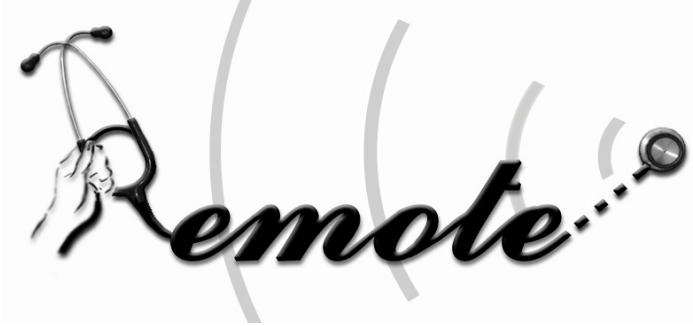
AMBIENT ASSISTED LIVING (AAL) JOINT PROGRAMME

Project

ICT based solutions for Prevention and Management of Chronic Conditions of Elderly People
(REMOTE, AAL-2008-1-147)



Deliverable

D6.3_v2: REMOTE Service and Component Identification and Specification.

Work package No	WP6	**Work package Title**	Integrated services
Task number	T6.3, T6.4	**Task Title**	Service and component identification Service and component specification
Status[1]	D	**Version no.**	v.3.0
Authors[2]	Roberto González, Pedro Muñoz, Rocío Paniagua (ABAMA). Isabel Martí, Alejandro Aracil (TSB), Dionisis Kehagias, Giorgos Zissis (CERTH/ITI), Claudiu Amza (Bluiepoint), Andreas Triantafyllidis (CERTH/IBBR), Taxiarchis Tsaprounis (CERTH/HIT), Michalis Foukarakis (FORTH).		
Document ID	REMOTE_D6.3_IPA		
File ID	REMOTE_D6.3_v2.doc		
Project start and duration	1 June 2009, 39 Months		

[1] Status values: D= draft, F= Final
[2] Per partner, if more than one partner, provide together

EXECUTIVE SUMMARY

The aim of this deliverable is to identify and describe the REMOTE Services after the process of user requirements elicitation and the definition of use cases. The list of services that are described in this deliverable is the result of a thorough process of service identification that takes into account the user needs, as well as existing relevant assets in the market, that are identified after conducting an existing asset analysis. Special care has been taken during this process in order to adhere to the user needs and requirements the identified services.

This deliverable presents work that is related with tasks T6.3 and T6.4. Task T6.3 include decomposition of the business domain into its functional areas and subsystems, analysis of existing systems in order to select viable candidates for providing lower cost solutions and Goal-service modelling to validate and identify services. On the other hand, Task T6.4 deals with the specification of the services to be supported, from a technical perspective.

The deliverable provides an overview of the architecture of the service infrastructure for REMOTE, and classifies the services into various applications domains, as a result of the application of decomposition of the REMOTE business processes into functional areas and subsystems. Moreover, in this deliverable the goal-based service modelling methodology that was adopted for the identification of services is outlined and the results of the existing assets analysis are presented in order to position the identified services with respect to the existing market solutions. Finally, the deliverable presents the list of the identified services and their specifications.

TABLE OF CONTENTS

HISTORY OF THE DOCUMENT

Revision History				
Revision No.	***Author***	***Date***	***Reason for Change***	***The changes***
0,1	Roberto González	02/02/2011	First draft	--
0,2	Rocío Paniagua	04/02/2011	Update	**1**.- Introducción **2**.- Overall Services Architecture
0,3	Pedro Muñoz	09/02/2011	Update	**4**.- Goal Service Modeling
0,4	Isabel Martí	17/02/2011	Update	**6**.- Service Identification and specification
0,5	Alejandro Aracil	17/02/2011	Update	**6**.- Service Identification and specification
0,6	Roberto González	14/03/2011	Update	**3**.-Domain Decomposition **8**.- Service Model
1,0	Rocío Paniagua	07/04/2011	Update	**9**.- Conclusions
1,1	Giorgos Zissis	08/04/2011	Update	**5**.- Existing Asset Analysis
2,0	Dionisis Kehagias	08/04/2011	Update	**5**.- Existing Asset Analysis
2,1	Michalis Foukarakis	03/10/2011	Update	**6.2.7**. - Health Personal Calendar **6.5.3**. - Care Planning Assistance
2,2	Claudiu Amza	20/10/2011	Update	**6.6**.- Medical Contact Centre
2,3	Andreas Triantafyllidis	09/11/2011	Update	**6.6**.- Medical Contact Centre
2,4	Taxiarchis Tsaprounis	14/11/2011	Update	**6.2.8**.- Brain and Skills Trainer
3,0	Alejandro Aracil	15/02/2012	Update	**7**.- Component Identification and specification
FINAL	Alejandro Aracil	27/02/2012	Update	Peer revision modifications
(+hydration)	Alejandro	09/07/2012	Update	Hydration vital sign added

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND DEFINITIONS

AmI	Ambient Intelligence
API	Application Programming Interface
BPM	Business Process Management
CCM	Content Connector Manager
COF	Common Ontological framework
DB	Data base
ECG	Electrocardiogram
GPRS	General Packet Radio Service
HTTP	Hypertext Transfer Protocol
ICT	Information and Communication Technology
LED	Light Emitting Diode
MD	Medical Doctor
OSGI	Open Services Gateway Initiative
PC	Personal Computer
PDA	Personal Digital Assistant
PPU	Portable Patient Unit
SAT	Service Alignment Tool
SOA	Service-oriented architecture
SOMA	Service Oriented Modelling and Architecture
UC	Use Case
UDDI	Universal Description, Discovery, and Integration
WS	Web Service
XML	Extensible Mark-up Language

1 INTRODUCTION

1.1 GENERAL OVERVIEW OF THE REMOTE PROJECT

The REMOTE project aims at the definition and implementation of an ICT-based integration approach for addressing identified needs of elderly users, especially of citizens at risk due to geographic and social isolation in combination with chronic conditions , such as hypertension, arthritis, asthma, stroke, Alzheimer's disease, and Parkinson's disease , and the coexistence of lifestyle risk factors, such as obesity, blood pressure, smoking, alcohol abuse, poor eating / drinking habits, stress, and low levels of physical activity.

In order to achieve its main goal the REMOTE project proposes a number of technologies that encompass both software and hardware elements that are integrated into the overall REMOTE platform in such a way to adhere to the main principled of the service-oriented architecture (SOA)[4] paradigm. More specifically, the project advances the state-of-the-art in fields of tele-healthcare and ambient intelligence (AmI) and target the enhancement of user personal environment with audio-visual, sensor / motoric monitoring, and automation abilities for tracing vital signs, activity, behaviour and health condition, and detecting risks and critical situations as well as providing, proactively and reactively, effective and efficient support at home.

The overall REMOTE platform will be the result of a combined effort that addresses the integration of existing research prototypes and the development of new systems for collecting, recording and analysing health- and context-related data tailored to the specific requirement of the REMOTE project. A number of hardware devices are taken into consideration to be included in the working prototypes that are foreseen for development. These include wearable devices and sensors for detecting intra-oral miniature wetness and jaw movements, body temperature, blood pressure, heart rate, human posture and motion / acceleration recognition, etc., as well as sensors and actuators to be installed in premises (and vehicles) for providing context information, e.g., air temperature, luminance, humidity, human location and motion, etc.

1.2 WORK IN WP6

The REMOTE project is structured in ten work packages. Among these, WP6 "Integrated Service" is dedicated to the design of the service-oriented architecture on which the development of the REMOTE integrated systems will be based. Specific objectives of WP6 include:

- The design and implementation of a software platform to realize the service-oriented architecture and support coordination of existing and integration of new services into a unified platform.
- The selection of a set of service-oriented methodologies in order to provide recommendations about their implementation, according to a set of unified criteria.
- The evaluation of existing service-oriented frameworks, in terms of reusability, functionality and performance.

In particular, deliverable D6.3 is associated with task T6.3: "*Service and component identification"*. This task deals with the identification of the REMOTE supported services, user needs and requirements. In particular the following activities will be covered: Decomposition of the business domain into its functional areas and subsystems, including its workflow or process decomposition into processes, sub-processes, and high-level business use cases; Existing systems are analyzed and selected as viable candidates for providing lower cost solutions to the implementation of underlying service functionality that supports the business process; Goal-service modelling to validate and unearth other services not captured by either top-down or bottom-up service identification approaches.

1.3 THE PURPOSE OF THE REMOTE SERVICES & COMPONENT IDENTIFICATION

The purpose of the REMOTE Services and Component Identification and Specification, and therefore the purpose of this document, is to provide a description about the Services Identification.
In this document are described how to identify, through the SOMA methodology, the services that implements the functionality required to REMOTE.

1.4 DOCUMENT STRUCTURE

This document is structured as follows:

- Section 2 provides an overview of the overall Services Architecture.
- Section 3 describes the domain decomposition from one of three techniques for services identification of SOMA methodology.
- Section 4 provides a detailed description of the goal service modelling, other technique for services identification of SOMA methodology.
- Section 5 describes the existing asset analysis.
- Section 6 provides a detailed description about the Services Identification and specification.
- Section 7 provides a detailed description about the Component Identification and specification.
- Section 8 provides a detailed description about the Service Model.
- Section 9 is the conclusions about the Services Identification and specification.

2 OVERALL SERVICES ARCHITECTURE

2.1 INTRODUCTION

This is a summary of services architecture. For more information, please see the document “REMOTE_D6.2_v1.0.doc”.

As it’s explained in this document, the Service Oriented Architecture (SOA) is an application architecture made up of a collection of services. These services can communicate with each other. The interaction among services is carried out through a communication protocol understandable to both. A SOA service is self-contained and without state and each service aims at providing information or facilitating the business data exchange from a consistence state to another.

The advantages SOA offers are: Interoperability, Reuse, Scalability, Flexibility and Cost efficiency

2.2 SERVICES

This architecture is related with a number of services located in Internet that can be accessed through web services. Focusing on the web services as a basis for SOA, there are several technologies that have been considered.

2.2.1 Web Services Technologies

A Web Service (WS) is a software component that communicates with other applications by means of codified XML messages sent through standard communication Internet protocols such as HTTP.
It is similar to a website, without user interface, giving service to applications. The WS receives requests from an application by means of a XML file. It executes the task demanded and sends another XML file as response to the application requester.

A WS is made up of five different blocks that are shown in the table below:

Web Services Blocks (Function)	Web Service Technology	Name
Discovery	UDDI	Universal Description, Discovery, and Integration
Description	WSDL	Web Services Description Language
Message Format	SOAP	Simple Object Access Protocol
Encoding	XML	eXtensible Markup Language
Transport	HTTP	Hypertext Transfer Protocol

Table 2-1. Web Services Technologies

An overview of a Web Service process and its work is represented in the following figure:

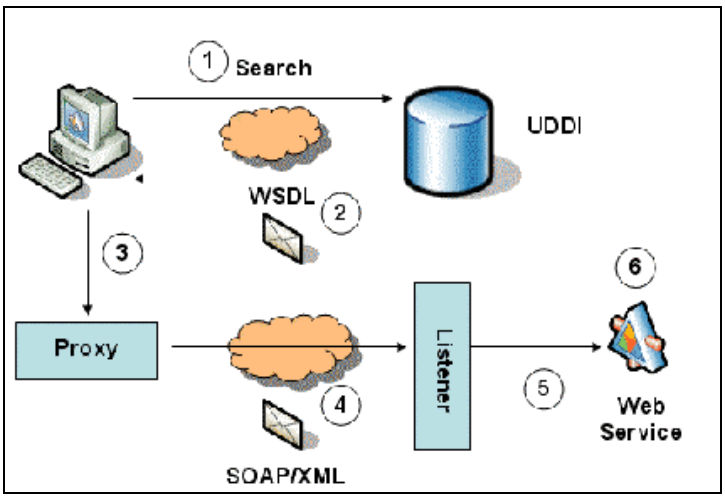


Figure 2-1: Web Service process

The Web Service is published in the UDDI directory so the client has to search for it being previously registered. The client invokes the web service through SOAP as the access protocol. The information that is exchanged is formatted in a XML file. The communication itself is done by means of HTTP protocol. The client receives the response to its request through SOAP again as another XML file transferred through HTTP.

2.2.2 OSGi Service

The OSGi services are services that can be sent and used in a logical unit called bundle. A bundle is an OSGi component made by the developers that can be installed in a framework.

An OSGi service is a java object instance, registered into an OSGi framework with a set of properties. Any java object can be registered as a service, but typically it implements a well-known interface.

OSGi's service register carries out the monitoring of the services consumers and providers. By means of this register the stability of the system is guaranteed. The platform can include bundles dynamically as well as delete them ensuring the critic functions.

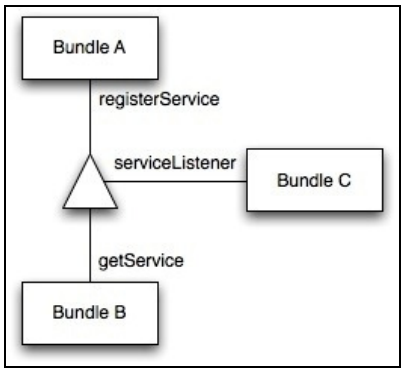


Figure 2-2: Bundle architecture

The above figure shows the interaction among three different bundles. Bundle A works as a service that is registered and can be called by other bundles. Bundle B works as a service that needs something from another bundle meanwhile Bundle C is a service that is only listening what is happening in order to accomplish a possible action requested.

2.2.3 Agents

Agents are computer systems with the ability to carry out autonomous actions within an environment in order to achieve its objectives. They can communicate with each other, setting up agent associations to accomplish a common task.

Agents provide different services that can be complemented by other agents in a multi-agent architecture. The implementation carried out within REMOTE is based in JADE architecture. Further information about agents and JADE can be consulted in D6.1 “REMOTE Services Methodology”.

2.2.4 Local Services

Local services are being developed within REMOTE project. These services are encapsulated within an OSGi bundle. These services can be the ones that analyze results coming from an agent as well as communicate with local databases in order to extract information stored in the database.

2.2.5 User Interface Service

The interaction with the user is carried out through an adaptable interface that is considered as a service itself. It provides a bundle developed in Java with a series of user interface elements that can be reused by each service application to develop their own interface.

The technological implementation is based on the usage of an OSGI bundle providing functions needed for the correct development of the final user interaction layer.

2.3 COMPONENT

"A component is a software object, meant to interact with other components, encapsulating certain functionality or a set of functionalities. A component has a clearly defined interface and conforms to a prescribed behavior common to all components within an architecture"[3].

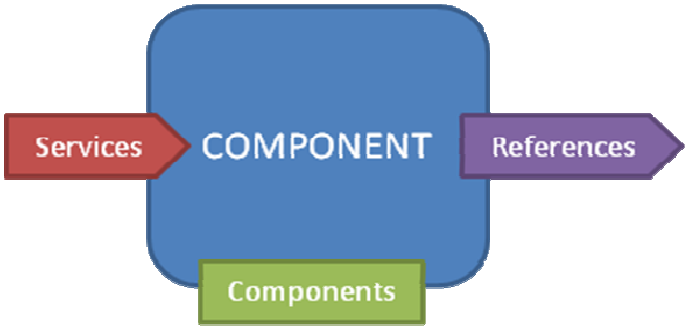


Figure 2-3. Component

A component is able to offer services to the rest of the platform. On the other side, clients do not need to know about the implementation of the component in order to make use of it.

In REMOTE project, this definition adopts a high level meaning because we are talking about services as a group of tele-health and tele-care applications, so we adapt the meaning of a component from a concept point of view.

2.4 METHODOLOGY

2.4.1 REMOTE Methodology

The main objective of this methodology is to integrate all the services within a common platform. Based on the Reference Model for Service Oriented Architecture 1.0 proposed by OASIS European project a few concepts have been considered. The first step is the definition of these concepts that should be present in the common platform.

[3] http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/

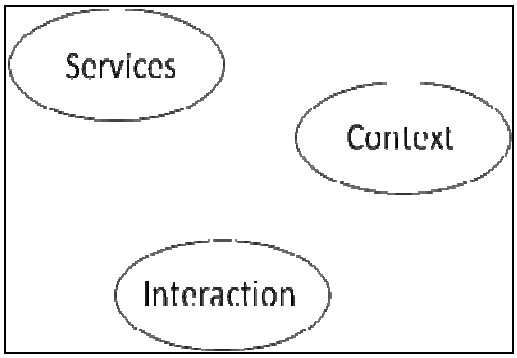


Figure 2-4: Methodology Concepts

Service

The service is the essential element of a Service Oriented Architecture.
There are two main services taking part in the architecture:

- Service provider. Entity that offers a service to other entities
- Service consumer. Entity that uses a service offered by another entity

The relation between them is based in the information exchange. In one side, the service provider has something to offer to the service consumer. In the other side, the service consumer needs something the service provider can offer, so the communication between them is necessary. This communication is based in a request-response process. The service consumer sends a service request to the service provider. The last one answers with the information asked through a service response.

