is automatically entered into the corresponding fields. The patient can add some comments to the drug, e.g. about intake conditions.

When selecting the tab *Drug Intake Times*, the patient can select and edit drug intake times from a list of predefined settings for receiving reminder notifications.

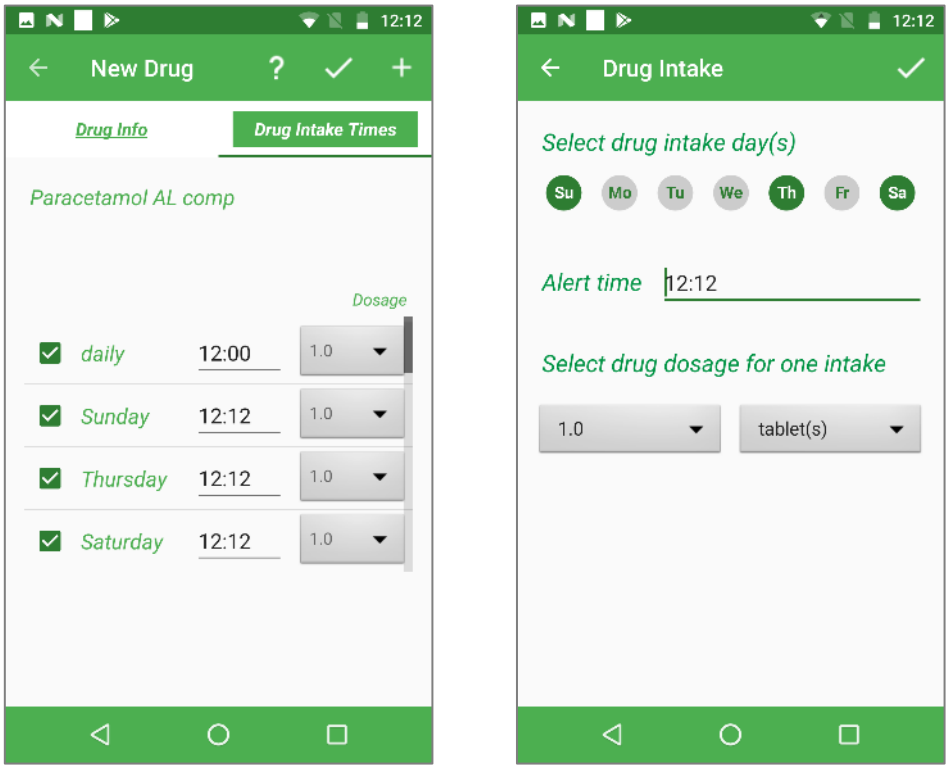

Figure 49: Defining drug intake times.

The patient can define other intake times by selecting the '+' sign on the title bar of the app. The new intake times will be added to the list of the drug intake times.

The patient has to complete adding of the new drug by using the 'save' – sign on the title bar of the app. The drug will be shown in the medication plan.

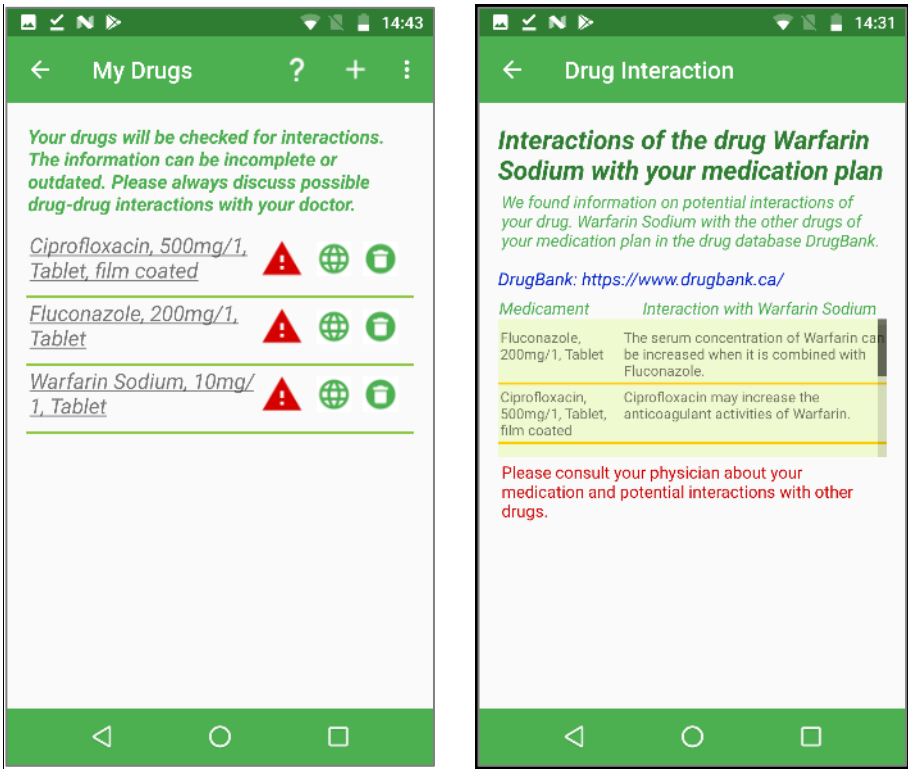

Figure 50: Drugs in the medication plan that can interact with each other. A warning symbol is presented.

The drugs are checked for interactions with each other. The following signs are used for showing the result of the drug-drug interaction checking:

- The system could not find interactions of this drug with other drugs of the medication plan.

- The drug could not be checked for interactions, e.g. because of the missing internet connection or accessing the external drug database was not possible.

- Drug not found in the drug database. In consequence, the system cannot search for interactions with the other drugs in the medication plan.

- A warning sign: The system found interactions of this drug with other drugs of the medication plan. The details about the detected interactions can be viewed by clicking on the warning sign.

If the patient has an iManageCancer account information about the new added drug will be sent to the patient's personal health record on the iManageCancer platform (iPHR).

Editing drugs

By touching the drug name the patient can edit the drug information. Currently, one can edit the comment field, drug pictures and drug intake times.
If the patient has an iManageCancer account updated information about the drug will also be sent to the patient's personal health record on the iManageCancer platform (iPHR).

Deleting drugs

The patient can delete a drug from the medication plan with the waste bin sign. The drug is not be really deleted in the system's database, but it is labelled as deleted.

If the patient has an iManageCancer account information about the deleted drug will be sent to the patient's personal health record on the iManageCancer platform (iPHR). Additionally, a small questionnaire about the effectiveness of the drug is presented. The patient is prompted to rate the drug and add a comment, if wished. Finally, the patient must confirm deleting the drug.