Context

Information gathered from the context provides useful data to understand not only the user but also his context. Services need the information coming from as many sources as possible in order to understand better the field where the user is involved. Context information can be provided by hardware devices, such as sensors, or it can be complementing information from the service knowledge and definition. It is important to highlight that the user is continuously interacting with the environment, so the information coming from it is a useful tool to increase the functionalities of the services offered.

Interaction

Interaction with the user is an essential point in the service methodology, and the information has to be obtained directly from the user and be communicated. For this purpose interaction services are needed. Interaction services should change depending on user type, adapting the communication to each user's capabilities and preferences.

The interaction can be done by several ways, graphical, voice, etc. This interaction also depends on the device used as output for the user and the one used as input for him. Output devices used for the interaction can be graphical

ones such as screens, mobiles, PDAs and input devices can be touchable screens, keyboard, mobiles, PDAs, and so on.

The centre of the service architecture is the services offered to the user. But additional information coming through sensors is also needed in order to understand the environment around the user. Service's operation will not be completed without the user's interaction in order to have control over the services provided. The user must be the centre of the architecture and the activities around should be known by him. He should also have the possibility of changing and choosing among the different possibilities offered from the service point of view.

2.4.2 Soma Methodology: The analysis and design of services

It's necessary to take into account two perspectives in a SOA: the service consumer and the service provider.

Web services are a tactical implementation of SOA. A number of important activities and decisions exist that influence not just integration architecture but enterprise and application architectures as well. These activities are from the two key views of the consumer and provider, described in Figure 2-4 below.

Role	Activities in this role				
Consumer view	Service identification	Service categorization	Service exposure decisions	Choreography or composition	Quality of service
Provider view	Component identification	Component specificaton	Service realization	Service management	Standards implementation
	Service allocations to components	Layering the SOA	Technical prototyping	Product selection	Architectural decisions (state, flow, dependencies)

Figure 2-5 Activities of service-oriented modelling

The activities described above can be depicted to flow within the service-oriented modelling and architecture method, as shown in Figure 5 below:

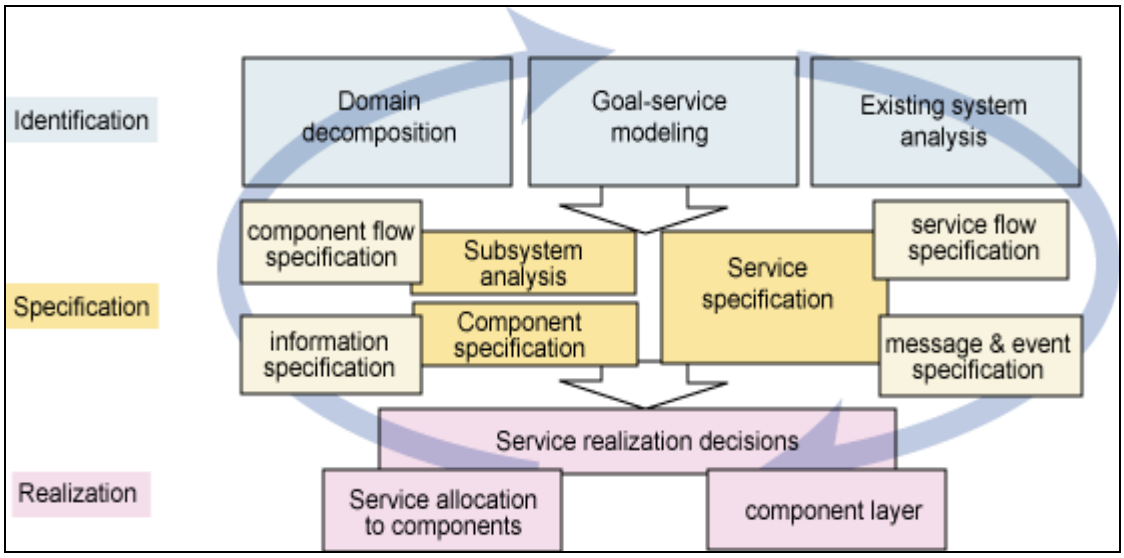


Figure 2-6: The service-oriented modelling and architecture method

The process of service-oriented modelling and architecture consists of three general steps: identification, specification and realization of services, components and flows (typically, choreography of services).

2.4.2.1 Service Identification

This process consists of a combination of techniques of domain decomposition, existing asset analysis, and goal-service modeling. A blueprint of business use cases provides the specification for business services. This process consists of the decomposition of the business domain into its functional areas and subsystems, including its flow or process decomposition into processes, sub-processes, and high-level business use cases.

These are the sub-tasks that make up the identity of the service:

1. Identify services from the objectives: in this phase is to identify all the functional services that will form part of the system, and later link them with all the functional requirements that will support.

2. Develop business process analysis, through a Top-Down approach, it is down from operating activities and to analyze business processes looking for your threads associated, activities and tasks along with identification of the services can support the elements that can be automated

3. Develop the existing asset analysis: applying a bottom-up view, analyzing the services, systems, existing applications and services that may be candidates to provide a low cost solution for implementing some of the functionality of business processes.

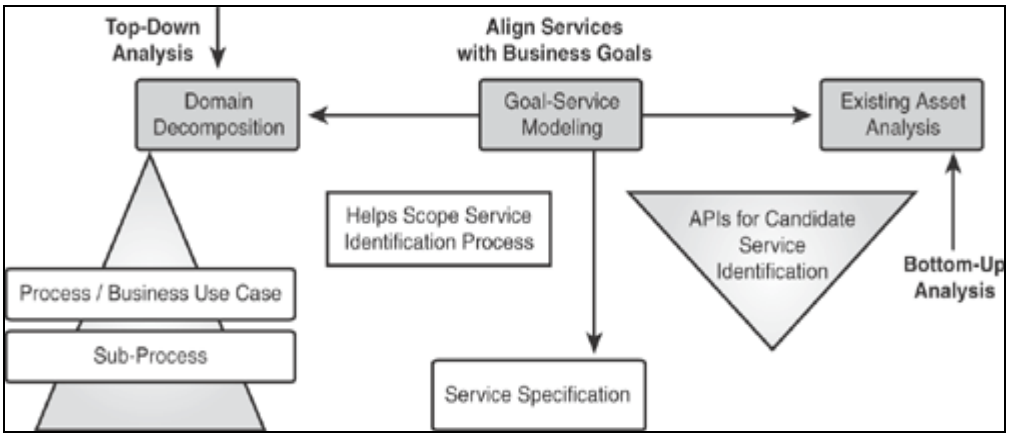


Figure 2-7: The three different techniques for service identification

2.4.2.2 Service Specification

2.4.2.2.1 Service classification or categorization

This activity is started when services have been identified. It is important to start service classification into a service hierarchy. Classification helps determine composition and layering, as well as coordinates building of interdependent services based on the hierarchy.

2.4.2.2.2 Subsystem analysis

This activity takes the subsystems found above during domain decomposition and specifies the interdependencies and flow between the subsystems. It also puts the use cases identified during domain decomposition as exposed services on the subsystem interface. The analysis of the subsystem consists of creating object models to represent the internal workings and designs of the containing subsystems that will expose the services and realize them.

2.4.2.2.3 Component specification

The details of the component that implement the services are specified:

- Data
- Rules
- Services
- Configurable profile
- Variations

Messaging and events specifications and management definition occur at this step.

2.4.2.3 Service realization

This step recognizes that the software that realizes a given service must be selected or custom built. Other options that are available include integration, transformation, subscription and outsourcing of parts of the functionality using

Web services. Realization decisions for services other than business functionality include: security, management and monitoring of services.

2.4.2.3.1 Service allocation

Service allocation consists of assigning services to the subsystems that have been identified so far. These subsystems have enterprise components that realize their published functionality.

Service allocation also consists of assigning the services and the components that realize them to the layers in your SOA. Allocation of components and services to layers in the SOA is a key task that requires the documentation and resolution of key architectural decisions that relate not only to the application architecture but to the technical operational architecture designed and used to support the SOA realization at runtime.

2.4.2.3.2 Component realization

Here, in this task decrease the level of abstraction with respect to the service model. A key factor is to start from what represents our input, the service model. It must create one service component (in the design model) for each service specification (in the service model).

3 DOMAIN DECOMPOSITION

In SOMA methodology, one of three techniques for services identification is through the domain decomposition. This technique is based on decomposing the domain into its business architecture, consisting of the value-chain, business processes and sub-processes and the use cases.

3.1 DOMAIN MODEL

From a business perspective, the domain consists of a set of functional areas. The Remote domain can be decomposed into functional areas across the value-net. The resulting functional areas are often good candidates for implementation as technology subsystems.

The functional areas identified in the Remote architecture are:

- **End-user applications**. This functional area is composed by all the applications which can be installed in the user's devices and provide them with all the Remote functionalities that they need. This functional area can be divided into several sub-functional areas
  - Monitoring, which are applications in charge of monitor the health of the users.
  - Personal self-care, which are applications in charge of helping the user in its personal self-care.
  - Augmented autonomy, which are applications which aims to augment the autonomy of the users
  - Professional healthcare, which are applications for the professional carers
  - Informal tele-assistance, which are applications to communicate users with the carers
  - Medical Contact Centre, which are applications installed in the medical centre
- **Device and Sensors**. This functional area is composed by all the devices and sensors that will be installed in the user's house in order to provide them with the appropriate devices to run the applications and to monitor them through the sensors.
- **Remote software modules**. This functional area is composed by all the software which acts as an intermediary between the end-user applications and the services.
  - User Profile, which is responsible for the communication with the user profile which is located on the server-side
  - Service Prioritization, which sorts the group of services returned as a result of a petition made by a client.
  - Service Provider, which is responsible for the communication with the CCM.
  - Content connector, which is responsible to interact with the Common Ontological Framework and launch the Web discovery process
- **Ontology.** The purpose of the ontology functional area is to provide a common vocabulary that will be used as a semantics infrastructure to

enable true integration and interoperability of heterogeneous web services and devices.

- CCM. The purpose of the CCM is to support automatic integration of Web Services and to receive a request for service by the end-user (client) application via the AmI framework and invoke the appropriate service that returns the required content to the client.
- SAT. Service alignment is a CCM functionality, which is undertaken by service providers through SAT that allows them manually define mappings between their services and the REMOTE services that are defined in the Ontology

- **Services.** This functional area is composed by all the services that the Remote project offers to the final users.

Based on these functional areas, the value-chain of the Remote architecture is shown in Figure 2-1.

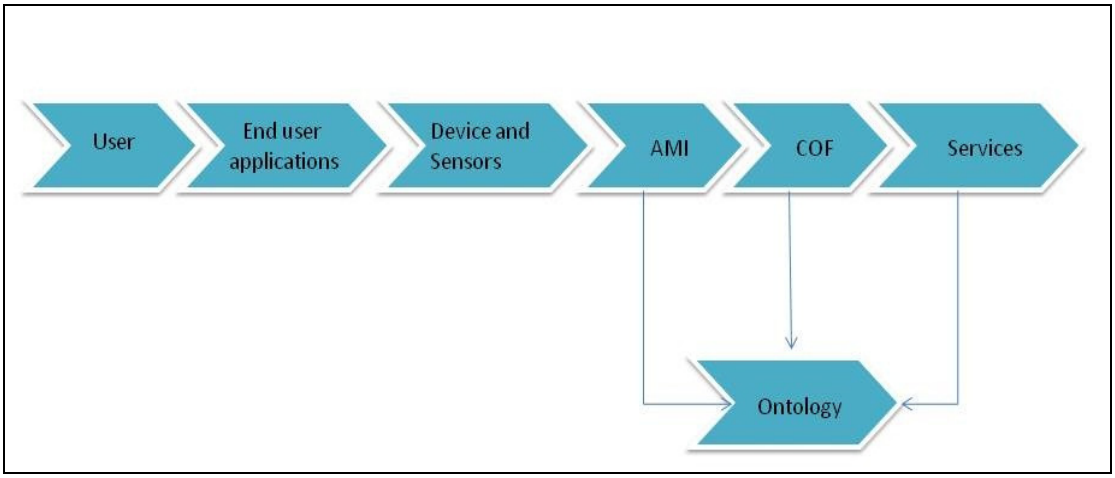


Figure 3-1 Value-chain of Remote architecture

3.2 BUSINESS PROCESS

Business processes are also good candidates for services. Business processes identification have as input business use cases defined in the deliverable *D1.1 User Requirements Definition of REMOTE and Use Cases*, from this use cases model have been identified the following business processes.

3.2.1 BPM / Workflow

The business processes are the following:

- **Elaborate Nutrition Plan**:
  - Name: Elaborate Nutrition Plan
  - Use Case source: UC8.2 Care Planning Assistant
  - Use Case related:
    - UC2.3 Nutritional Advisor

        - UC2.6 Monitoring of user compliance to nutritional plans
        - UC7.3 Patient Nutrition Monitoring
    - Type: Workflow
- **Elaborate Training Plan**:
    - Name: Elaborate Training Plan
    - Use Case source: UC8.2 Care Planning Assistant
    - Use Case related:
        - UC3.1 Brain and skills trainer for memory support
        - UC3.2 Brain and skills trainer for memory assessment
        - UC3.3 Cooperative brain and skills trainer
        - UC7.4 Cognitive problem prognosis
    - Type: Workflow
- **Elaborate Trip Plan**:
    - Name: Elaborate
    - Use Case source: UC4.1 Health Trip Advisor
    - Use Case related:
        - UC6.3 Vital trip assistant
        - UC9.2 Emergency Management Service
    - Type: Workflow
- **Elaborate Patient Medication Plan**:
    - Name: Elaborate Patient Medication Plan
    - Use Case source: UC8.2 Care Planning Assistant
    - Use Case related:
        - UC2.2 Medical Advisor
        - UC2.5 Monitoring of user compliance to medical treatment
        - UC7.1 Record Monitoring
    - Type: Workflow
- **Elaborate Activity Plan**:
    - Name: Elaborate Activity Plan
    - Use Case source: UC8.2 Care Planning Assistant
    - Use Case related:
        - UC2.1 Activity Advisor
        - UC2.4 Monitoring of user compliance to activity plans
        - UC7.2 Activity Monitoring.
    - Type: Workflow

3.2.2 Sequence Diagram

- Elaborate Trip Plan:

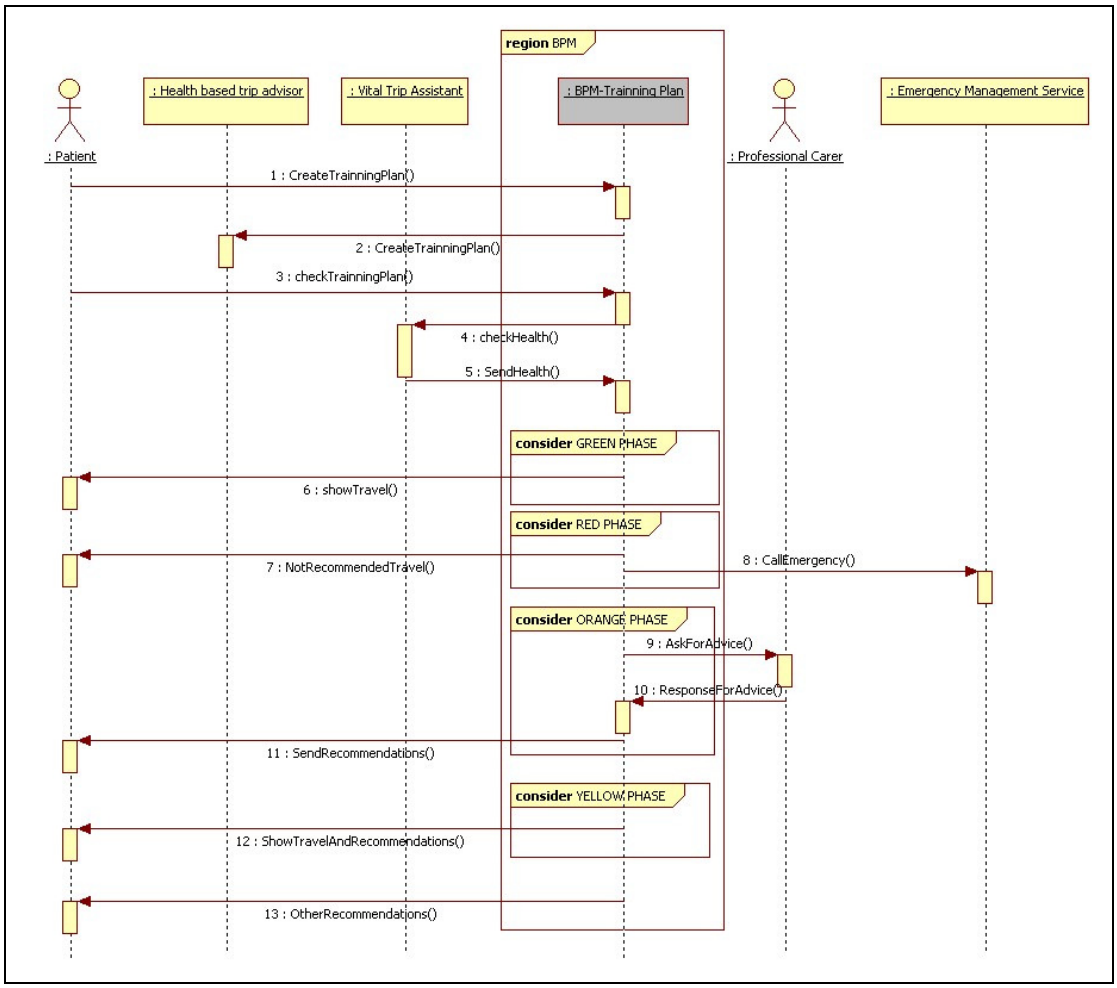


Figure 3-2 Elaborate Trip Plan Sequence Diagram

- Elaborate Nutrition Plan:

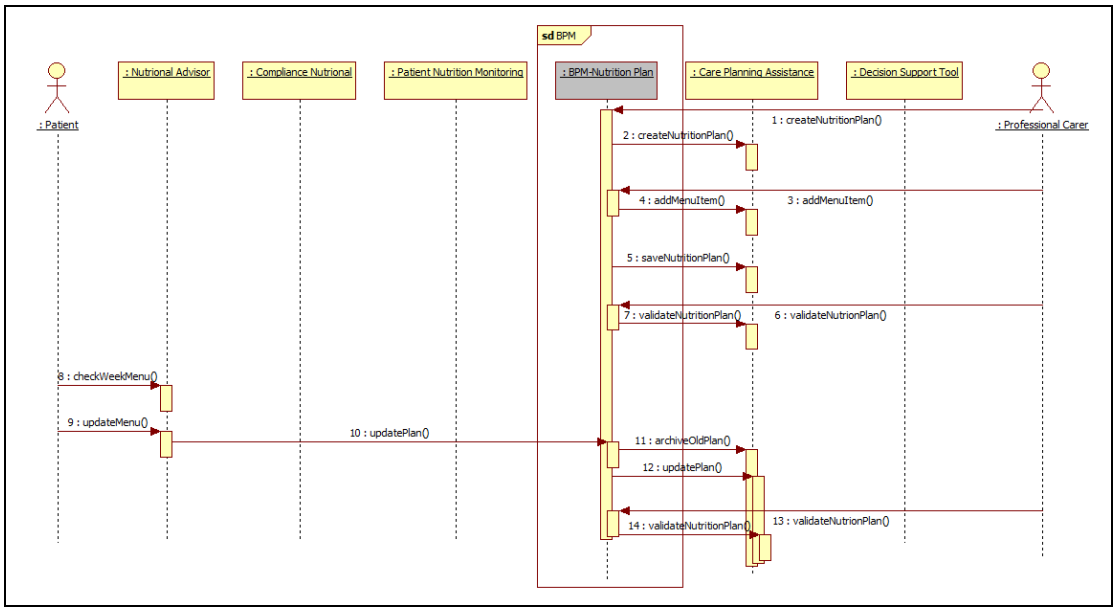


Figure 3-3 Elaborate Nutrition Plan Sequence Diagram

- Elaborate Training Plan:

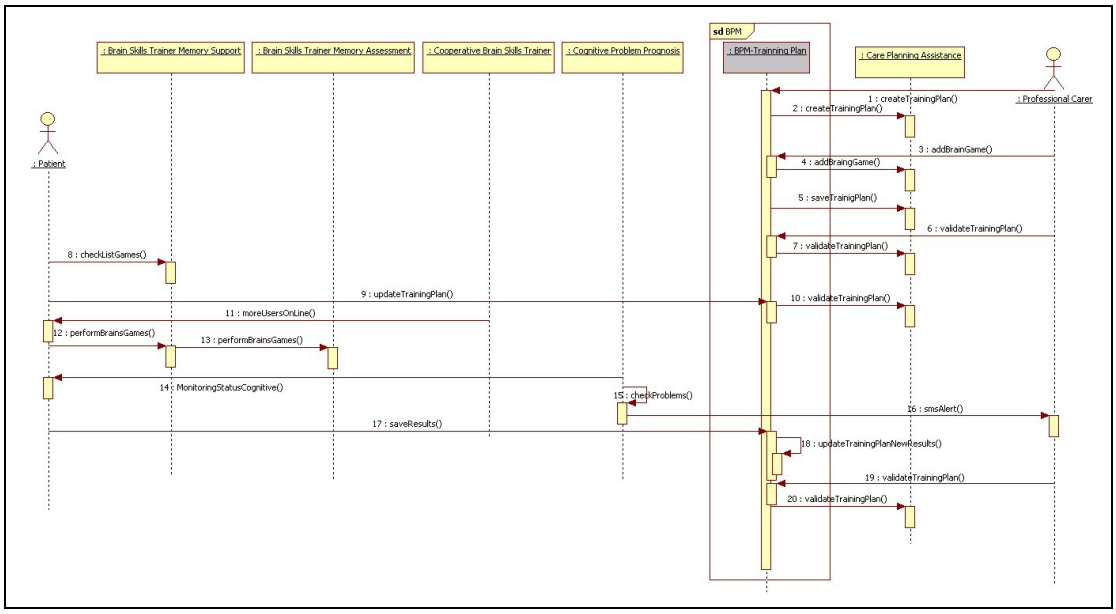


Figure 3-4 Elaborate Training Plan Sequence Diagram

- Elaborate Patient Medication Plan:

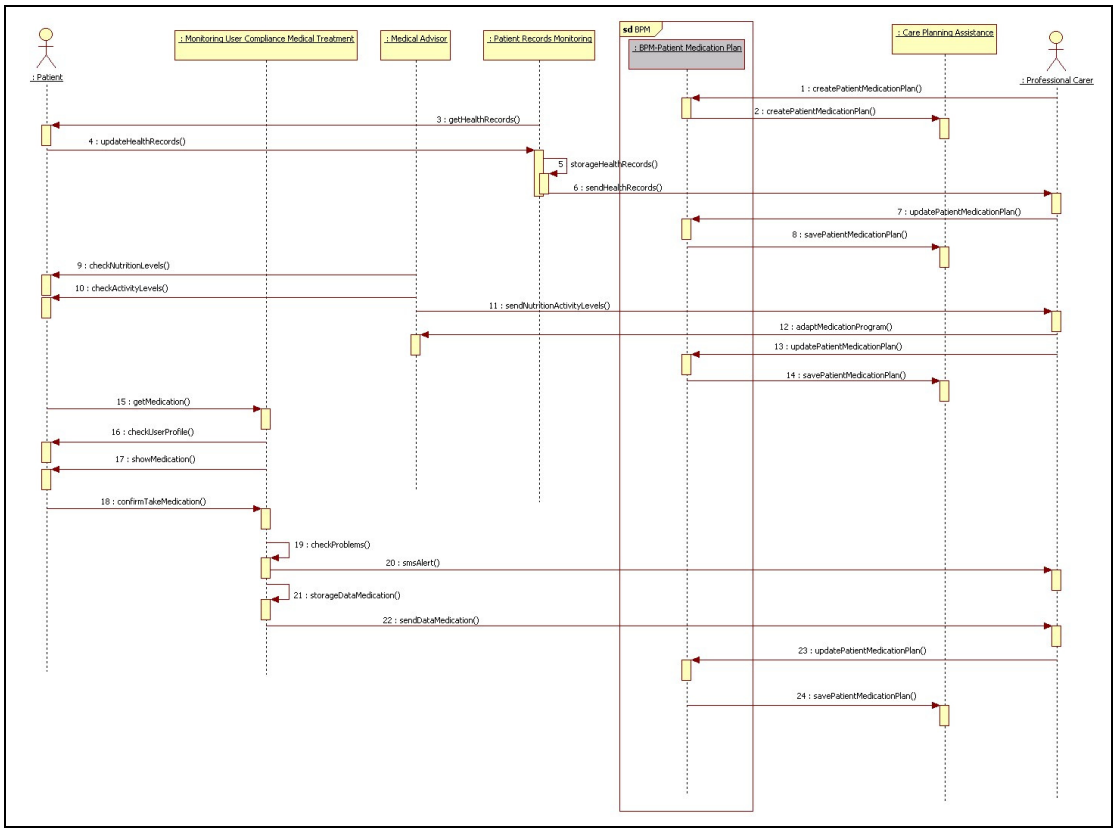


Figure 3-5 Elaborate Patient Medication Plan Sequence Diagram

- Elaborate Activity Plan:

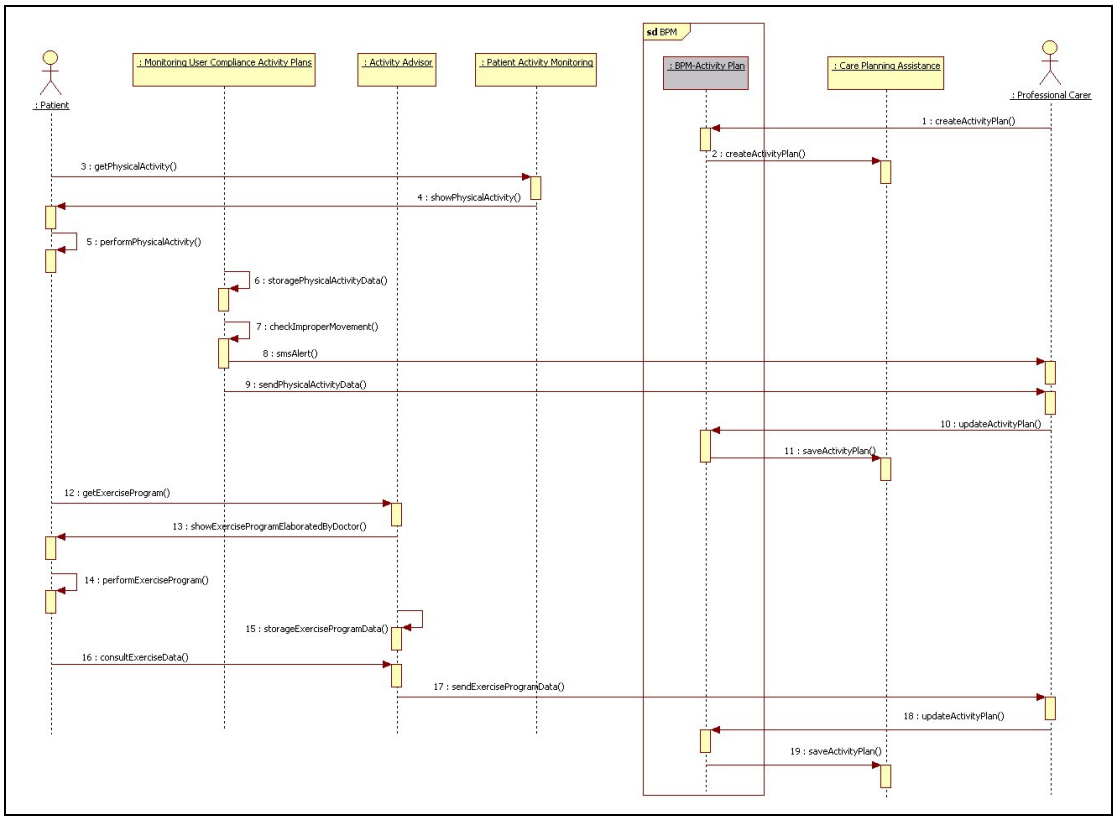


Figure 3-6 Elaborate Activity Plan Sequence Diagram

3.3 BUSINESS USE CASES

The business use cases can also be used to further decompose the domain since they are usually good candidates for services that will be ultimately exposed as Web Services.

In Remote, several use cases, in different functional areas, which are candidates to become Web Services can be identified:

Monitoring Services

- **Vital Signs Monitoring:** Monitoring of physiological signs of the patient on regular basis and per request. The aims of this module (hereafter: Vital-signs-monitoring module) are:
  1. To provide higher system levels, (higher in terms of data processing) in real-time, raw data that reflects the most updated health status of the given patient (hereafter '**Polling mode'**).
  2. Allow REMOTE system higher-levels (such as point of care request, emergency service, etc.) modules (UC's) to initiate a request to the Vital-signs-monitoring module to take measurement of specific parameters (hereafter '**Interrupt-mode'**).
- **User activity recognition and characterization:** Monitoring of user activity and characterization of it according to norms and models (i.e. sleeping, walking, viewing TV). For some of them it requires cooperation

with other UCs, i.e. UC5.1, to know that TV is on and UC5.2 to localize the user.
- **User profile building:** Building the initial, explicit profile of the user (i.e. medication, driving or not,…), also with the help of carer. Ability of the user to view and easily understand and modify the explicit profile.

Personal self-care services
- **Calendar.** The *Personal Calendar* application will be used for scheduling/managing the daily tasks of the elderly (managing nutrition, medication, to-do lists, etc.) under the unobtrusive supervision of carers.
- **Activity Advisor:** Advices to the user on daily or weekly basis on recommended activities, taking into account his/her implicit health profile actual health status previous activities record, nutritional plan and actual nutrition uptake.
- **Health Advisor:** Adaptation by MD of medication program of the patient. The MD is allowed to define a medication plan, and adapt it after taking into account his/her actual nutrition level and activities level.
- **Nutritional Advisor.** Advices to the user on daily or weekly basis on recommended menus and calories uptake, taking into account his/her medication and allergies, activities level and actual health status.
- **Monitoring of user compliance:** Provide feedback to the user and potential warnings to the professional carer.
- **Brain and Skills Trainer.** The aim of this UC is to focus on memory support of Alzheimer or other demented patients. Brain games will help to improve cognitive abilities and possibly to delay Alzheimer's disease onset and/or progress.
- **Social Networking:** Tools to allow and promote user friendly communication of the elderly users with their family members as well as other users with common interests.
- **Trip Advisor (Guarding Angel).** Provides all sorts of pedestrian and/or multimodal trip planning and route guidance. Virtual trip assistant, will monitor the health conditions of the user on the move and provide, when necessary, alert messages and guidance for preventing dangerous events (dehydration, high blood pressure, arrhythmia, etc.). This service aims to detect automatically when the elderly is in abnormal situations when he/she is outside home and help them to exit from this situation by guiding them or requesting for help. The system is able to detect when the user is lost, and knowing the current position it offers alternatives of destination points.

Augmented autonomy services
- **Environmental Control.** Monitoring and control of many devices at home for user comfort, activity monitoring purposes and especially safety and security reasons. Functions automation to protect a patient with cognitive problems to forget open the oven or kitchen (safety) as well as the lights and water boiler (economy). Gradually deploy home automation facilities to substitute functional limitations in performing every day domestic duties.

- **In home user localization:** Localization of the user at home at the level of room (no higher accuracy required) for activity monitoring purposes as well as for safety reasons (i.e. not to close the light by another user while he/she is negotiating stairs).
- **Fall protection:** Detects when the user falls down at home and alerts the emergency services. A communication link can be open with the user to give him/her support to manage the situation while help arrives.
- **Home gateway for services:** Gateway to allow fast, cheap and secure incoming and outcoming communication between user home, medical centre, his/her professional and informal carers, family members and third party services he/she subscribes in.

Professional healthcare services

- **Monitoring and storage of the periodic status report.** The status of the system and of the patient's activities and health should be monitored and stored for a periodic report.
- **Patient nutrition monitoring.** Monitoring and storage of patient nutrition patterns data for use by his/her carer.
- **Cognitive problem prognosis:** MD support in monitoring of patient cognitive status, to help prognoses its deterioration and adapt accordingly the treatment.
- **Decision support tool for patient and treatment administration:** Professional carer support tool that supports the patient state evaluation, problems prognosis and treatment planning, based upon his/her profile of UC1.3, as well as UCs 2.4, 2.5, 2.6, 7.1, 7.2, 7.3 and 7.4 data. Is a dynamic tool, running upon professional carer request at any moment. It is not deciding itself; it is simply for carer support.
- **Treatment planning assistant:** Professional carer support tool to establish new or updated treatment procedures his/her patients.
- **MD-patient dialogue support:** Tool to support direct written and verbal interaction/discussion between carer and patient. For simplicity the tool will be referred to as CPDS tool (Carer – patient dialogue support tool).

Informal tele-assistance services

- **Informal assistant-patient dialogue support:** Tool to support remote contact between informal assistant, formal carer and patient, including data transmission.
- **Care planning assistant:** Tool for the caregiver, to manage the information collected about the patient through different ways.
- **Automated alerts and periodic health status reporting:** Service to provide monitoring and storage of patient health record for use by the informal caregiver.

Medical Contact Centre

- **Medical Centre Administration tool:** The Medical Contact Centre (interface) allows the coordination of patients living at home, patients' caregivers (living close), primary care physician at the rural areas, doctors at the specialized clinic and home care stuff/agencies. It also

allows user (patients, doctors, domotics system, etc.) and service administration and archiving.

- **Emergency Management Service:** Visualisation and management of automatically driven or patient driven emergency calls and alerts.

Other

- **System Administration:** The Administration of each system and especially those of REMOTE is of high importance as it is going to administer the whole system and to provide rights of access to the user, etc. Is of critical importance, as it maintains the viability of such a demanding system.
- **Patient subscription:** Tool and procedure for patient subscription/ unsubscription.
- **Professional carer subscription:** Tool and procedure for professional carer subscription/ unsubscription.
- **Informal carer subscription.** Tool and procedure for informal carer subscription/ unsubscription
- **Statistical and help tools:** Creation of database for centrally allocation of relevant data of all partners so as to create several different links for the statistical inference and manipulation of future implementation. In addition, it is important to note that the data will be used for anonymous use for research purpose.