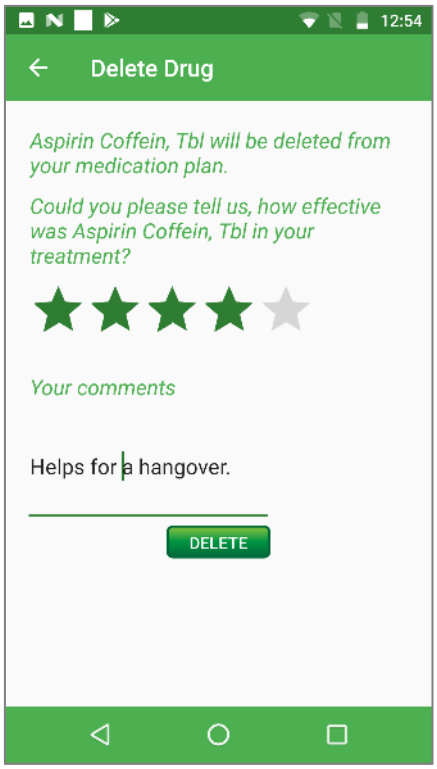

Figure 51: A survey about drug effectiveness when deleting a drug.

View information about a drug

After selecting a drug from the list, the patient is forwarded to a web site containing structured information about the drug in the language configured on the app. Such information is provided by www.beipackzettel.de for German users, www.torrinomedica.it for Italian users and www.drugs.com for other users.

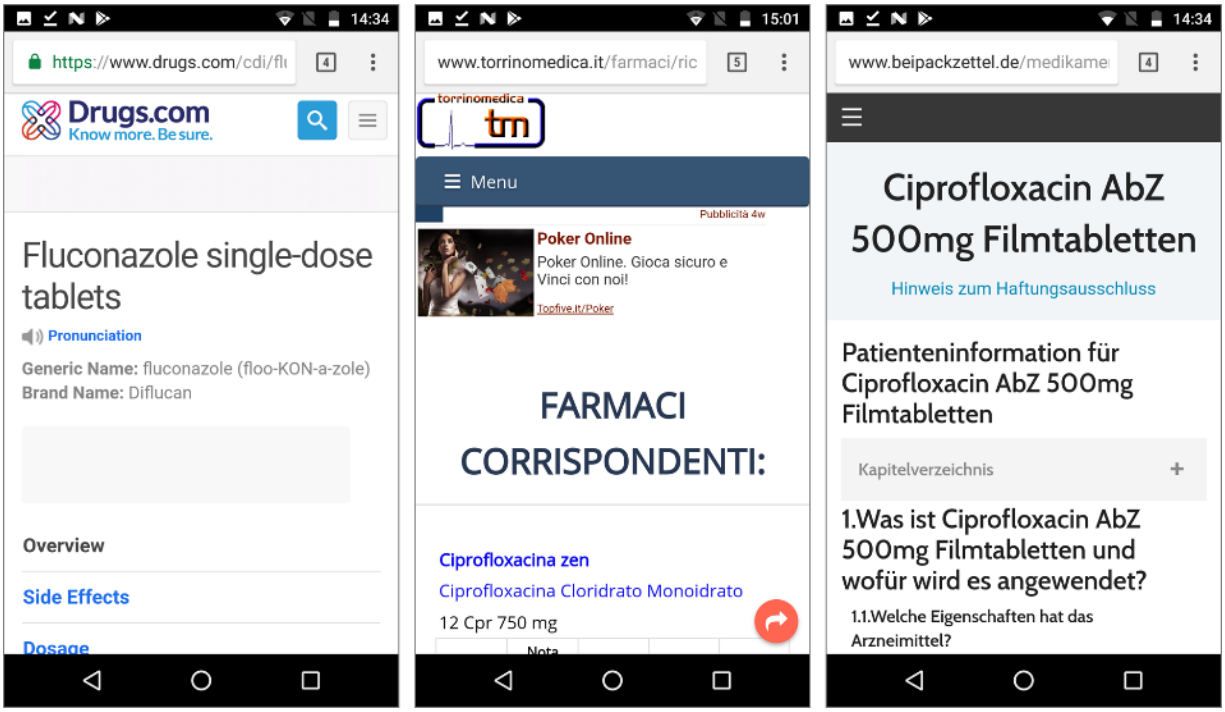

Figure 52: English, Italian and German website that is called to provide further drug information.

Drug intake notifications

The app sends reminders to the patient according to the defined drug intake times. They are shown on the top of the device screen with the icon ■. The full notification text is shown when swiping the icon down.

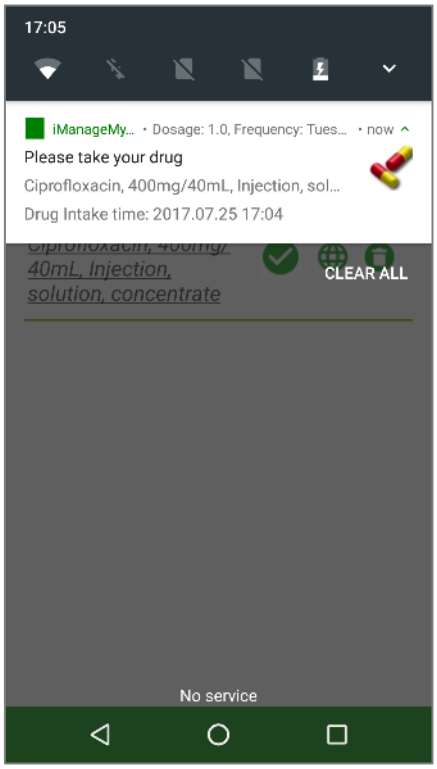

Figure 53: Drug intake reminder.

Rating of the app modules

The patient can send reports about his user experience with the 4 different modules Measurements, Health Docs, My Drugs and Health Management. To do so, he has to select the *Rate Feature* link under the corresponding module on the main menu. A screen to rate the feature will open. The patient can rate the selected module and answer some questions as well as make helpful suggestions for future versions of the module. The results are stored locally and are also sent to the iManageCancer platform. When the patient wants to rate a specific module again, his last rating is shown and he can modify it.

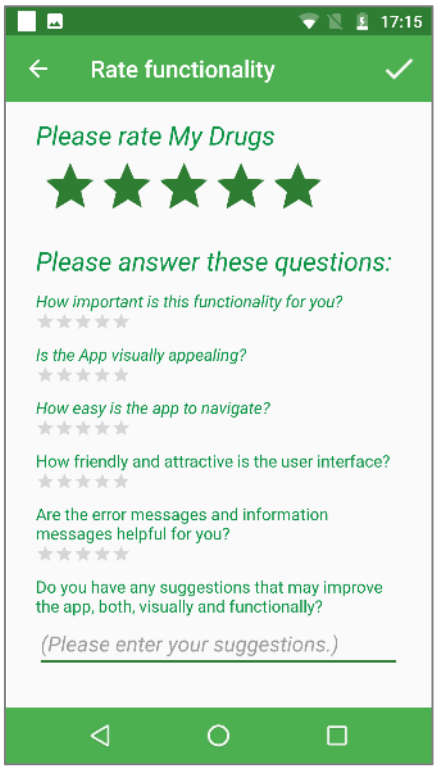

Figure 54: Rating of the module *My Drugs*.

8 Personal Health Information Recommender

The work within iManageCancer project aims to build an advanced, standards-based and scalable semantic integration environment enabling seamless, secure and consistent bi-directional linking of clinical research and clinical care systems which, among others, will empower patients to extract the relevant data out of the overwhelmingly large amounts of heterogeneous data and treatment information.

PHIR [3] is targeted at improving the opportunities that patients have to inform themselves in the internet about their disease and possible treatments, and providing to them personalized information and recommendations. Its goal is threefold: (1) to deliver relevant information to patients, based on their current profile as represented in their personal healthcare record (PHR) data, (2) to ensure the quality of the presented information by giving medical experts the chance to control the information that is given, and (3) to facilitate an easy uptake of the new system by minimizing the necessary manual effort.

The PHIR is integrated into a PHR as a set of individual apps. The patient is able to select the PHIR search app to look for useful information and a medical expert can select the Semantic Annotator app to register high quality web documents. Both the search engine and the automatic recommendation mechanism exploit the individual patient profile and patient preferences to

provide personalized information. To the best of our knowledge PHIR is the only medical information search engine and recommendation system that exploits the individual profile of the patient to provide personalized empowerment.

The Personal health information recommender (PHIR) service enables patients to find higher quality (rather than searching the Internet) and more relevant information and better discern between the different sources of knowledge with respect to quality and relevance. By providing high quality information targeted to their actual information goal we give the patients a better starting point to search for the information that they need. Well-informed patients will also be able to find better sources of information, understand better the content and decide what is relevant for them. Recommends specific informative applications, serious games and literature according to the disease type and psycho-emotional status of the patients to promote encouragement, awareness and reduce anxiety and depression from patients.

Information can be found in three different languages: English, Italian and German. The user can search by writing the question in natural language including the preferred keywords. The results are separated in two categories: text and video. The recommendations are based on both the given question and user's profile. The search results can be rated by user so that the use of the recommender in long term can be improved and even more personalized.

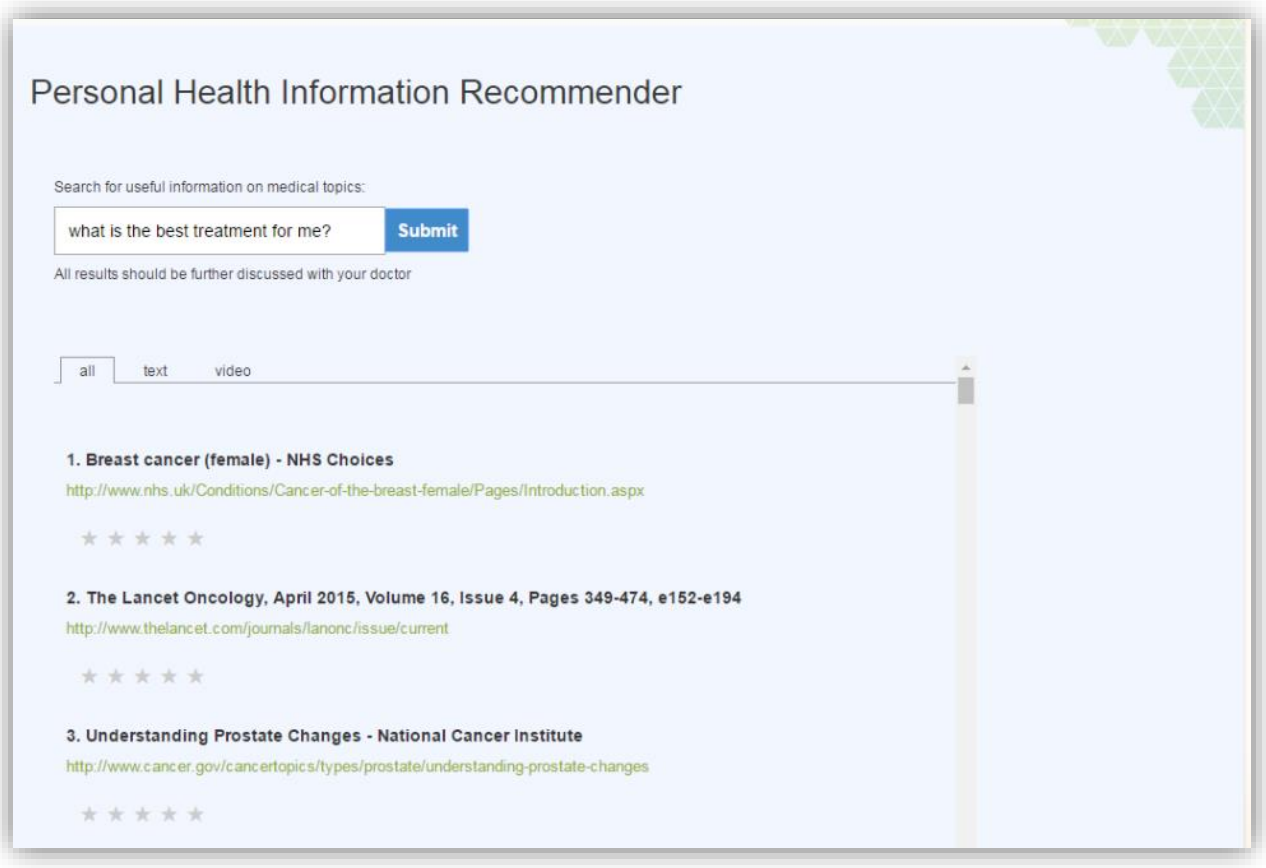

Figure 55: Graphic user interface for PHIR.

An API of the service has also been developed in order to use the recommender from different applications. The result at this case is a list of the answers that can be handled by the request application as needed.