The UML of those Business Uses Cases can be seen in figure 3-7:

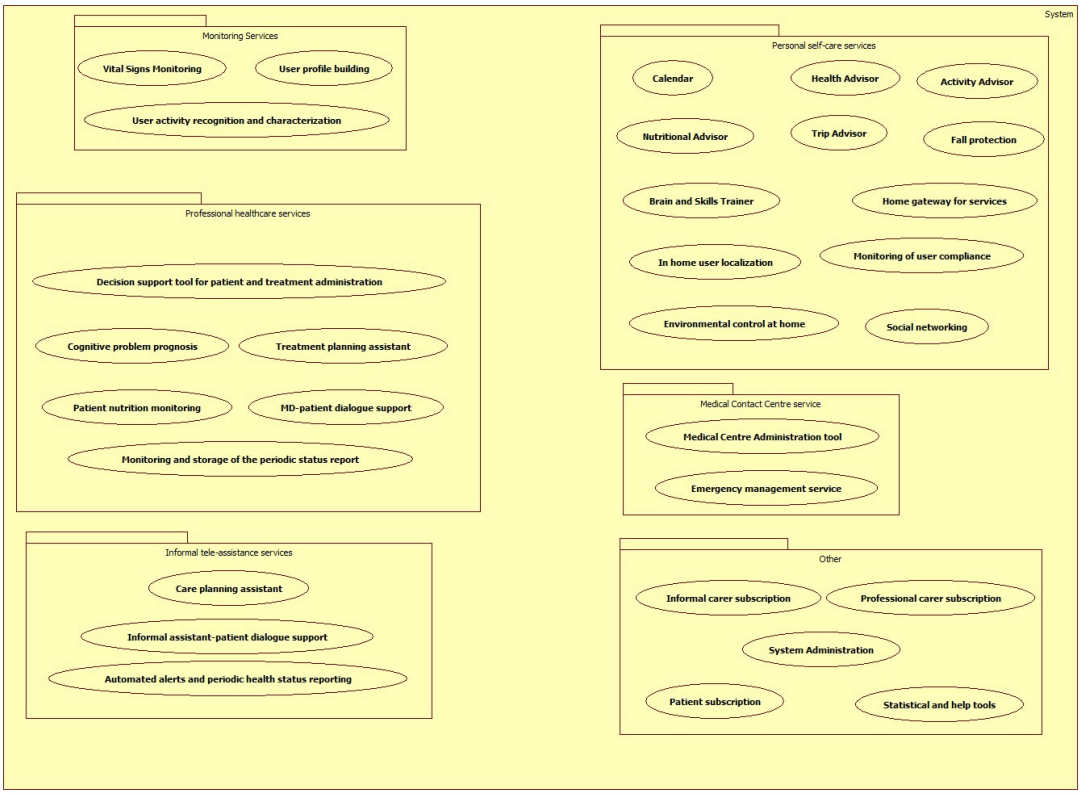


Figure 3-7 UML Business Uses Cases

4 GOAL SERVICE MODELING

In SOMA methodology, another technique for services identification is through the goal service modelling. Service identification implies the discovery of business aligned services for the entire organization. The business level use cases identified in domain decomposition are good candidates for services. In this step we create a goal-service model through business process identification and business goals identification.

4.1 BUSINESS GOALS

"A business goal is a requirement that must be satisfied by the business. Business goals describe the desired value of a particular measure at some future point in time and can therefore be used to plan and manage the activities of the business" [5].

We use the business goals to make sure that we clearly understand what steps we have to take to achieve the business strategy. Here are some businesses Goals:

1. **REMOTE reduce health spending**: Through a lens technology that provides remote will be the reduction of health expenditure, largely due to medical interventions to be made through REMOTE, reducing the number of visits to health workers.
    a. **Reduce waiting in health care**: REMOTE reduce the number of patients attending their clinics for simple routine examinations. Since their homes may be monitored, easing the centres for more specific or urgent care.
2. **Improving the quality of life of elderly and physically disabled**: The elderly may be controlled from their own homes, avoiding travel to medical centres and reducing waiting times between patients. This impact positively on the welfare of this range of population, health and comfort.
    a. **Access to rural areas**: With REMOTE may cover the access to medicine and medical personnel in rural areas, decentralized and difficult to access. Could be addressed and monitored patients and patients in their own homes through technology. Improving the standard of living of the population and reaching any place that is inaccessible.
    b. **Patients monitored 24 / 7**: REMOTE gives patients the security of being constantly watched, even in your own home. With the help of sound technology and communications implemented, the doctor may state the condition of his patient at any time of day, earning the patient quality of life and hospital care.
    c. **Reduce response times in emergency patients (Thanks to constant monitoring)**: REMOTE can provide the patient a rapid

response in case of emergency. With constant monitoring of the patient, REMOTE can sound an alarm if the patient suffers a medical absence, indicating the nearest health centre's need for medical care without the patient's request.

4.1.1 Metrics

Let us define the metric for the points mentioned above:

1. **REMOTE reduce health spending**: % money / time**.**
    a. **Reduce waiting in health care**: Decrease of % patients / time
2. **Improving the quality of life of elderly and physically disabled**:
    a. **Access to rural areas**: Increase the number of % patients / time
    b. **Patients monitored 24** / **7**: Decrease of % patients / time
    c. **Reduce response times in emergency patients (Thanks to constant monitoring)**: Decrease % time of emergency / time.

An example of% money / time would decrease spending in 6 months.

5 EXISTING ASSET ANALYSIS

According to the table 4 Services/UCs in D6.2 “REMOTE Integration Platform Architecture” [3] in this chapter it will be presented similar existing custom applications, packaged applications and industry models to determine what can be leveraged to realize service functionality. The existing services will be presented in the following groups as it was determined in the above mentioned table 4 D6.2 [3]:

- Monitoring services
- Personal self-care services
- Augmented autonomy services

5.1 MONITORING SERVICES

In this area of services will be presented advanced technological solutions which are implemented in such way that preserve the autonomy and mobility of the end-user without interrupting the daily living habits and activities but continuously and unobtrusively monitor the most important, according to the user needs, vital parameters condition.

5.1.1 WEALTHY

WEALTHY system [6] is developed as the integration of several function modules. The main functions of the modules are namely sensing, conditioning, pre-processing, data transmission and remote monitoring.

Sensing module consists of the garment connected with the Portable Patient Unit (PPU), where local processing as well as communication with the network is performed.

The PPU is small and lightweight, only 145g, it is easy to use, with two LEDs and a buzzer for user warning purpose and a button to let a manual trigger of alarm, data transmission is done over GPRS link. The device is powered by a Li-Ion battery autonomy up to 4 hours with real time streaming of all signals over GPRS ECG signals are sampled on the PPU at 250Hz where a local processing is applied in order to extract parameters with the highest sampling rate, in a way to compute ECG parameters, such as heart rate (HR) value and QRS duration.

The complete list of the signals interfaced by the PPU is given below:

- Six ECG electrodes configurable in Einthoven configuration (lead I, II and III) and Wilson configuration (V2 andV5). Only one lead is transmitted at a time (for GPRS bandwidth limitation reasons). The ECG lead to be transmitted can be selected remotely by the monitoring centre.
- Respiration by impedance measurement
- Up to four I2C skin temperature sensors (monolithic circuits)

- One 3D accelerometer (integrated in the unit)
- Four piezo-resistive strain sensors
- SpO2 (oximetry) from a commercial device (NONIN) (serial interface and power supply provided).

In order to offer full mobility to the patient or the user, acquired signals are wirelessly transmitted from the PPU to the remote Monitoring System. The communication is based on TCP/IP that is the standard protocol for GPRS communication. All signals are sent in quasi real-time to the Remote Monitoring Centre.

A different version uses Bluetooth connection for short range transmission.

In order to increase the comfort of the garment a design study has been carried out and for each model a prototype has been realized and tested. The final garment is easier to wear, comfortable from the thermal aspect as well as from the ergonomics aspect, washable and good enough from the look and feel perspective.

In the next figure is shown the old and the new version of Wealthy garment.

Figure 5-1 Wealthy System, old version on the left, new version on the right.

5.1.2 PROETEX (inner Garment)

The inner garment prototype of the PROETEX system [7] is to improve the safety and efficiency of the end users (emergency workers) by empowering them with wearable sensing and transmission systems that monitor their health, during such risky situations.

The health condition parameters that the Proetex inner garment system monitors are:

- Heart rate
- Breathing
- Core Temperature
- Pulse Oximetry (SpO2)
- Dehydratation

The Proetex inner garment system is the presented in the following figure:

Figure 5-2 Proetex prototype of the inner garment

5.2 PERSONAL SELF-CARE SERVICES

5.2.1 Nutrition software review 2011

The following table show a comparative of the nutrition software that you can find in the market:

	DietMaster 2100	Nutrinote	DietPower	Kathleen's Diet Planner	DietOrganizer	FitDay	BeNutriFit	Diet Pro	Mealformation	Do-It

	DietMaster 2100	Nutrinote	DietPower	Kathleen's Diet Planner	DietOrganizer	FitDay	BeNutriFit	Diet Pro	Mealformation	Do-It
General Nutrition Features										
Personal Home Use	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Supports Commercial Diets	✓	✓		✓			✓			✓
Goal Setting	✓	✓	✓	✓	✓	✓	✓	✓		✓
Track Multiple Individuals	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Weight History Tracking	✓	✓	✓	✓	✓	✓	✓	✓		✓
Portion Editor	✓	✓	✓	✓	✓		✓	✓	✓	
Nutrient History	✓	✓	✓	✓	✓	✓	✓	✓		✓
Medical Factor Tracking	✓	✓	✓	✓						
Calculators	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Body Measurements	✓	✓	✓	✓	✓	✓		✓		
Body Composition Screen	✓	✓		✓						
Calendar	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Food Management										
Food Log	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Brand Name & Restaurant Foods Included	✓		✓	✓	✓	✓	✓	✓	✓	✓
Add New Food Items	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Favourites	✓	✓	✓	✓	✓	✓				
Calorie Bank		✓	✓	✓	✓		✓	✓		
Food Recommendations to Meet Daily Goals		✓	✓							
Nutrient Summary	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Recipes & Meals										
Create A Recipe	✓	✓	✓	✓			✓	✓	✓	✓
Create A Meal Plan	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Exercise Integration										
Exercise Log	✓		✓	✓	✓	✓	✓	✓		✓
Preset Exercises with Calorie Burn Rate	✓	✓	✓	✓	✓	✓	✓	✓		
Reporting for Motivation										
Reports	✓	✓	✓	✓	✓	✓	✓	✓		✓
Graphs	✓	✓	✓	✓	✓	✓	✓	✓		✓
Technical Help/Support										
Email	✓	✓	✓	✓	✓	✓	✓		✓	✓
Toll Free Phone Support	✓		✓							✓
Phone Support	✓		✓	✓					✓	✓
FAQ			✓		✓	✓		✓		
User Forum			✓	✓				✓		
Supported Configuration										
Vista	✓	✓	✓	✓	✓					
XP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2000	✓		✓	✓		✓	✓	✓	✓	✓
Mac	✓									

Table 5-1. Nutrition Software Review

5.2.2 Fitness software review 2011

The following table show a comparative of the fitness software that you can find in the market:

	VidaOne Diet & Fitness	Weightmania	ProTrack	Open Fitness	Fitness Assistant	BodyTrans	Weight-By-Date
Features							
Personal Use	✓	✓	✓	✓	✓	✓	✓
Commercial Use	✓	✓	✓	✓		✓	
Tracks Major Fitness Indicators	✓	✓	✓	✓	✓	✓	✓
Tracks Medical Details	✓	✓					
Journal	✓	✓	✓	✓	✓	✓	✓
Calendar	✓	✓	✓	✓	✓	✓	✓
Aerobics/Sports Module	✓	✓		✓	✓	✓	
Strength Training Module	✓	✓	✓	✓	✓	✓	
Summary View	✓	✓	✓		✓		
Fitness							
Fitness Routine	✓	✓	✓	✓	✓	✓	✓
Add a Routine	✓	✓	✓	✓	✓	✓	✓
Share a Routine		✓		✓	✓	✓	
Goal Setting & Tracking	✓	✓	✓	✓	✓	✓	✓
Add/Delete Exercises	✓	✓	✓	✓	✓	✓	✓
Workout Planner	✓	✓	✓	✓	✓	✓	✓
Training Analysis	✓	✓	✓	✓	✓	✓	✓
Nutrition							
Day's Nutrition Facts	✓	✓	✓	✓	✓	✓	✓
RDA Calcultor	✓	✓	✓			✓	
Customizable # Meals/Snacks		✓			✓		✓
Food Library Filter	✓						✓
Create Shopping List		✓					
Share Meal Plans		✓		✓	✓		
Other Features							
Export/Import to Handheld Device	✓	✓	✓	✓	✓	✓	✓
Reports	✓	✓	✓	✓	✓	✓	✓
Graphs	✓	✓	✓	✓	✓	✓	✓
Technical Support							
Email	✓	✓	✓	✓	✓	✓	✓
FAQ	✓	✓		✓	✓	✓	✓
Toll Free Phone Support							
Phone Support							✓
User Forum				✓	✓		
Supported Configurations							
Vista	✓	✓	✓	✓	✓	✓	✓

	VidaOne Diet & Fitness	Weightmania	ProTrack	Open Fitness	Fitness Assistant	BodyTrans	Weight-By-Date
7	✓	✓			✓		✓
XP	✓	✓	✓	✓	✓	✓	✓
Mac		✓		✓			

Table 5-2. Fitness Software Review

5.2.3 Brain training games Brain Age^2

Exercise is the key to good health, both for body and mind - and Brain Age^2 is able to provide simple mental exercises. Inspired by the work of Japanese neuroscientist Dr. Ryuta Kawashima, the Brain Age games feature activities designed to help stimulate the brain and give it the workout it needs like solving simple math problems, counting currency, drawing pictures on the Nintendo DS touch screen, and unscrambling letters.

In order to install and play the Brain Age application the Nintendo DS handheld game system in needed. The DS and Brain Age are sold separately.

Figure 5-3 Nintendo DS

The touch screen let users write their answers with a Stylus pen, as if they were writing on paper or using a Personal Digital Assistant or "PDA." And, with Nintendo DS's voice recognition technology, Brain Age can identify particular words during certain activities.

The first time the user play Brain Age, take a series of tests and get a score that determines the DS brain age. Brain Age tracks the user's progression with easy-to-read line charts. By using Brain Age, the better user get at the exercises and the lower the DS Brain Age will become.

5.3 AUGMENTED AUTONOMY SERVICES

5.3.1 The Smartest House of the Netherlands, Eindhoven, Netherlands

In 2001, Smart Homes built the so-called Smartest House of the Netherlands. In the first place it is a showcase for demonstration of state of the art smart home technology, but it also serves as test house for various new technical developments.

One can distinguish 5 levels of home automation, where, within the vision of Smart Homes, we only speak of real home automation from the third level onwards.
Homes which contain intelligent objects – homes contain single, stand-alone applications and objects which function in an intelligent manner.

- Homes which contain intelligent, communicating objects – homes contain appliances and objects which function intelligently in their own right and which also exchange information between one another to increase functionality.
- Connected homes – homes have internal and external networks, allowing interactive and remote control of systems, as well as access to services and information, both within and beyond the home.
- Learning homes – patterns of activity in the homes are recorded and the accumulated data are used to anticipate users' needs and to control the technology accordingly.
- Attentive homes – the activity and location of people and objects within the homes are constantly registered, and this information is used to control technology in anticipation of the occupants' needs.

The house is an average size bungalow type, which the average citizen would recognize as a "wanted to have" house. In the house 4 main issues are addressed: (1) accessibility for anybody, (2) physical adaptability according to changing needs, (3) sustainability, energy saving and sustainable energy and (4) smart solutions and services for anybody in entertainment, communication, tele-working, tele-learning, health and care.

Figure 5-4 The smartest house in Netherlands

The test house is open for workshops and tours through weekdays and open for experiencing by families who like to spend a weekend in it. From 2001 up to now, Smart Homes has moved the demo house to several places across the country. From April 2008 to mid 2012, the house will be positioned in

Eindhoven and will be available for testing, evaluation, validation and demonstration in various research projects.

5.4 CONCLUSION

This section show a set of services/applications you can find in the market and how they fit with the REMOTE project challenges.

As it is described, there are several assets that are in line with the objectives of the project, but these products are design to satisfy a wide range of population from a business perspective.

On the other side, REMOTE assets are design to satisfy a specific kind of people and following the integration concept, it means the main objective of the developers is to design a services that could be integrated in a common framework.

6 SERVICE IDENTIFICATION AND SPECIFICATION

In this part of the document are described the Services Candidates in Remote.