8.1 Summary of use case and requirements from deliverables D2.2 and D2.3

The Scenario SC5 in D2.2 describes personal health information recommender targeted at improving the opportunities that patients have to inform themselves in the internet about their disease and possible treatments, and providing to them personalized information and recommendations. The three main goals are as follows:

1. to deliver relevant information to patients, based on their current profile as represented in their PHR data,
2. to ensure the quality of the presented information by giving medical experts the chance to control the information that is given, and
3. to facilitate an easy uptake of the new system by minimizing the necessary manual effort.

According to SC5 the Personal Health Information Recommender (PHIR) is an engine allowing patients to get personalized search results on various topics of interest related to their disease. To achieve that, domain experts are registering web-sites with useful information which are then parsed and semantically annotated.

The resulting use cases that belong to this scenario are presented in this overview.

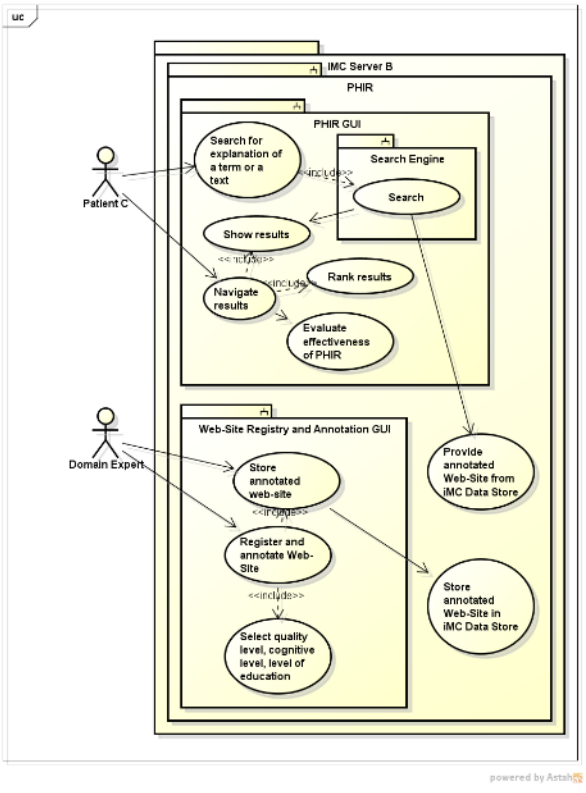

Figure 56: Use cases diagram for PHIR.

The following table lists the use cases presented in **Fehler! Verweisquelle konnte nicht gefunden werden.** and gives an indication whether they have been implemented in this demonstrator. All use-cases have been implemented.

ID	Name	Description	Priority	Imple-mented?

UC.PHIR.1	Web-Site Registry and Annotator.	Registering and annotating high-quality web sites by domain experts	Required	Yes
UC.PHIR.2	Personal Health Information Recommender	The patient selects or enters some text from the interface and personalized information about the aforementioned text is presented.	Required	Yes

In consequence, D2.3 lists 7 functional and non-functional system requirements which were derived from these use cases. They are briefly listed in the following table.

ID	Name	Description	Implemented?
REQ.PHIR.1	GUI for searching recommended information	The platform should provide a GUI for searching for relevant information	Yes
REQ.PHIR.2	GUI for registering and editing web-sites	The platform should provide a GUI for easily registering new web-sites and editing existing ones	Yes
REQ.PHIR.3	Secured access to PHR	The PHR should be secured by a user authentication mechanism	Yes
REQ.PHIR.4	Semantic Annotator services and/or API	There should be a system or an API available capable of annotating web sites with ontology terms. This annotation will be done automatically as the new URLs are entered.	Yes
REQ.PHIR.5	Best relevance of personalised recommendations	The returned results should be as personalized as possible and as relevant as possible with user profile and query	Yes
REQ.PHIR.6	API for requesting recommended information	There should be an API available for calling the PHIR using a phrase as input and returning a link or a text with the recommended information	Yes

The table above shows that all functional and non-functional requirements have been realised in this demonstrator.

8.2 Internal Architecture

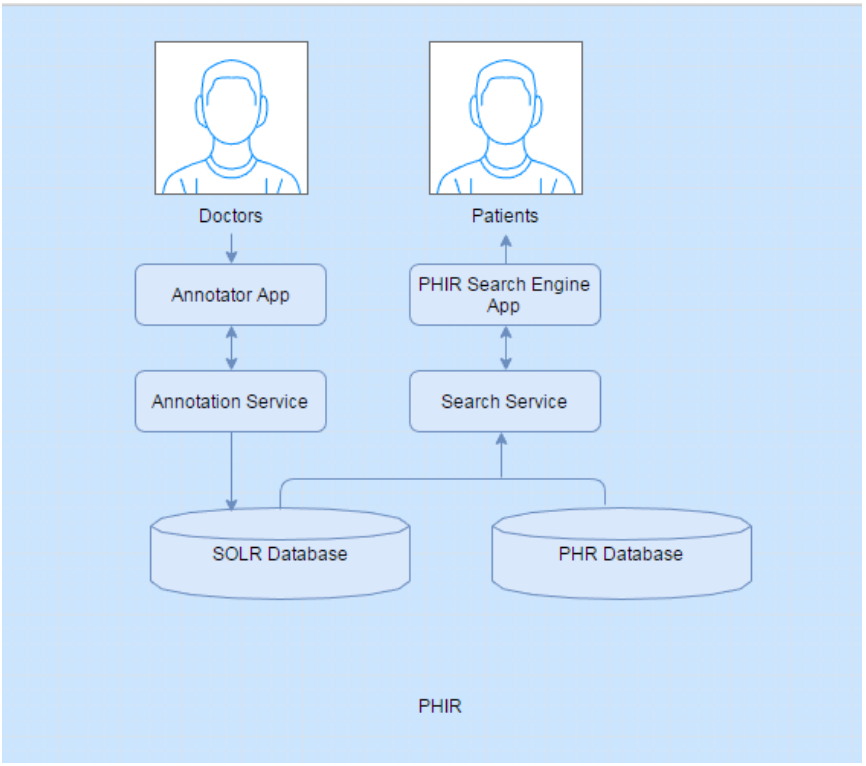

Figure 57: PHIR Internal Architecture.

The architecture of the system is shown in Figure 57 and consists of three layers: The database layer including two different databases, the service layer including two services and the front-end layer including two individual apps. All modules are implemented within the iPHR system through which the whole functionality of the PHIR is offered. Bellow we analyse in detail the main modules.

8.3 The Annotator App & Service

The first step in providing useful information to patients is the identification of useful, reliable, high-quality online health information and its appropriate and efficient use. To cope with the unprecedented volume of healthcare information available on the net, PHIR uses domain experts (doctors etc.) in order to identify and register appropriate web documents.

Using the Semantic Annotator app, an expert is able to register external documents that contain useful information to be further elaborated. Those web documents are high quality web resources (web pages, pdfs, docs etc.) selected carefully by the appropriate experts targeting patients. The interface of the app is shown in Figure 58.

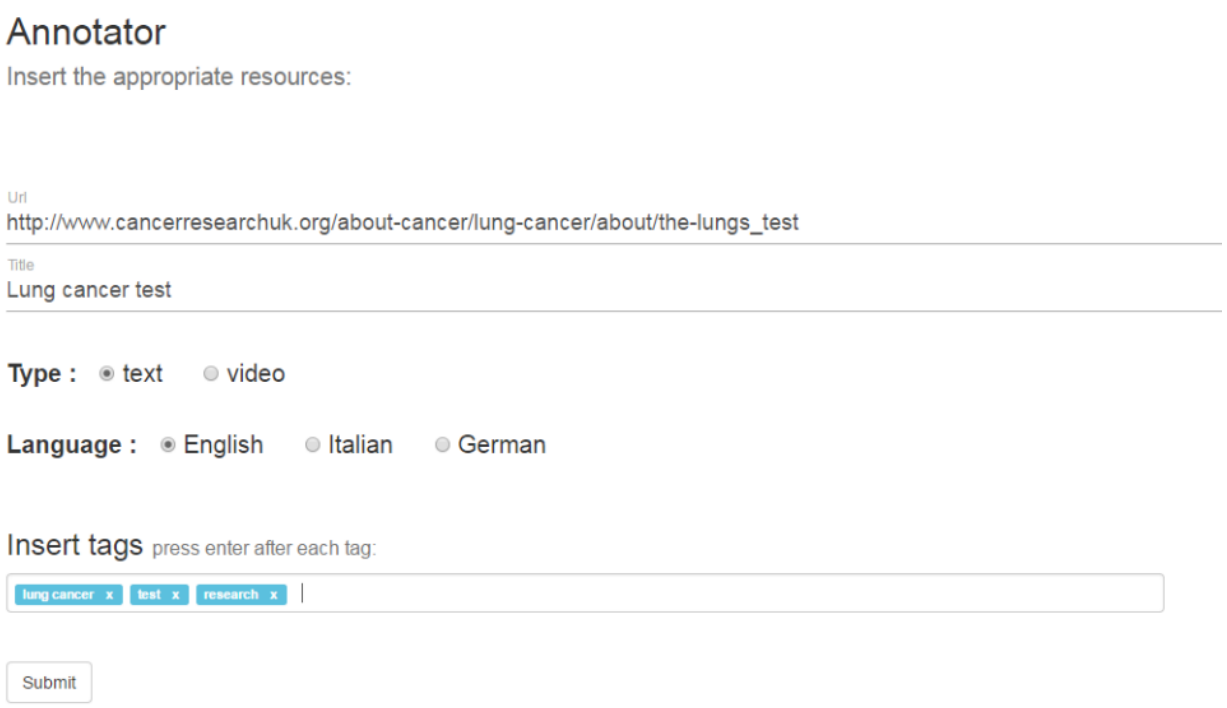

Figure 58: Annotator App.

This database is later used to match documents with patient profiles and patient queries.

8.4 The PHIR Search Engine App

The Personal Health Information Recommender includes implementation of a web rest service which can be used from other applications and a responsive interface that has been integrated in the PHR portal.


```
[
    {
        "url": "https://www.kinderkrebsinfo.de/patienten/fragen_zu_krebs/index_ger.html",
        "title": "Krebs bei Kindern: Fragen zu Krebs",
        "type": "text",
        "lang": "ger"
    },
    {
        "url": "http://www.krebs-kompass.de/",
        "title": "krebs forum",
        "type": "text",
        "lang": "ger"
    },
    {
        "url": "https://www.youtube.com/watch?v=v4KFwiFrmOI",
        "title": "Bewegung gegen Krebs",
        "type": "video",
        "lang": "ger"
    },
    {
```


Figure 59: Example or PHIR rest service results.

The PHIR app that is integrated in the PHR portal sends the user query to the recommendation service and then the app visualizes the returned results. The interface of results includes one more option for the user, to rate the results in order to improve the user-service interaction.

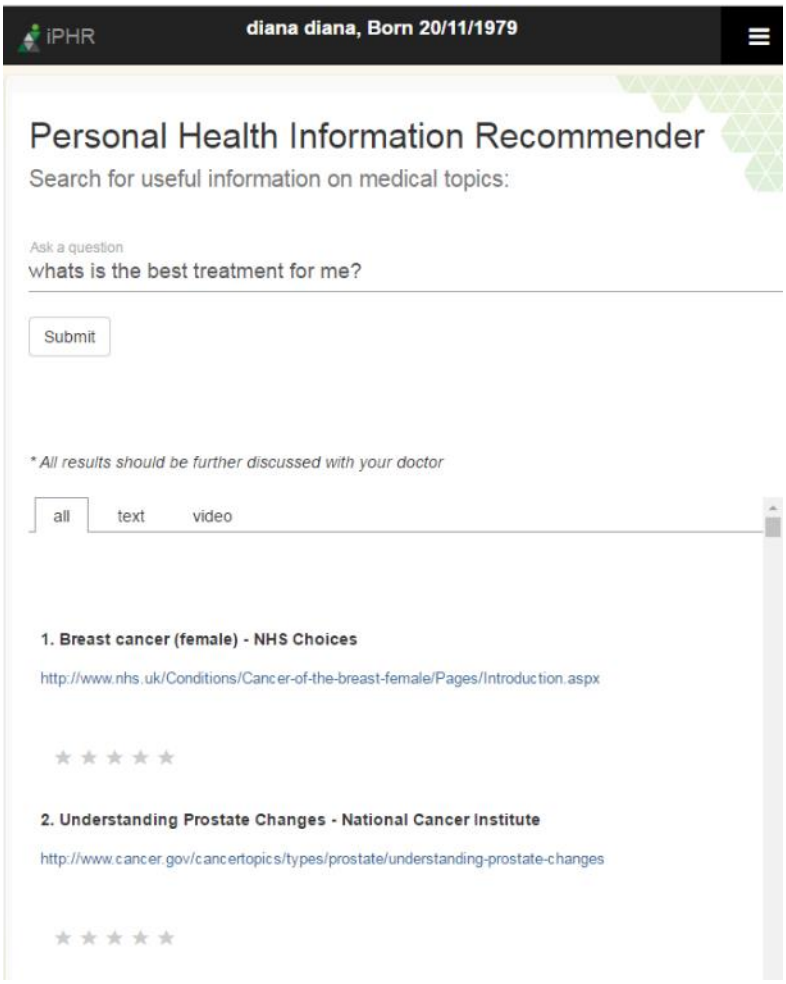

Figure 60: PHIR interface integrated in the PHR Portal.