The list of the Services is divided depending on their functionalities:

- Monitoring
- Personal self-care
- Augmented autonomy
- Professional healthcare
- Informal tele-assistance
- Medical Contact Centre
- COF (Common Ontological Framework)

6.1 MONITORING

6.1.1 Functionalities

Services that monitor user's health state, activities and surrounding conditions. These services are used by the self-care applications

6.1.2 Services Candidates

Services Candidates List:

- Vital Sings Monitoring
- User profile building

6.1.3 Vital Sings Monitoring

6.1.3.1 Service Description

Name	Vital Sings Monitoring
Description	Monitoring of physiological signs of the patient on regular basis and per request
Actors	Primary actors: Patient Secondary actors: Formal care-givers (both inpatient and outpatient), Public / private Social security service providers and insurance companies, Service Centre, Health care and emergency support service providers
Related UCs	U.C 1.1
Inputs	Patient identification

Outputs	Sensor Data

Table 6-1. Vital signs monitoring service description

6.1.3.2 Data Storage

This service will use a date base which will contain the thresholds for all kinds of vital signs being monitored for every patient, as well as the measurements taken by the patients.

Figure 6.1 shows the data base schema.

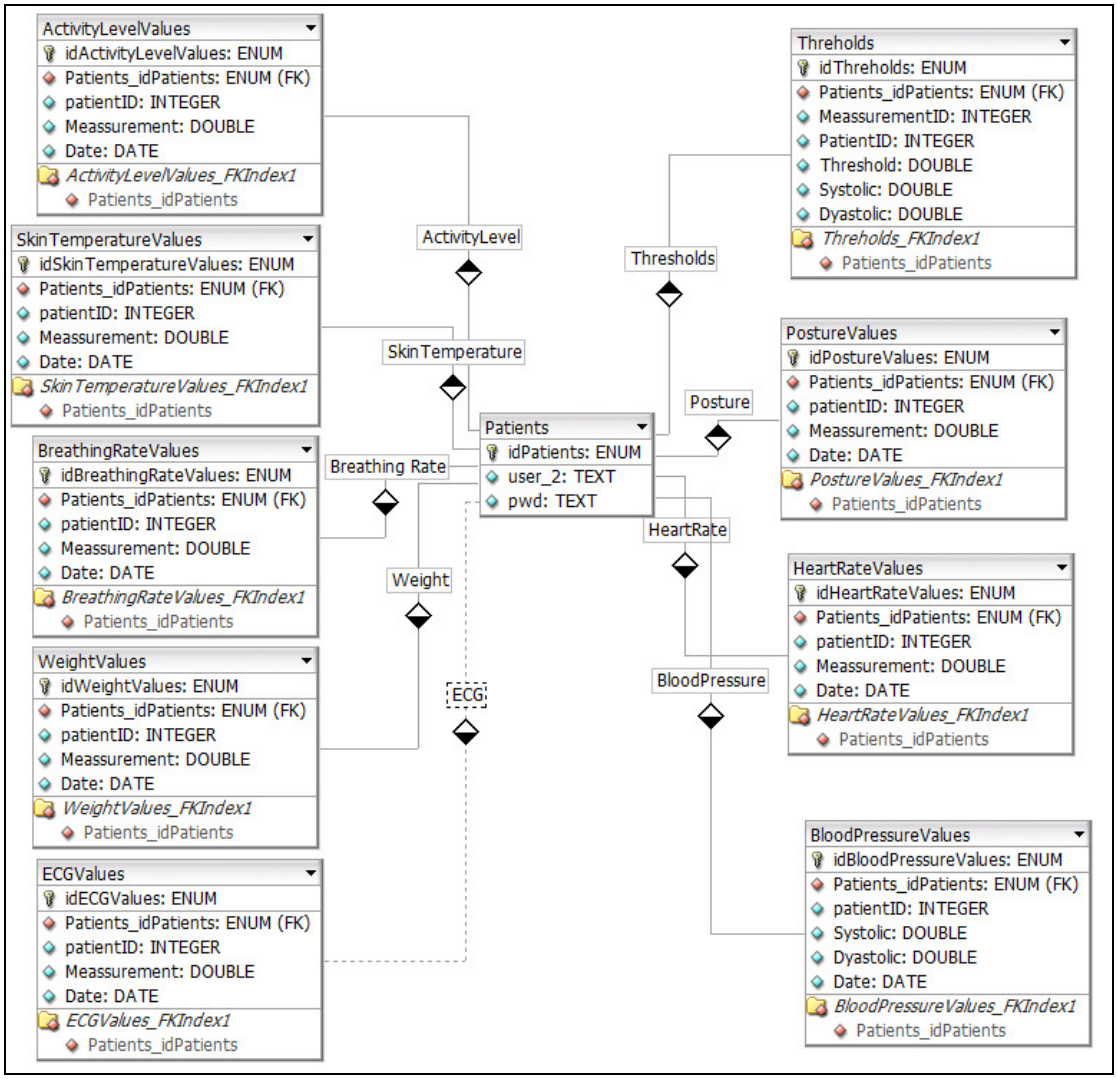


Figure 6-1. Schema of Vital Signs DB

6.1.3.3 Classes

There are only two new classes created to work with the Vital Signs service:

- **measurements:** This class is used to set and return the values of each measure. Is compound by:
  - *String Date*
  - *Double Value*
- **bloodPresMeasurements:** This class is used to set and return the values of the blood pressure measure. Is compound by
  - *String Date*
  - *Double Systolic*
  - *Double Diastolic*

6.1.3.4 Operations

Name	Description	Inputs	Outputs
`getActivityLevelThresholdsValueByPatientId`	Returns activity levels threshold for a given patient	Int patientID	Double Threshold
`getActivityLevelValuesByPatientIdDates`	Returns activity levels for a given patient between the dates	Int patientID String startdate String enddate	List Double values
`getAllThresholdsValuesByPatientId`	Returns all threshold values for a patient	Int patientID	List Double thresholds
`getBloodPressureValuesByPatientIdDates`	Returns blood pressure values for a given patient between the dates	Int patientID String startdate String enddate	List Double values
`getBloodPressureThresholdsValueByPatientId`	Returns the blood pressure thresholds for a patient	Int patientID	List Double Values
`getBreathingRateThresholdsValueByPatientId`	Returns the breathing rate thresholds for a patient	Int patientID String startdate String enddate	List Double Values
`getBreathingRateValuesByPatientIdDates`	Returns breathing rate values for a given patient between the dates	Int patientID String startdate	List Double values

		String enddate	
getECGThresholdsValueByPatientId	Returns the ECG thresholds for a patient		
getECGValuesByPatientIdDates	Returns ECG values for a given patient between the dates	Int patientID String startdate String enddate	List Double values
getHeartRateValuesByPatientIdDates	Returns the heart rate thresholds for a patient between the dates	Int patientID String startdate String enddate	List Double values
getHeartRateThresholdsValueByPatientId	Returns heart rate values for a given patient	Int patientID	Double value
getLastActivityLevelValuesByPatientId	Returns last activity level value of a patient	Int patientID	Double value
getLastBloodPressureValuesByPatientId	Returns last blood pressure value of a patient	Int patientID	Double value
getLastECGValuesByPatientId	Returns last ECG value of a patient	Int patientID	Double value
getLastBreathingRateValuesByPatientId	Returns last breathing rate value of a patient	Int patientID	Double value
getLastHeartRateValuesByPatientId	Returns last heart rate value of a patient	Int patientID	Double value
getLastPostureValuesByPatientId	Returns last posture value of a patient	Int patientID	Double value
getLastSkinTemperatureValuesByPatientId	Returns last skin temperature value of a patient	Int patientID	Double value
getLastWeightValuesByPatientId	Returns last weight value of a patient	Int patientID	Double value
getPostureThresholdsValueByPatientId	Returns the posture thresholds for a patient	Int patientID	Double value
getPostureValuesByPatientIdDates	Returns posture values for a given patient between the dates	Int patientID String startdate String enddate	List Double values
getSkinTemperatureThresholdsValueByPatientId	Returns the skin temperature thresholds for a patient		
getSkinTemperature	Returns skin temperature	Int	List

ValuesByPatientIdDates	values for a given patient between the dates	patientID String startdate String enddate	Double values
getWeightThresholdsValueByPatientId	Returns the weight thresholds for a patient		
getWeightValuesByPatientIdDates	Returns weight values for a given patient between the dates	Int patientID String startdate String enddate	List Double values
setActivityLevelByPatientId	Sets current Activity Level value for a patient	Int patientID Double Meassurement	True/false
setActivityLevelThresholdsValueByPatientId	Sets Activity Level threshold for a patient	Int patientID Double Threshold	True/false
setAllThresholdsValuesByPatientId	Sets all threshold values for a patient	Int patientID List Double Threshold	True/false
setBloodPressureThresholdsValueByPatientId	Sets current blood pressure threshold for a patient	Sets current Activity Level value for a patient Int patientID Double Systolic Double Dyastolic	True/false
setBloodPressureValueByPatientId	Sets current blood pressure value for a patient	Int patientID Double Systolic Double Dyastolic	True/false
setBreathingRateThresholdsValueByPatientId	Sets breathing rate threshold for a patient	Int patientID Double True/false Threshold	True/false
setBreathingRateValuesByPatientId	Sets current breathing	Int	True/false

	rate value for a patient	patientID Double Meassurement	
setECGThresholdsValueByPatientId	Sets ECG threshold for a patient	Int patientID Double Threshold	True/false
setECGValueByPatientId	Sets current ECG value for a patient	Int patientID Double Meassurement	True/false
setHeartRateThresholdsValueByPatientId	Sets heart rate threshold for a patient	Int patientID Double Threshold	True/false
setHeartRateValueByPatientId	Sets current heart rate value for a patient	Int patientID Double Meassurement	True/false
setPostureThresholdsValueByPatientId	Sets posture threshold for a patient	Int patientID Double Threshold	True/false
setPostureValueByPatientId	Sets posture ECG value for a patient	Int patientID Double Meassurement	True/false
setSkinTemperatureThresholdsValueByPatientId	Sets skin temperature threshold for a patient	Int patientID Double Threshold	True/false
setSkinTemperatureValueByPatientId	Sets current skin temperature value for a patient	Int patientID Double Meassurement	True/false
setWeightThresholdsValueByPatientId	Sets weight threshold for a patient	Int patientID Double Threshold	True/false
setWeightValueByPatientId	Sets current weight value for a patient	Int patientID Double Meassurement	True/false

setWeightThresholdsValueByPatientId	Sets weight threshold for a patient	Int patientID Double Threshold	True/false
setWeightValueByPatientId	Sets current weight value for a patient	Int patientID Double Meassurement	True/false
getHydrationThresholdsValueByPatientKey	Returns the hydration thresholds for a patient		
getHydrationValuesByPatientKeyDates	Returns hydration values for a given patient between the dates	Int patientID String startdate String enddate	List Double values

Table 6-2. Vital Sings Monitoring operations

6.1.4 Service User profile

6.1.4.1 Service Description

Name	User profile
Description	The User Profile module is responsible for the storage of different properties / parameters an application wants to store. These properties contain user and application specific information. In addition, all other common user related data are stored in the user profile (e.g. name, username, age, etc.). The way the user profile is realized in REMOTE is very similar to the one used in OASIS EU project.
Actors	Primary actors: Patient
Related UCs	• UC1.2 • UC1.3.1 • UC1.3.2 • UC1.3.3 • UC2.1 • UC2.3 • UC2.4 • UC3.1

Inputs	User name
Outputs	User Profile

Table 6-3. User profile building service description

6.1.4.2 Data Storage

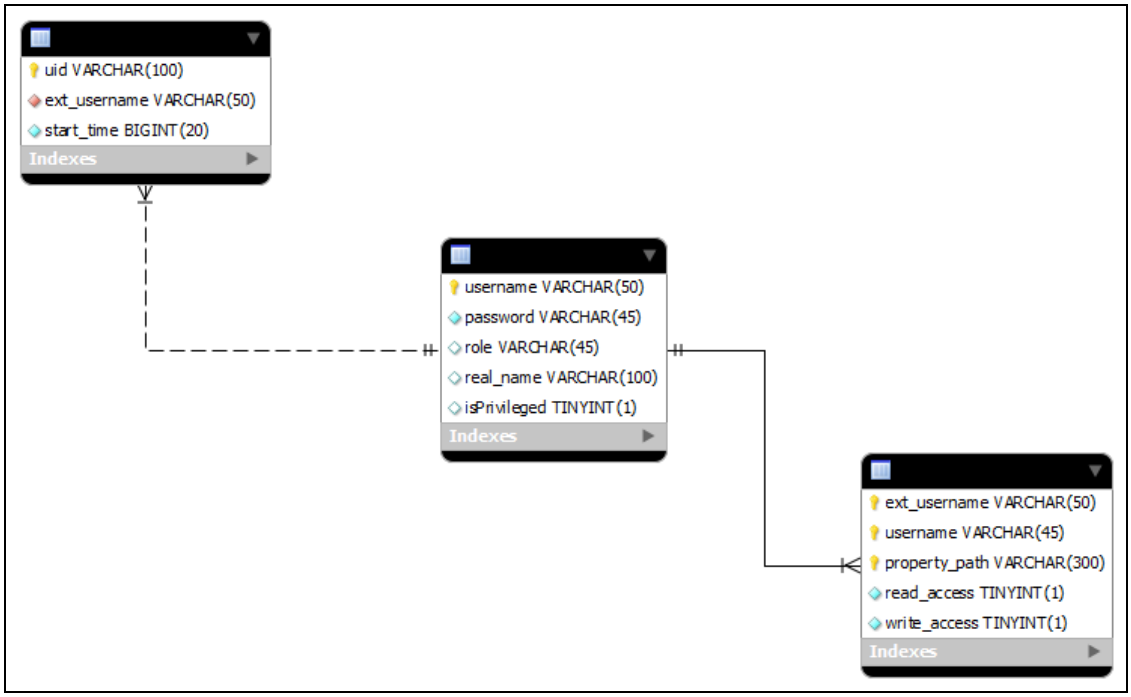


Figure 6-2: User Profile Web Service Database Schema

6.1.4.3 Classes

BooleanReply: This class is used to describe a reply which contains:

- Boolean: a Boolean output of the operation
- String: reply output information

StringReply: This class is used to describe a reply which contains:

- String: a string output of the operation
- String: reply output information

StringArrayReply: This class is used to describe a reply that contains:

- String[]: an array of strings as output of the operation
- String: replyoutput information

6.1.4.4 Operations

An external user, in order to have access to the User Profile of one or more REMOTE users, needs to be a registered user. Being a registered user doesn't

mean that he/she has access to the ***whole*** Profile of ***all*** REMOTE users: During registration process, the external user has to declare to which REMOTE users he/she wants to have access, and which part of their profile (for example, external user *eu1* has access to the medical profile of REMOTE user "Nikos").

There are six methods available:

Name	**Description**	**Inputs**	**Outputs**
authenicateEntity	This is the first operation you to call, in order to start the session with the Profile Service. The StringReply objects contains two String values: a reply message and a token (String).	String entityName, String password	StringReply
getProperty	Call this operation to get the value of the property "*propertyName*" of the REMOTE user "*username*". The "*token*" is the string value you have from authenticateEntity() method call.	String token, String username, String propertyName	StringArrayReply
getSingleProperty	Call this operation to get the first value of the property "*propertyName*" (or when you know that its is a single-value property) of the REMOTE user "*username*". The "*token*" is the string value you have from authenticateEntity() method call.	String token, String username, String propertyName	StringReply
getProperties	Call this operation to get the values of more than one property at once.	String token, String username, String[] propertyNam	StringArrayReply[]

		es	
setSingleProperty	Call this operation to set the value of the property "*propertyName*" of the REMOTE user "*username*" to "*value*". The "*token*" is the string value you have from authenticateEntity() method call. Use this methods operation instead of the *setProperty()* when you know that the property holds just one value (single-value property).	String token, String username, String propertyName, String value	BooleanReply
gainAccessForUser	Call this operation to gain access to the profile of the REMOTE user "*username*". The "*token*" is the string value you have from authenticateEntity() method call. The "profileDomain" is the part of the user's profile that you are interested for. For example, if you want to have access to the whole profile of the user, then profileDomain = "/", if you want to have access only to the common profile, then profileDomain="/common" etc. The "*readAccess*" and "*writeAccess*" input values, specify if you want to have read access and/or write access to the profile, respectively.	String token, String username, String profileDomain, boolean readAccess, boolean writeAccess	BooleanReply

Table 6-4. User Profile buildings operations

6.2 PERSONAL SELF-CARE

6.2.1 Functionalities

Services to foster the independent living of elderly considering their specific needs regarding physical isolation and specific chronic conditions. In these services are considered those related with health, calendar, social activities and trips.