The data that are available for the system are stored in a SOLR server instance (http://lucene.apache.org/solr/) which improved search time compared to previous version that used MySQL database.

The service interacts with SOLR server in order to query the dataset and retrieve the results and with the PHR portal database in order to retrieve user's profile information which are also used as an input to the query.

9 Decision aid to support patients' participation in consultations

Depending on dispositional and contextual factors, cancer patients may require different amounts of information before they feel in the position to make a decision about their treatment. Most doctors try to provide the amount of information they think their patients need, but doctors and patients often have different opinions about what that information should be. Consistently, information is often reported as an unmet need of patients with different cancer diseases, but in particular in one of the target population for the pilot study of the iManageCancer platform (Davison et al., 1995; Lintz et al., 2003; Harrison et al., 2009; Watson et al., 2016). Furthermore, according to a qualitative study performed within the works of the iManageCancer project, many patients reported little to no possibility to share information and questions with healthcare providers, and that patients felt they often lacked sufficient information in order to make decisions

(e.g., on the therapeutic approach which suited them better or in order to take decisions related to self-management during or after therapy) (Renzi et al., 2017).

A Decision Aid tool for prostate cancer treatment choice is under development in the iManageCancer platform. The purpose of this Decision Aid is to assist men with early-stage prostate cancer in making a treatment decision.

According to SC7 in D2.2, prostate cancer patients are guided through pros and cons of the choice prior to consultation. At the end of the decision aid, patients can write up a list of questions to ask during consultation. Lists of important questions to a particular diagnosis or treatment situations could be created (e.g., questions around prostatectomy or radiotherapy, etc.)

This information can be provided to the clinician in order to guide consultation based on themes that are relevant for the patient and to his family. Expected outcomes from using the decision aid would be greater information comprehension/understanding, increased satisfaction with consultation, and greater choice satisfaction (i.e., reduced regret), and eventually higher empowerment and self-management abilities.

Furthermore, the information provided by the Decision Aid could be combined with patient information about his family, and with the patient profile provided by the Psycho-emotional monitoring questionnaire.

9.1 Summary of use case and requirements from deliverables D2.2 and D2.3

The resulting use cases that belong to this scenario are presented in Figure 61.

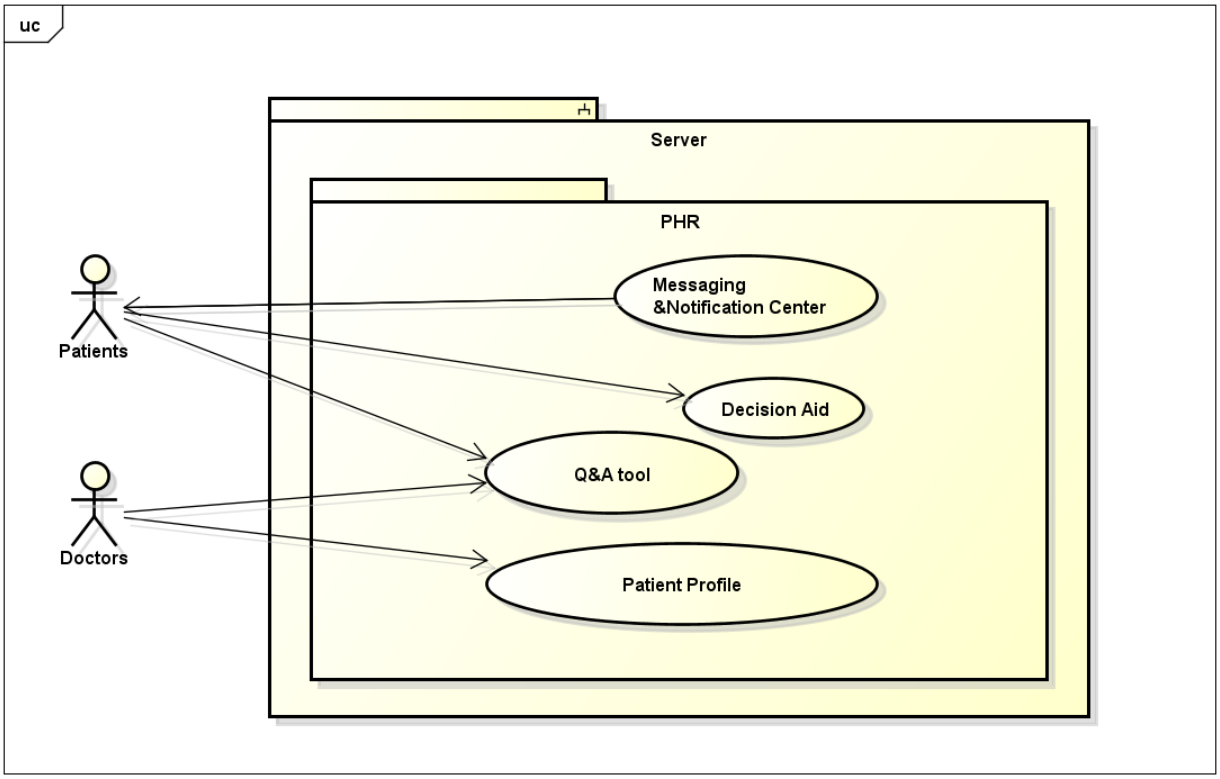

Figure 61: Use cases diagram for Decision Aid to support patients' participation in consultation.

Because the technical implementation of the Decision Aid is still in progress, we present here an overview of all the steps that characterize the Decision Aid. The final implementation in iPHR will be presented in D5.4.

Patient user

- Prior to consultation (at arrival in the hospital or 24 hours in advance) the system reminds the patient to perform a whole or a partial psycho-emotional assessment
- Data are computed and scoring is generated.
- Decision Aids tool is activated.
- The DA will take patient through the following steps:
    - Background information: about prostate, PSA, Gleason Score, etc.;
    - Personal data: patient gives information about his prostate cancer;
    - Initial thoughts: patient considers his treatment preference and factors that affect his choices;
    - Information: patient view information about his treatment options;
    - Rethink: patient reconsiders his treatment preference in light of information presented, and narrow down his treatment options (two options);
    - Conclusion: patient final consideration of his treatment preference between his top two treatment options.
- The patient can write up a list of questions or choose from a list of pre-compiled questions to be submitted to the physician.
- A profile is generated.