6.2.2 Services Candidates

Services Candidates List:

- Health Advisor
- Nutritional Advisor
- Activity Advisor
- Trip Advisor
- Personal Calendar
- Brain and Skills Trainer

6.2.3 Health Advisor Service

6.2.3.1 Service Description

Name	Health Advisor
Functional Area	Personal Self Care
Description	Adaptation by MD of medication program of the patient. The MD is allowed to define a medication plan, and adapt it after taking into account his/her actual nutrition level (from UC2.6) and activities level (from UC1.2).
Actors	Primary actors: Healthy Elderly, Elderly with Stroke, Elderly with Alzheimer, Elderly with Asthma, Elderly with Hypertension and Elderly with Diabetes. Secondary actors: Service Centre, Health care and emergency support service providers, Home automation service providers and Telematic service providers
Related UCs	UC2.2 Input: UC1.2, UC2.6, UC 2.1
Inputs	UC1.2, UC2.6, UC 2.1
Outputs	Medication program.

Table 6-5. Health Advisor service description

6.2.3.2 Data Storage

The data base of this service will store information about which medicaments each patient is taking and when he's suppose to take them. Also, it stores if the patient has taken the medication when the patient marks it in the calendar app.

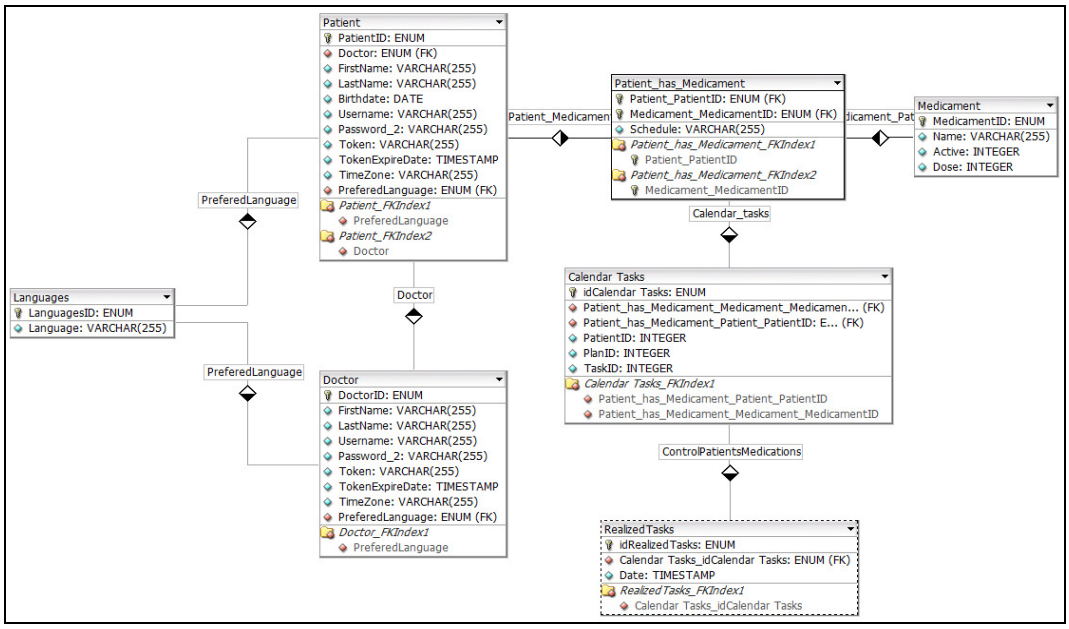


Figure 6-3. Schema of the medical DB

6.2.3.3 Classes

In order to be able to work with the data stored in the data base, several personalized classes have been created for this service.

- **Doctor:** this class is used to store Doctor's information, such as name, Lastname, user, pwd, etc.
  - Int DoctorID
  - String firstName
  - String lastName
  - String preferedLanguage
  - String token
  - String username
  - String password
- **Patient**, which is used to get information from patients, and it's compound by.
  - int PatientID
  - String FirstName
  - String LastName
  - Date BirthDate
  - String Username

  - String Password
  - String Token
  - String TokenExpirationDate
  - String TimeZone
  - String PreferedLanguage
  - int DoctorID
- **Medicament,** which stores all information about a medicament
  - int MedicamentID
  - String Name
  - String Active
  - String Dose
- **MedicationPlan,** which stores all information about a medicament assigned to a patient
  - int MedicationPlanID;
  - int PatientID;
  - Medicament medicament;
  - int Frequency;
  - String[] Days;
  - String[] schedule;
  - Date DueDate;
  - String Quantity
- **MedicalTask,** is used to store the Calendar tasks in the data base
  - int ID
  - int userID
  - int planID
  - int medicamentID
  - String taskID
- **Control,** which stores the information about taken medications in order to follow thw patient's treatment
  - int controlID
  - GregorianCalendar date
  - int taskID
  - int medicament

6.2.3.4 Operations

There are any operations related to the client side of the Health advisor, since the medications are shown as tasks in the Calendar, so the Calendar operations are the ones used to create and erase tasks.

6.2.4 Nutritional Advisor

6.2.4.1 Service Description

Name	Nutritional Advisor
Description	Advices to the user on daily or weekly basis on recommended menus and calories uptake, taking into account

	his/her medication and allergies, activity levels and current health status
Actors	Primary actors: Healthy Elderly, elderly with alzheimer, hypertension or diabetes Secondary actors; family members, informal care-givers, formal care-givers, service centre, telematic service providers and public / private Social security service providers and insurance companies
Related UCs	UC 2.3 Also related to UC1.3, UC1.2, UC1.1, UC2.2, UC2.1 and UC2.6
Inputs	User Profile information and nutrition questionaires
Outputs	Daily or weekly menu, Illustrated recipes, Nutritional habits, Advices: e.g. “Remember, you cannot eat sweets”.

Table 6-6. Nutritional Advisor service description

6.2.4.2 Data Storage

The data base schema for the Nutritional Advisor is quite complex, since it has to contain information about ingredients, recipes, meals and menus, as well as the users and nutritionist, and advises assigned to each user.
In figure 6-4 the initial schema of the data base is shown.

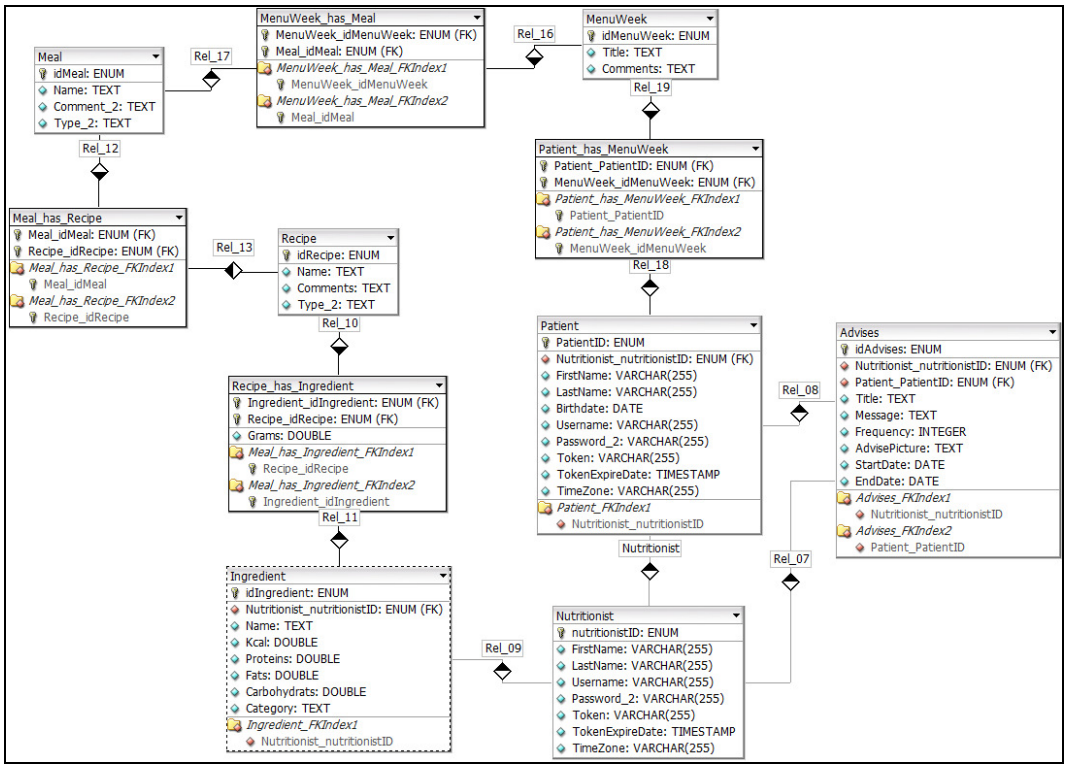


Figure 6-4. Nutritional data base schema

6.2.4.3 Classes

The Nutritional Advisor Service has multiple personalized classes used to work with all information of the data base.

- **Nutritionist:** this class is used to store nutritionist's information
  - Int DoctorID
  - String firstName
  - String lastName
  - String preferedLanguage
  - String token
  - String username
  - String password
- **Patient**, which is used to get information from patients, and it is compound by.
  - int PatientID
  - String FirstName
  - String LastName
  - Date BirthDate
  - String Username
  - String Password
  - String Token
  - String TokenExpirationDate
  - String TimeZone
  - String PreferedLanguage
  - int nutritionistID
- **Advise,** which contains all relevant information related to the advises
  - int id;
  - String title;
  - String message;
  - String picture;
  - NutriCalendar startDate;
  - NutriCalendar endDate;
  - int frequency;
- **Ingredient**
  - String name
  - String categoryID
  - double kCal
  - double proteins
  - double carbohidrates
  - double fat
  - String comment
  - int authorId
  - double grams
- **Recipe**
  - int id;
  - String recipeName
  - List<Food> ingredients

- double dishKCal
- int dishCarboH
- int dishProteins
- int dishFat
- String category
- double totalGrams

- **Meal**
  - int id
  - List<Recipe> recipes
  - String title
  - String comment
  - int categoryID
  - double totalQuantityGrams
  - int authorId
  - String eaten
  - String liked
- **Day**
  - int id;
  - int order;
  - List<Meal> meals;
  - int weekMenuID;
  - String weekMenuTitle;
  - String comment;
  - int authorId;
- **nutritionMenu**
  - int id;
  - List<Day> days;
  - String purpose;
  - String comment;
  - int authorId;

6.2.4.4 Operations

The Nutritional Service operations related to the client side are the following.

Name	Description	Inputs	Outputs
`changeMeal`	Changes a meal for the one supposed for the next day	String token	True/False
`getAdvisePictu re`	Returns the picture of a given advise	String token Int adviseID	Base64Bina ry picture
`getCustomShopp ingListDays`	Returns the shopping list for a personalized number of days	String token int numdays	shoppingLis t
`getDayMenu`	Returns the daily menu for a concrete weekday	String token Int weekday	DayMenu
`getDishPicture`	Returns the picture of a recipe	String token Int recipeID	Base64Bina ry picture
`getFoodCategor`	Returns all food categories	String token	List

ies			foodCategories
getMyAdvises	Returns the list of advises	String token	List Advises
getFoodsByCategory	Returns all the ingredients listed in a concrete category	String token Int categoryID	List FoodCategories
getThisWeekMenu	Returns the complete menu for current week	String token	List DayMenu
getTodayMenu	Returns the menu for today	String token	DayMenu
getToken	Returns user token	String user String pwd	String token
getTomorrowMenu	Returns the menu for tomorrow	String token	DayMenu
getUserRecipe	Returns a concrete recipe	String token Int recipeID	Recipe
getWeeklyShoppingList	Returns the shopping list for next week	String token	shoppingList
isTokenValid	Returns if the stored token is valid	String token	True/False

Table 6-7. Nutritional Service operations

6.2.5 Activity Advisor

6.2.5.1 Service Description

Name	Activity Advisor
Description	Advices to the user on daily or weekly basis on recommended activities, taking into account his/her implicit health profile actual health status, previous activities record, nutritional plan and actual nutrition uptake. MD controlled.
Actors	Primary actors: Healthy Elderly, Elderly with Arthritis, Elderly with Alzheimer, Elderly with Co-morbidity issue, Elderly with Diabetes, Elderly with Hypertension Secondary actors: Formal care-givers (both inpatient and outpatient), Service Centre, Health care and emergency support service providers Elderly associations, National, local and regional authorities, Policy makers
Related UCs	UC 2.1 Inputs: UC1.1, UC1.2, UC1.3, UC2.3, UC2.6 Outputs: UC 2.2, UC 3.1, UC 3.2, UC

	3.3, UC 7.2
Inputs	Information from UC1.1, UC1.2, UC1.3, UC2.3, UC2.6
Outputs	Friendly application interface with activities recommended and how to do them are shown by the system, e.g.: exercising, speed, energy expenditure, etc…

Table 6-8. Activity Advisor service description

6.2.5.2 Data Storage

The data base stores information related to patients, and doctors, and the personalized activity plans designed for the patients.

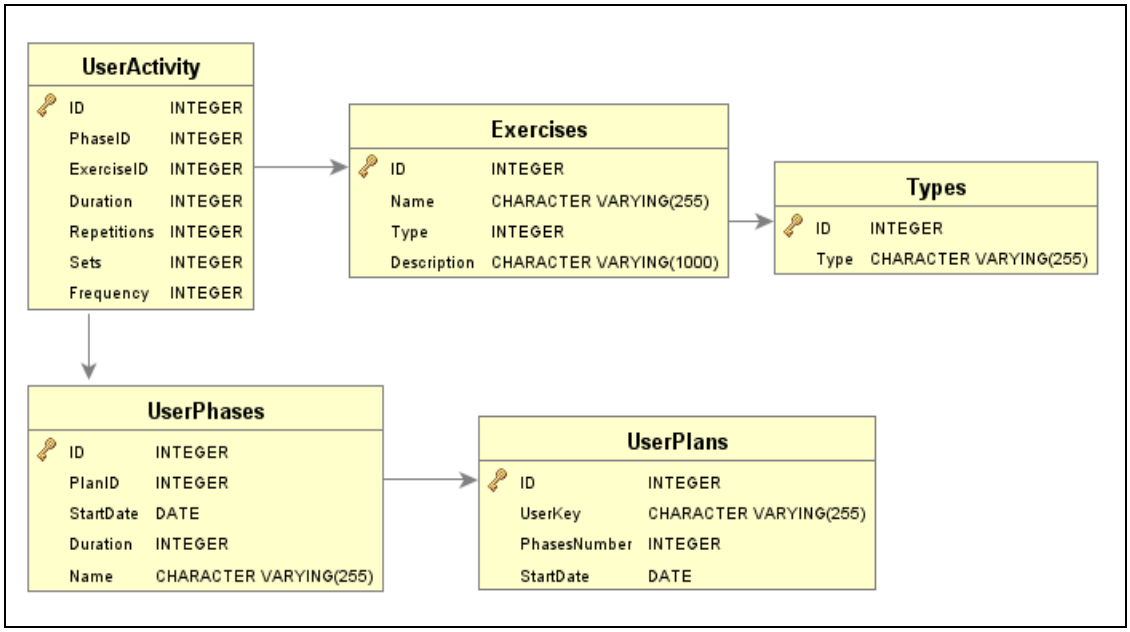


Figure 6-5. Activity Plan data base schema

6.2.5.3 Classes

In order to work with the information stored in the data base, several classes have been created.

- **Activity:** this class is used to store activity information, it is a specific and complete exercise: duration, repetitions, sets,...
  - int id
  - Exercise exercise
  - int duration
  - int repetitions
  - int sets
  - int frequency

- **Exercise**, which stores the exercise description:
  - int id
  - String name
  - String type
  - String description
- **ExerciseType,** which only stores exercise's type:
  - int id
  - String type
- **Patient**, which is used to get information from patients, and it is compound by.
  - String patientKey
  - String FirstName
  - String LastName
  - String BirthDate
  - String Adress
  - String phone
  - String insuranceNumber
  - String Sex

- **Phase,** the activity plans are divided in different phases, which contain several activities.
  - int id
  - String name
  - int planID
  - Date startDate
  - int duration
  - Activity[] activities
- **Plan,** which contains all the phases of a personalized plan
  - int id
  - String userKey
  - Phase[] phases
  - Date startDate
  -

6.2.5.4 Operations

The operations related with the client part of the service are the following

Name	Description	Inputs	Outputs
`getPatients`	Returns a list of patients assigned to a professional	String username String password	List of patients
`getExercisesTypes`	Returns a list of exercises 's type	n/a	List of Types
`getExercises`	Returns a list of exercises	n/a	List of Exercises
`setExercise`	Create a new exercise	Exercise exercise	True/false

setPlan	Create a new Plan	Plan plan	True/false
getPlan	Returns a plan	String userKey	Plan object
deletePlan	Deletes a plan	Int planId	True/false

Table 6-9. Activity Advisor operations

6.2.6 Health Trip Advisor

6.2.6.1 Service Description

Name	Trip Advisor
Description	The user wishes to travel (locally or further away) and plans his/ her trip in a trip planner. System calculates key trip attributes (i.e. time of day, external temperature, planned walking distance, transportation mode changes, need to drive, etc.), and in combination to his/her health status (actual from UC1.1 and in the UC1.3 profile) it recommends the user to change his/her plans or trip. It also suggests to the user what type of clothes to take with him/her, etc.
Actors	Primary actors: Healthy Elderly, Elderly with Diabetes, Elderly with Hypertension Secondary actors: Service centre, Health care and emergency support service providers, Telematic service providers and Other: Public transport operator, traffic operator
Related UCs	UC 4.1 Input: UC1.3, UC1.1
Inputs	Vital signs monitoring of UC 1.1. and User medical profile of UC 1.3.
Outputs	OK / not OK to follow travel plan , Alternative travelling time and Alternative travelling mode.

Table 6-10. Trip Advisor service description

6.2.6.2 Data Storage

The trip advisor web service does not use local data. It provides a simple assessment based on the following inputs:

- user id
- current heart rate (retrieved from sensors)

- heart rate limits (retrieved from user profile)
- outside temperature (retrieved either from sensors or provided by another web service connected through the alignment tool
- desired trip length entered by user
- optionally inside temperature retrieved from sensors

A professional application is used to modify the limits through medical personnel. It provides a method to query and modify stored limit values. The front end application for the user saves the user id when installed on the user's device.

6.2.6.3 Classes

The trip advisor web service is a single class providing a single method retrieving heart rate, outside temperature and the desired trip duration in minutes as input and returning a string containing respective recommendations to the end user.

The limit values of the user are retrieved from the user's profile.
The trip advisor application provides a method to query and modify the limit values by querying the health profile of the user.

6.2.7 Health Personal Calendar

6.2.7.1 Service Description

Name	Personal Calendar
Description	The *Personal Calendar* application will be used for scheduling/managing the daily tasks of the elderly (managing nutrition, medication, to-do lists, etc.) under the unobtrusive supervision of carers
Actors	Primary actors: All kinds of elderly Secondary actors: Family members, Formal care-givers (both inpatient and outpatient)
Related UCs	UC1.4 Related to UC2.1, UC2.2 and UC2.3
Inputs	Activity advisor, Medical advisor and Nutritional advisor
Outputs	Summaries of pending notifications. Visual, Auditory, and tactile (e.g., vibration if available) notifications.

Table 6-11. Personal Calendar service description

6.2.7.2 Data Storage

The database stores information about users and tasks. The schema of the data base can be seen in figure XX

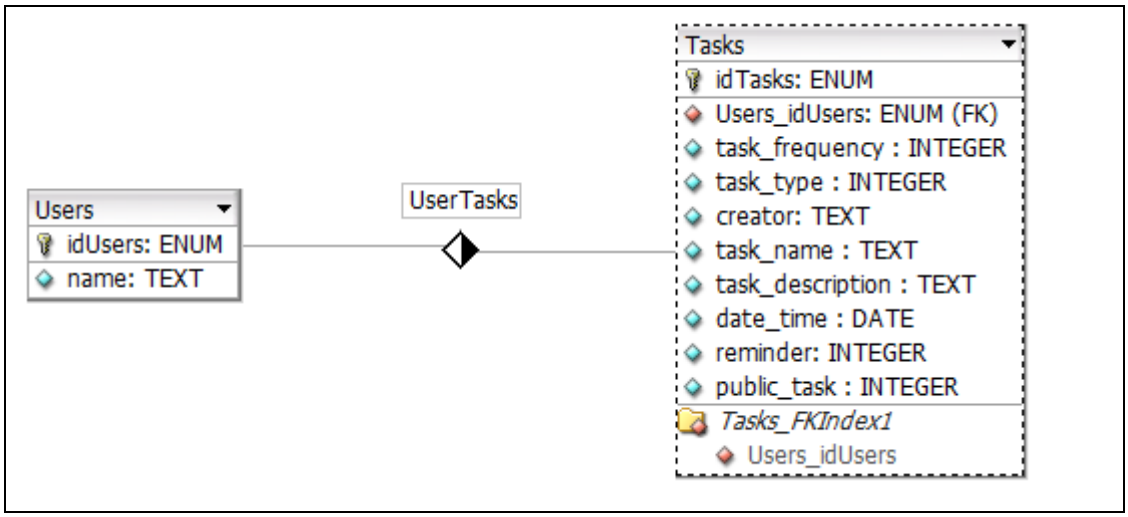


Figure 6-6. Schema of the Personal Calendar data base

6.2.7.3 Classes

The personal Calendar just requires the Task class to be defined:

- **Task**
  - long id
  - int task_frequency
  - int task_type
  - String creator
  - String task_name
  - String task_description
  - DateTime datetime
  - int reminder
  - int public

6.2.7.4 Operations

The operations related with the client part of the service are the following:

Name	Description	Inputs	Outputs
`getPrivateTasks`	Returns the private tasks or all tasks for the user, according to selected categories and/or frequencies	String user int taskFrequency int taskType boolean privateOnly String calendarAccessKey	Task [ ]
`addPrivateTask`	Adds a task to the user's calendar (public or private)	String user int taskFrequency int taskType Calendar datetime String name String description boolean reminder	Task

		boolean publicTask String calendarAccessKey	
`getTasks`	Same as getPrivateTasks, but only for public tasks	String user int taskFrequency int taskType	Task [ ]
`addTask`	Same as addPrivateTask, but only for public tasks	String user String creator int taskFrequency int taskType Calendar datetime String name String description boolean reminder	Task
`editPrivateTask`	Edits any previously existing task	String user int taskFrequency int taskType Calendar datetime String name String description boolean reminder boolean publicTask String calendarAccessKey long taskId	Task
`editTask`	Edits a public task	String user String creator int taskFrequency int taskType Calendar datetime String name String description boolean reminder long taskId	Task
`deleteTask`	Deletes a task	String user long taskId	True/false

Table 6-12. Personal Calendar client operations

6.2.8 Brain and Skills Trainer

6.2.8.1 Service Description

Name	Brain and Skills Trainer
Description	The aim of this UC is to focus on memory support of Alzheimer or other demented patients. Brain games will help

	to improve cognitive abilities and possibly to delay Alzheimer's disease onset and/or progress.
Actors	Primary actors: Healthy Elderly, Elderly with Stroke, Elderly with Parkinson and Elderly with Alzheimer. Secondary actors: Family members, Informal care-givers (i.e. volunteers, neighbours, etc.), Formal care-givers (both inpatient and outpatient), Service Centre, Health care and emergency support service providers and Infotainment service providers
Related UCs	UC3.1 and UC3.2
Inputs	n/a
Outputs	Game instructions, List of games and Scores and results.

Table 6-13. Brain and Skills Trainer service description

6.2.8.2 Data Storage

The data are stored in a flat file. There is no need for a database infrastructure, because data are quite simple and few.
Each line in the flat file contains a simple record of the following format:

12/3/2011, 58.8, Nick
23/2/2011, 60.4, George
…

6.2.8.3 Classes

The following class has been defined for the web service:

- **GameRecord:** This class is used to handle game scores:
  - string date: the date the game played
  - double score: the score of the game
  - string player: the player's name

6.2.8.4 Operations

The operations related with the client part of the service are the following

Name	Description	Inputs	Outputs
`retrieveHighScores`	Returns the best scores among all players	-	List<GameRecords>
`retreivePlayersBest`	Returns the	String	GameRecord

	best score of a specific player	username	
uploadScore	Uploads the score of a user	GameRecord	True/False

Table 6-14. Brain and Skills Trainer client operations

6.3 AUGMENTED AUTONOMY

6.3.1 Functionalities

Multi-user controlled environments for the elderly at home, in order to offer the comfort, security and safety required, especially in rural and isolated areas.

6.3.2 Services Candidates

Services Candidates List:

- Environmental Control at Home.
- Fall protection.

6.3.3 Service Environmental Control at Home

6.3.3.1 Service Description

The Environmental Control service that was initially implemented and developed for the OASIS EU project needs may be described as following:

Name	Environmental Control at Home
Description	Monitoring and control of many devices at home for user comfort, activity monitoring purposes and especially safety and security reasons. Functions automation to protect a patient with cognitive problems to forget open the oven or kitchen (safety) as well as the lights and water boiler (economy). Gradually deploy home automation facilities to substitute functional limitations in performing every day domestic duties.
Actors	Primary actors: Patient Secondary actors: Formal care-givers (both inpatient and outpatient), Service Centre, Health care and emergency support service providers, Home automation service providers
Related UCs	• UC1.2 User activity recognition

	and characterization • UC 1.3 User profile • UC1.4 Personal calendar construction and maintenance • UC5.2 In home user localization • UC5.4 Home gateway for services • UC9.2 Emergency management service
Inputs	Patient Identification, device Identification
Outputs	Environmental Data

Table 6-15. Environmental Control at Home service description

6.3.3.2 Data Storage

Mobile access model:

The database of the DomoticServer is realized as a simple XML-file based persistent storage. The following listing shows an example content of that XML-file:

```
<?xml version="1.0" encoding="UTF-8"?>
<settings>
  <properties>
    <entry key="websettingsport" value="8080"/>
    <entry key="loglevel" value="2"/>
    <entry key="dpwsport" value="11222"/>
    <entry key="wsport" value="11223"/>
    <properties node="location.configuration">
      <properties node="3bf0d9aa-0236-4a19-8b70-fa00dcb63962">
        <entry key="location" value="05f72c0b-5a9f-4468-9d6f-95c748a099c2"/>
        <entry key="name" value="Bedroom"/>
      </properties>
      <properties node="05f72c0b-5a9f-4468-9d6f-95c748a099c2">
        <entry key="name" value="Ort2"/>
      </properties>
    </properties>
    <properties node="device.configuration">
      <properties node="d00c323c-42a5-4347-8fa3-20c64aba5e6d">
        <entry key="name" value="Busware CUL-Stick"/>
        <entry                                                    key="type"
value="http://www.domologic.com/devices/BuswareCul"/>
        <entry key="serial" value="20100702"/>
        <properties node="fs20">
          <entry key="devices">
            <value>76836fed-b1db-4e7d-8ee2-e48feda7c3b1</value>
            <value>36dca36c-8968-43b2-97fb-b83d8050a24a</value>
          </entry>
        </properties>
        <properties node="hms">
          <entry key="devices">
            <value>69447f10-416e-4f7b-8152-3fd931f59d58</value>
            <value>6329f71c-11d8-486e-b47b-5937bb0de41b</value>
          </entry>
        </properties>
      </properties>
      <properties node="c8b3c424-a023-4a8c-be0e-a7b32f932952">
```

```
        <entry key="name" value="Gateway"/>
        <entry key="type" value="http://www.domologic.com/devices/Gateway"/>
      </properties>
      <properties node="76836fed-b1db-4e7d-8ee2-e48feda7c3b1">
        <entry key="address" value="0"/>
        <entry key="name" value="Motion Sensor"/>
        <entry key="type" value="http://www.domologic.com/devices/FS20PIRI"/>
        <entry key="housecode" value="24429"/>
      </properties>
      <properties node="36dca36c-8968-43b2-97fb-b83d8050a24a">
        <entry key="address" value="1"/>
        <entry key="location" value="3bf0d9aa-0236-4a19-8b70-fa00dcb63962"/>
        <entry key="name" value="Switch 1"/>
        <entry key="type" value="http://www.domologic.com/devices/FS20ST"/>
        <entry key="housecode" value="24429"/>
        <properties node="power"/>
      </properties>
      <properties node="69447f10-416e-4f7b-8152-3fd931f59d58">
        <entry key="address" value="20137"/>
        <entry key="name" value="Fluid Detector"/>
        <entry key="type" value="http://www.domologic.com/devices/HMS100W"/>
      </properties>
      <properties node="6329f71c-11d8-486e-b47b-5937bb0de41b">
        <entry key="address" value="18977"/>
        <entry key="name" value="CH4-Gas Sensor"/>
        <entry key="type" value="http://www.domologic.com/devices/HMS100MG"/>
      </properties>
    </properties>
  </properties>
</settings>
```

Figure 6-7. XML Example

It is stored as a simple hierarchical property tree describing all properties of the DomoticServer. There are two main tags: location-configuration that describes all locations; and device-configuration that contains the settings for all known devices on the DomoticServer.

PC access model:

The database stores the information about the actions that have been taken on the smart home environment (Events history, e.g. “Monday 05/09/2010 8:00 a.m : Bedroom ligths switched on”)

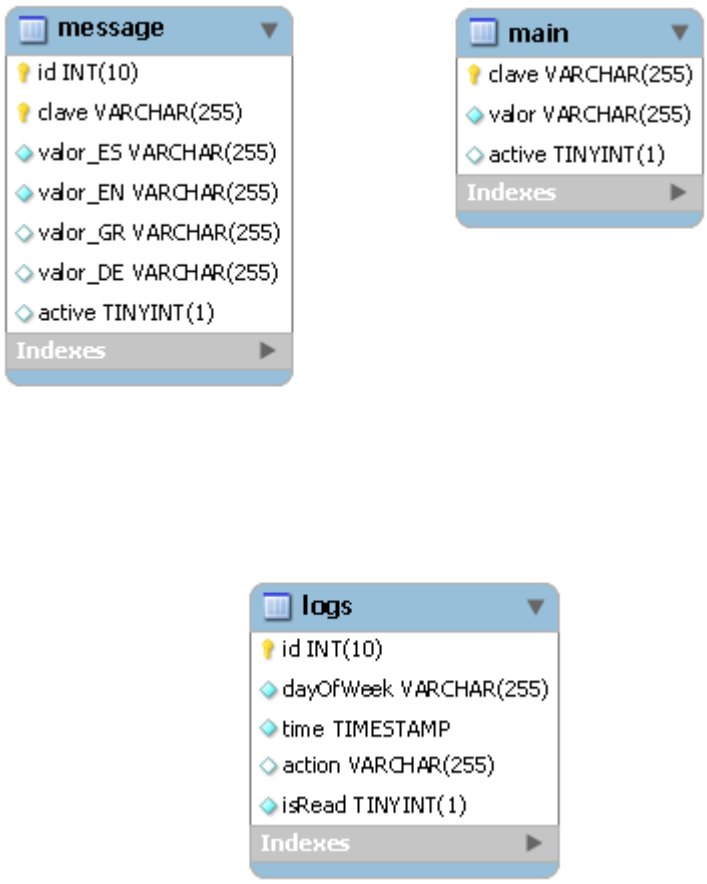


Figure 6-8 PC Data model

6.3.3.3 Classes

Mobile model:

The following table describes all interfaces supported by the DomoticServer until now:

Interface	For a device, that…
`BatteryStateProvider`	has a battery and can provide its state
`ButtonProvider`	contains some buttons that can be pressed
`ConsumerSwitchMonitor`	monitors a consumer device and determines, if it was enabled or disabled
`ContactSensor`	monitors a contact (for example a reed-contact, or simply two wire contact) and determines, if it was closed or open
`DeviceProvider`	provides a list of devices, that are described by this interfaces
`Dimable`	can be dimmed by a value, usually used for standard dimmer devices
`FluidSensor`	can detect fluids
`GasSensor`	can detect gasses

Interface	For a device, that…
Heating	can control or monitor a heater
HumiditySensor	provides humidity values
LightSensor	provides light (intensity) values
Location	is assignable to a location, all devices on the DomoticServer are assignable by default
LocationProvider	can control the configuration of all locations
MotionSensor	monitors motions
PowerConsumptionMeter	monitors power consumption
PropertiesProvider	provides a storage for properties (key-value pairs), all devices on the DomoticServer provides a storage for properties by default
RainMeter	monitors precipitation
Signaling	can make a noise or attract attention in other way, for example a gong
Switchable	contains something, that can be turned on and off
TemperatureSensor	provides temperature values
WindSensor	measures wind speed

Table 6-16. Domotic server interfaces

PC model:

The service for the Environmental Controller has multiple personalized classes:

- **EnvControl_AppController:** Main class.
- **ConnectionManager:** This class decides between either AMI or HomeAutomation object to interact with the home automation technology.
- **AMIConnection:** This class is intended to connect with the home automation technology by using the AMI technology.
- **HomeAutomationMgmt:** This class is intended to connect with the home automation technology by using the WS operations directly.
- **GUI:** The GUI class.
- **EnvController_HTTPManagement:** It checks if a HTTP connection is available.

6.3.3.4 Operations

Mobile model:

Domotic devices supported by the DomoticServer have to be organized meaningful. This affects the hierarchy of the devices, to be controlled by other applications. Partitioning the devices into *white goods* (e.g. washing machines, refrigerators etc.) and *brown goods* (e.g. TV or CE player) is applicable for catalogs or shops. To categorize devices supported by the DomoticServer, this approach brings no relevant information. Also distinguishing between a sensor (providing measurement data) and an actor (executing commands) is not

useful: E.g. if we turn on an actuator, and then we ask afterwards if the actuator has been turned on, we are converting the actuator to a sensor indirectly. This way the actuator is always a sensor, because we are able to cache its current state (it makes no difference if the cache functionality is implemented directly on the device, providing a get-operation for its current state, or if the cache functionality the cache is on the application side, controlling the device).

What we need is an abstraction layer for device features. We can always describe what features are provided by a specific device. For example the feature **Switchable** has the ability to turn on and off something and retrieve its state. This can be described by an interface with the operations **isEnabled** and **setEnabled**. Now we can take all devices, that can be turned on and off, and describe them with that interface.