Clinician user

- The doctor receives a notification that a profile is ready for a scheduled patient.
- The profile contains indications on areas, which generate significant emotional activation, which areas are evaluated as cognitively relevant for decision, patient's preferences on decision style and self-management and family resources.

10 Conclusions and future work

10.1 Care Flow Engine and related self-management services

The system provides two possibilities to add new or improved Care Flow Plans for self-management services. The first possibility involves the software developer since novel Care Flows are deployed as BPMN activiti resources with updates of the Care Flow Engine. The second possibility is much more elegant because clinical experts can design and deploy novel Care Flow Plans in the system with the Care Flow Designer. We put considerable efforts in this tool, however its current capabilities limit its use to very simple Care Flows. More complex Care Flows can be included in the system with the first possibility. We will continue to extend the capabilities of the design tool.

In the meanwhile, we elaborated realistic management services for pain and fatigue. They shall be further extended, but this must go hand in hand with a closer integration of the Care Flow Engine and the app iManageMyHealth with iPHR.

Further to this, we envisage to extend the functionality of the app iSupportMyPatients to further improve the user experience with this app for health professionals. The related decision support services for health professionals will be evaluated by oncologists in a workshop at Kings College London.

10.2 Predictive models extended to workflows

10.2.1 Febrile Neutropenia (FN)

The Multinational Association of Supportive Care of Cancer (MASCC) risk index score has been implemented. This index is one of most widely accepted examples of a neutropenic complication predication model. The model was developed on a cohort of 756 patients and validated on a 383 patient validation set ((age > 16 years, granulocyte count < 500/mL and temperature > 38°C). Predictive factors (Table 1) were discovered and weighted upon the derivation set using logistic regression with regard to two patient outcomes: favourable (fever for 5 days without a serious complication) or unfavourable (fever for 5 days with serious complication(s) including death). The system has a maximum theoretical score of 26, with a score less than 21 regarded as high risk.

Characteristic	Weight
No/mild symptoms (burden of illness)	5
No hypotension	5
No chronic obstructive pulmonary disease	4
Solid tumour or no previous fungal infection	4
No dehydration	3
Moderate symptoms (burden of illness)	3
Outpatient status	3
Age < 60 years	2

Table 1: MASCC scoring system (Klastersky et al., 2000).

Although the MASCC risk index is a powerful tool for stratifying low risk patients once a febrile neutropenic episode has occurred, it has weak specificity when targeting high risk patients and does not predict individual risk of neutropenic complications ([10] Klastersky and Paesmans, 2013).

A pre-treatment model for neutropenic complications is currently being developed within the iManagecancer project. Several thousand retrospective patient records are being collected from Guy's hospital, London (GSTT). The cohort will contain a broad range of tumour types, including both common solid tumours and malignant lymphoma patients who are undergoing chemotherapy treatment. The patient records will include fields for cancer type, drug regimen, medications, demographics and laboratory information, both blood and microbiology. Patient data will be collected covering several years and contain pre-treatment and cycle-specific information for each patient.

Once the collection is complete the data will go forward to the modelling stage of the work. The predictive risk model primary outcome will be defined as a febrile neutropenia or severe neutropenia event in the first cycle of treatment. Secondary models may investigate FN in subsequent cycles. Feature engineering will be used to create or combine additional variables that we think may be predictive. All variables will be tested in the model for their association with the binary outcome variable and for collinearity with each other.

Validation is an important step in confirming the utility of the model in the patient population. For validation as a GSTT model, the final model will be validated using a 'hold out' set of the patients. These patients will be selected prior to model construction and will only be used for the sole purpose of validating the final model. It may be necessary to recalibrate the model (recalculate the parameter coefficients) for use in specific hospital populations outside of GSTT to obtain optimal performance.

A number of parametric and machine learning algorithms will be tested and the final model will be benchmarked against the published Lyman et al., 2011 [11] model using the GSTT data. Standard measures of performance such as AUC, sensitivity, specificity, PPV and NPV will be reported for comparison. Standard statistical software will be used to analyse the data such as SAS, SPSS or R.

The risk model itself will enable the health professional to enter patient data variables and obtain a personal risk score for the development of neutropenia in the first cycle of treatment. This will form part of an assessment for the needs of that particular patient. If a patient is deemed in a high risk group, several clinical and non-clinical steps can be taken in the treatment of that patient to manage the increased risk.

The model is intended as an aid to the health professional in making a complex multifactorial decision. The model is not intended to replace the clinical decision itself but can be used as a tool to aid in the identification of patients who are at risk of developing FN. Moreover the model allows the risk to be quantified in a reproducible standardised manner for each patient which can then be systematically recorded and incorporated in a clinical care plan.

Many nursing studies have shown that systematic assessment of cancer patients can improve clinical outcomes for cancer patients through the reduction in the incidence of FN, dose reductions and treatment delays ([12] O'Brien et al., 2014;[13] Flores & Ershler, 2010;[14] Miller, 2010; [15] Doyle, 2006;[16] Donohue, 2006).

10.2.2 Hypermodel validation workflow

The objective is to develop a workflow that could be used to validate any model within clinical trials. The idea is to compare, for a given patient, the value proposed by the predictive model and the real value.

Our workflow implementation proposes to use the Tumor size prediction as the model to be validated.

10.2.2.1 Models

The workflow 'Hypermodel validation' contains one model **"*Tumor size model*"**, a new model to be added to the model repository.

10.2.2.2 Models input and output parameters

As the tumor size model is still in research, no input/output has been defined yet.

10.2.2.3 Workflow description

The workflow starts with 2 parallel tasks:

1. The predictive model gives the predicted tumor size based on patient data.
2. The surgeon makes a pre-operative chemotherapy and a surgery, then the real tumor size can be measured.

Then the 2 values, predicted and real, are compared; this comparison process could be done many times within a clinical trial in order to validate the accuracy of the predictive model.

Finally the predicted value is compared to the previous value of the tumor size:

1. Tumor growth, **"Model recommends surgery first"**
2. Tumor shrinkage, **"Model recommends pre-op chemo + surgery"**

10.2.2.4 Implementation with BPMN 2.0

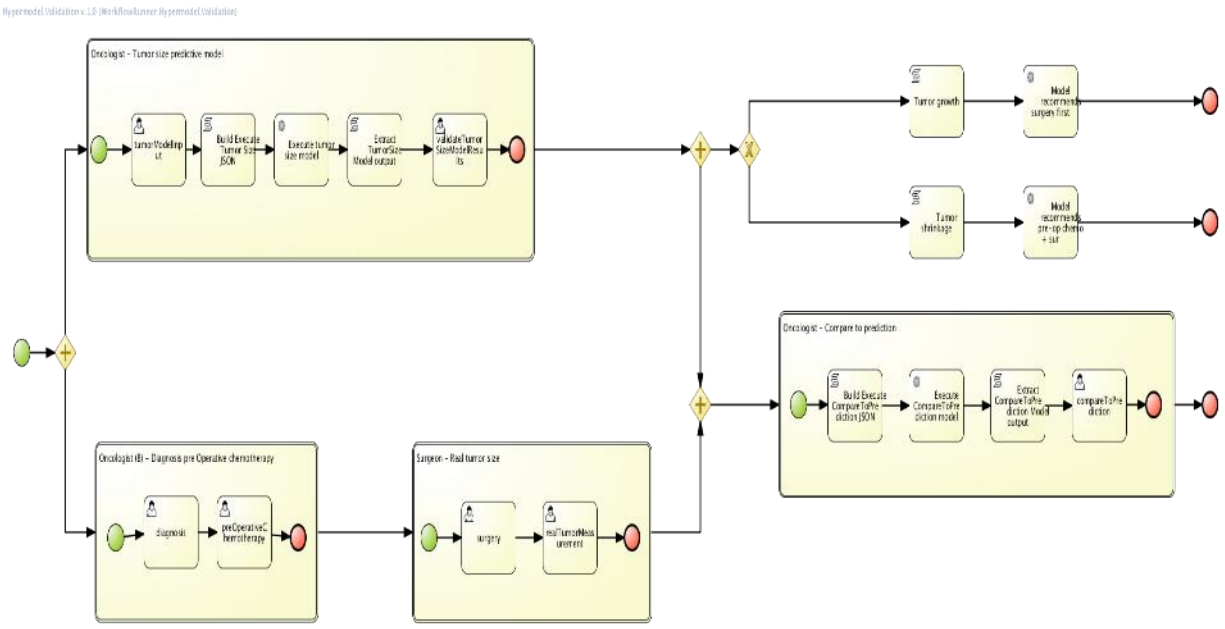

Figure 62: Hypermodel validation for the tumour size predictor model.

10.3 Drug self-management and drug safety

Usability of this component has been improved with a revision of the user interface following the recommendations of citizens with cancer who tested the iManageCancer tools in two sequential end user workshops. In particular, we managed it to incorporate also the official list of approved drugs in Italy for the Italian users of the app.

As future improvements we consider to capture structured information from the patients about side effects that they experience when taking their drugs. Further to this, with the aim to increase drug safety we will investigate whether health information of the patient can be used to identify potential contraindications from which the patient can be warned.

10.4 Personal Health Information Recommender

Searching time has been improved in the latest version and more data resources haven been available in the system. German and Italian resources have also been part of the data set. The interface is responsive and easy to use by different devices.

As future improvements can be seen the extension of the dataset and the extension of the algorithm in order to include more parameters into account for retrieving the corresponding results.

11 References

[1] Berg GM, Hervey AM, Atterbury D, Cook R, Mosley M, Grundmeyer R, Acuna D. Evaluating the quality of online information about concussions. JAAPA. 2014;27(2):1-8.

[2] McMullan M. Patients using the Internet to obtain health information: how this affects the patient–health professional relationship. Patient education and counseling, 2006;63(1): 24-28.

[3] Kondylakis H, Koumakis L, Rüping S, Kazantzaki E, Marias K, Tsiknakis M. PHIR: A Personal Medical Information Recommender, Medical Informatics Europe (MIE), 2014. pp. 1193.

[4] Jurafsky D, Martin JH, Kehler A, Vander LK, Ward N. Speech and language processing: An introduction to natural language processing, Computational Linguistics, and Speech Recognition, 2000.

[5] Melucci, M.: Vector-Space Model. In: Liu, L., ÖZsu, M.T. (eds.) Encyclopedia of Database Systems, 2009, pp. 3259-3263. Springer.

[6] Agliardi A, Jadad AR. Examination of instruments used to rate quality of health information on the internet: chronicle of a voyage with an unclear destination. BMJ. 2002;324:569–73.

[7] Scullard P, Peacock C, Davies P. Googling children's health: reliability of medical advice on the internet. Archives of disease in childhood. 2010;95(8):580-582.

[8] Berg GM, Hervey AM, Atterbury D, Cook R, Mosley M, Grundmeyer R, Acuna D. Evaluating the quality of online information about concussions. Journal of the American Academy of Physician Assistants. 2014;27(2):1-8.

[9] Wiesner M, Pfeifer D. Adapting recommender systems to the requirements of personal health record systems. In Proceedings of the 1st ACM International Health Informatics Symposium. 2010. pp. 410-414.

[10] Klastersky J, Paesmans M. (2013) The Multinational Association for Supportive Care in Cancer (MASCC) risk index score: 10 years of use for identifying low-risk febrile neutropenic cancer patients. Support Care Cancer. 21(5):1487-95.

[11] Lyman GH, Kuderer NM, Crawford J, Wolff DA, Culakova E, Poniewierski MS, Dale DC. ,2011 Predicting individual risk of neutropenic complications in patients receiving cancer chemotherapy. Cancer. 117(9):1917-27.

[12] O'Brien, C., Dempsey, O, Kennedy, MJ, 2014 Febrile neutropenia risk assessment tool: improving clinical outcomes for oncology patients. European Journal of Oncology Nursing 18 (2):167-174.

[13] Flores and Ershler, 2010. Managing neutropenia in older patients with cancer receiving chemotherapy in a community setting. Clinical Journal of Oncology Nursing, 14 (1): 81–86.

[14] Miller, 2010. Using a computer-based risk assessment tool to identify risk for chemotherapy-induced febrile neutropenia. Clinical Journal of Oncology Nursing, 14 (1):87–91.

[15] Doyle, A.M., 2006. Prechemotherapy assessment of neutropenic risk. Oncology Nursing Edition, 20 (10):32–40.

[16] Donohue R.B., 2006. Development and implementation of a risk assessment tool for chemotherapy-induced neutropenia. Oncology Nursing Forum, 33 (2):347–352.

[17] http://www.oncoconferences.ch/mm/Consensus_SG-2013.pdf

[18] https://www.ncbi.nlm.nih.gov/pubmed/24110412

[19]http://doi.org/10.1186/s12911-016-0314-3

12 Tables of figures