Some devices may contain two or more independent features described by the same interface. To support this, all the interfaces and their operations need an index parameter. For example, if a device with two actors is described by the **Switchable** interface, we can call **setEnabled(index)** and **isEnabled(index)**, where the index lets us distinguish between the two actors. Also we need an operation, which returns the possible indices. Because the first index should always be a 0, and no index should remain “unused”, we only need an operation which returns the count of all possible indices. For **Switchable** this operation is called **getSwitchableCount**.

PC model:

Name	Description	Inputs	Outputs
light	To act over lights	String symbolicName, Int valor	Void
lightValue	To get the lights state	String symbolicName	String
blind	To act over blinds	String symbolicName, Int valor	Void
blindValue	To get the blinds state	String symbolicName	String
door	To act over main door	String symbolicName,int valor	Void
doorValue	To get the door state	String symbolicName	String
largeWindow	To act over largewindows	String symbolicName, int valor	void

Table 6-17 PC model. Operations

6.3.4 Fall Protection

6.3.4.1 Service Description

Name	Fall Protection
Description	Detects when the user falls down at home and delivers appropriate alerts to the emergency services. A communication link can be open with the user to give him/her support to manage the situation while help arrives. The detection of user's fall can be extended to outside, where the user should be equipped with sensor-enhanced devices (e.g. a wearable device) and carry a smartphone
Actors	Primary actors: Patient Secondary actors: Family members, Informal care-givers (i.e. volunteers, neighbours, etc.), Formal care-givers (both inpatient and outpatient), Service Centre, Health care and emergency support service providers
Related UCs	• UC 1.3 User profile building (to get the contact number) • UC 3.5 Social networking (to carry out the emergency call) • UC5.2 In home user localization • UC5.4 Home gateway for services • UC 7.2 Patient activity monitoring • UC9.2 Emergency management service
Inputs	Data Sensors
Outputs	A message in the screen of the PC or the mobile device asking the user for confirmation of the incident and help support triggering. Generation of a call in case the user needs that help or – automatically – if he/she does not answer the request within a minute

Table 6-18. Fall Protection

6.3.4.2 Data Storage

This service adopts the database which is described in section 6.1.3 - Vital Signs Monitoring, for all the necessary storage requirements.

6.3.4.3 Classes

The main class used is the class **"FallMeasurement"**. It is consisted of the properties:

- String patientID
- String Date
- Double Value
- Double lowThreshold
- Double highThreshold

6.3.4.4 Operations

Name	Description	Inputs	Outputs
`setPostureThresholdsValueByPatientId`	Sets posture threshold for a patient	Int patientID Double Threshold	True/false
`setPostureValueByPatientId`	Sets posture ECG value for a patient	Int patientID Double Meassurement	True/false

Table 6-19. Fall protection services operations

6.4 PROFESSIONAL HEALTHCARE

6.4.1 Functionalities

Services offered to the professional in order to visualize patient data, to access electronic medical records, having decision support tools form a Web-based point of view.

6.4.2 Services Candidates

Services Candidates List:

- BPM-Activity Plan
- BPM-Nutrition Plan
- BPM-Patient Medication Plan
- BPM-Training Plan
- BPM-Trip Plan
- Cognitive Problem Prognosis
- Decision Support Tool
- Patient Activity Monitoring
- Patient Nutrition Monitoring
- Patient Records Monitoring

6.4.3 Service BPM-Activity Plan

6.4.3.1 Service Description

Name	BPM-Activity Plan
Description	Business Process Management for Elaborate an Activity Plan
Actors	Patient, Professional Career
Related UCs	• UC2.1 Activity Advisor • UC2.4 Monitoring of user compliance to activity plans • UC7.2 Activity Monitoring
Inputs	UC8.2 Care Planning Assistant
Outputs	Patient Activity Plan

Table 6-20. BPM-Activity Plan

6.4.3.2 Data Storage

The schema of the data base can be seen in section 6.2.5.2.

6.4.3.3 Classes

The personalized classes used are described in section 6.2.5.3.

6.4.3.4 Operations

The operations corresponding to the professional part of the Activity Advisor are the following

Name	Description	Inputs	Outputs
`getPatients`	Returns all patients of a doctor	String token Int doctorID	Patient[] patients
`createPlan`	Creates a new activity plan for a patient	String token Int patientID ActivityPlan plan	True/false
`retrievePlanbyNHC`	Returns the activity plan of a patient	Int patientID	ActivityPlan plan
`createActivity`	Allows doctor to create new activities	String token Activity activity	True/false
`createPhase`	Allows doctors to create personalized	String token Phase phase	True/false

	phases		

Table 6-21. Activity advisor operations

6.4.4 Service BPM-Nutrition Plan

6.4.4.1 Service Description

Name	BPM-Nutrition Plan
Description	Business Process Management for Elaborate a Nutrition Plan
Actors	Patient, Professional Career
Related UCs	• UC2.3 Nutritional Advisor • UC2.6 Monitoring of user compliance to nutritional plans • UC7.3 Patient Nutrition Monitoring
Inputs	UC8.2 Care Planning Assistant
Outputs	Patient Nutrition Plan

Table 6-22. BPM-Nutrition Plan

6.4.4.2 Data Storage

The schema of the data base can be seen in section 6.2.4.2.

6.4.4.3 Classes

The personalized classes used are described in section 6.2.4.3.

6.4.4.4 Operations

The operations that allows the professionals to manage the information of the data base are the following

Name	Description	Inputs	Outputs
`addNewAdviseToPatient`	Adds a new advise to a patient	String token, int patientID, Advise	True/False
`editAdvise`	Allows editing existing advises	String token, int patientID, Advise	True/False
`getPatientsAdvises`	Returns all advises of a patient	String token, int patientID,	List Advises
`adjustMenuToGoalKCal`	Adjust a given menu to a goal Kcal	String token, Int menuID Double	nutritionMenu

		goalKcal	
assignMenuWeekToPatient	Assigns a menu week to a patient	String token, nutritionMenu menu	True/false
deleteMenuFromMenuCalendar	Deletes a menu for a patient's plan	String token, int patientID, Int menuID	Treu/false
getMenuSourceList	Gets all menus of a source	String token,	List nutritionMenu
getMenuWeek	Returns a menu week	String token, int menuID	nutritionMenu
getPatientsList	Returns the list of all patients of the nutritionist	String token	List Patients
getToken	Returns the token of the nutritionist	String user String pwd	String token
Login	Logsn in the nutritionist in the service	String user String pwd	String token
searchPatientByNameLastname	Allows nutritionist to search for a patient by name or last name	String Name, Strin Lastname	Patient
storeMenuWeek	Stores a new week menu in the data base	String token nutritionMenu menu	True/false
updateMenuWeek	Updates a menuweek	String token nutritionMenu menu	True/false

Table 6-23. BPM-Nutrition Plan Operations (I)

Name	Description	Inputs	Outputs
createMeal	Creates a new meal in the DB	String token Int nutritionistID	String success/error
createNewFood	Creates a new ingredient in the DB	String token Int nutritionistID	String success/error
createRecipe	Creates a new recipe in the DB	String token Int nutritionistID	String success/error
createWeek	Creates a new week menu in the DB	String token Int nutritionistID	String success/error
getDay	Returns the meals of a day	String token Int nutritionistID Int day	Day day
getDishTypes	Return all recipes types	String token	List

		Int nutritionistID	DishCategory
getFood	Returns the information of a given ingredient	String token Int foodID	Ingredient
getFoodCategories	Returns a list with all the categories of the ingredients	String token Int nutritionistID	List foodCategory
getFoods	Returns all ingredients	String token Int nutritionistID	List Ingredient
getMealCategories	Returns all categories of the meals	String token Int nutritionistID	List mealCategory
getMeal	Returns a given meal	String token Int mealID	Meal
getMeals	Returns all meals	String token Int nutritionistID	List Meal
getNutritionists	Returns all nutritionists	String token Int nutritionistID	List nutritionist
getRecipe	Returns a given recipe	String token Int recipeID	Recipe
getRecipes	Returns all recipes	String token Int nutritionistID	List Recipe
getToken	Returns the nutritionist token	String user String pwd	String token
getWeeks	Returns all menu week	String token Int nutritionistID	List nutritionalMenu
updateDay	Changes one day	String token Int nutritionistID Day day	String success/error
updateFood	Changes one ingredient	String token Int nutritionistID Ingredient ing	String success/error
updateMeal	Changes one meal	String token Int nutritionistID, Meal meal	String success/error

updateRecipe	Changes one recipe	String token Int nutritionistID Recipe re	String success/error
updateWeek	Changes one menu	String token Int nutritionistID nutritionMenu nm	String success/error

Table 6-24. BPM-Nutrition Plan Operations (II)

6.4.5 Service BPM-Patient Medication Plan

6.4.5.1 Service Description

Name	BPM-Patient Medication Plan
Description	Business Process Management for Elaborate a Medication Plan
Actors	Patient, Professional Career
Related UCs	• UC2.2 Health Advisor • UC2.5 Monitoring of user compliance to medical treatment • UC7.1 Record Monitoring
Inputs	UC8.2 Care Planning Assistant
Outputs	Patient Medication Plan

Table 6-25. BPM-Patient Medication Plan service description

6.4.5.2 Data Storage

The schema of the data base can be seen in section 6.2.3.2.

6.4.5.3 Classes

The personalized classes used are described in section 6.2.3.3.

6.4.5.4 Operations

The operations related to the professional part of the Health advisor are the described in the following table.

Name	Description	Inputs	Outputs
loginDoctor	Logs in the doctor in the service	String user, String password	Int doctorID
getPatients	Gets the patient's list of a	Int doctorID	List

	doctor		patients
assignMedicationtoPatient	Assigns a medicament to a patient	medicamenPlan	True/false
removePatientMedication	Deletes a medicament from a patient	Int medicamentID	True/false
removeCalendarTasks	Deletes calendar tasks before removing a medicament	Int patientID, int planID	Int task
createMedication	Creates a new medicament which is not in the DB	Medicament	True/flase
getMedicament	Gets all the information about a medicament	Int medicamentID	Medicament
getMedications	Returns a list with al medicaments in the DB	Int doctorID	List Medicaments
getTakenMedications	Returns a list of marked tasks as done, which means the medication has been taken	Int patientID, Date from	List takenMedicaments
isMedicationTaken	Returns if a task as already been marked as done today	long taskID	True/false
getExpiredMedications	Returns a list with the medications with expiration date before today	Int patientID	List medications
markMedicationasTaken	Mars a medication as taken	long taskID	True/false
searchByActive	Searches all medicaments with an active	String Active	List medicaments

Table 6-26. Health advisor operations.

6.4.6 Decision Support Tool (Health Advisor)

6.4.6.1 Service Description

Name	Decision Support Tool
Description	Professional career support tool that supports the patient state evaluation, problems prognosis and treatment planning, based upon his/her profile of UC1.3, as well as UCs 2.4, 2.5, 2.6, 7.1, 7.2, 7.3 and 7.4 data. Is a dynamic tool, running upon professional career request

	at any moment. It is not deciding itself, it is simply for career support.
Actors	Primary actors: Professional career Secondary actors: Formal care-givers (both inpatient and outpatient), Service Centre, Health care and emergency support service providers, Telematic service providers
Related UCs	• UC7.1 to 7.4 • UCs 2.4, 2.5, 2.6 • UC 1.3
Inputs	
Outputs	Patient related data in the server database.

Table 6-27. Decision Support Tool service description.

6.4.6.2 Data Storage

Decision support tool is based on all the information that is collected through the Health Advisor web application. There are two important sources:

1. Health profile information:

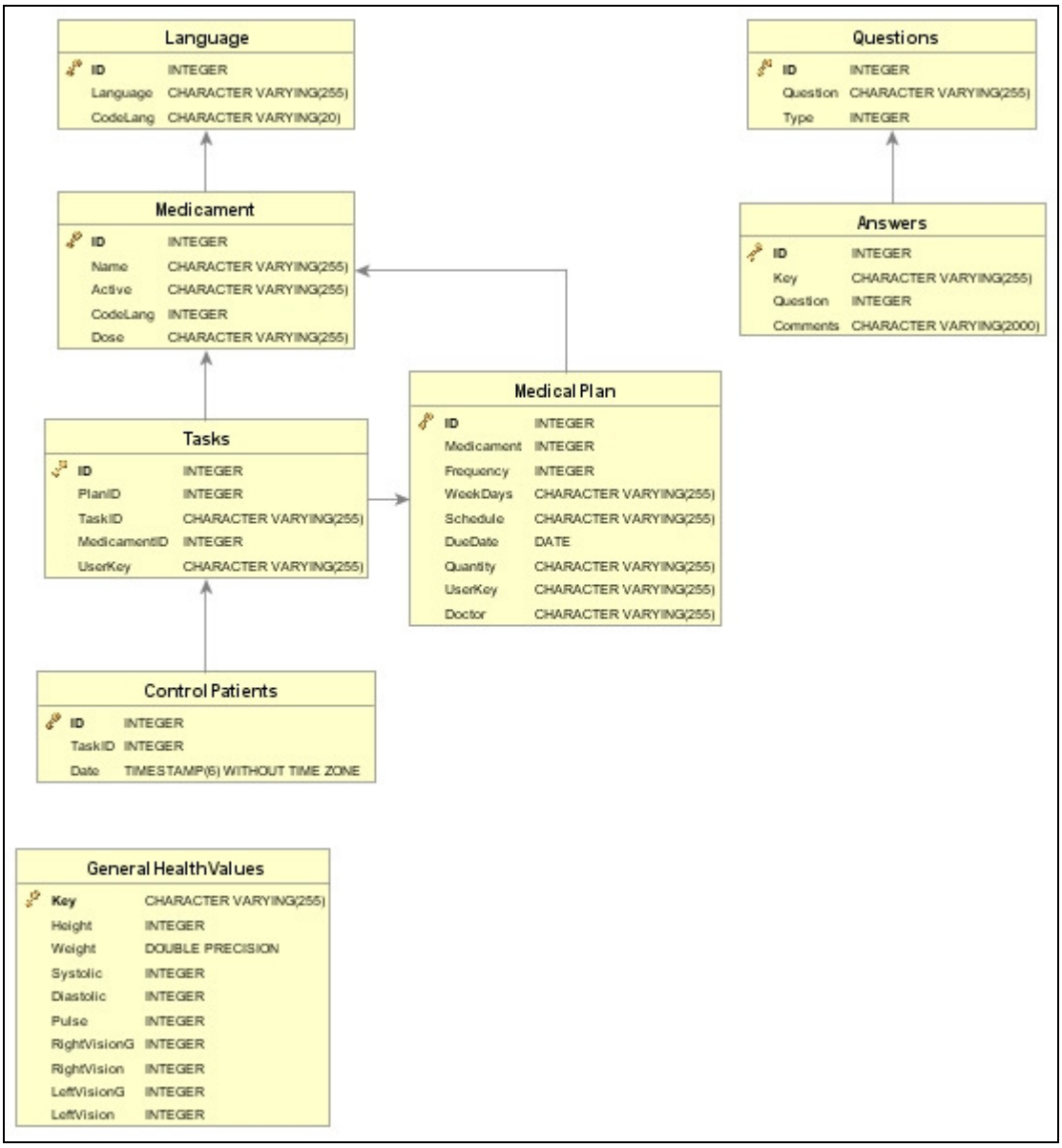


Figure 6-9. Health Advisor DB

2. User vital sign values retrieved from 6.1.3.

6.4.6.3 Classes

The personalized classes used are described in section 6.1.3.3 and 6.2.3.3.

6.4.6.4 Operations

The related operations used are described in 6.1.3.4 and 6.2.3.4.

6.4.7 Patient Activity Monitoring

6.4.7.1 Service Description

Name	Patient Activity Monitoring
Description	Monitoring activity of the patient on regular basis and per request. The objectives of this use case are: • Indicate to the patient the physical activities to be performed. • Monitor that physical activity is performed correctly. • Detect improper motion and falls.
Actors	Primary actors: Patient Secondary actors: Family members, Informal care-givers (i.e. volunteers, neighbours, etc.), Formal care-givers (both inpatient and outpatient)
Related UCs	• UC1.2 • UC 5.3 • UC 2.4 • UC 2.1.
Inputs	Patient Activity
Outputs	System provides information, guidelines, instructions and support for physical activity. System monitors physical activity of a patient. System sends results of a physical activity to a supervisor. System detects an improper motion and alert to carers.

Table 6-28. Patient Activity Monitoring service description.

The monitoring of the activity will be done through the Vital Signs monitoring service (section 6.1.3), so this service does not require further specification.

6.4.8 Patient Nutrition Monitoring.

6.4.8.1 Service Description

Name	Patient Nutrition Monitoring

Description	Monitoring and storage of patient nutrition patterns data (collected from UC2.6) for use by his/her carer. Suggestions of recipes and meals will be provided by UC2.3. Selection of key parameters to communicate and store data according to the patient's nutrition patterns under the carer choice. Automatic weight monitoring. Algorithms for carer alerts raising (mainly carer defined).
Actors	Primary actors: Patient Secondary actors: Formal care-givers (both inpatient and outpatient), Service Centre, Health care and emergency support service providers, Telematic service providers
Related UCs	Input from: • UC2.3 • UC2.6. Interacts with: • UC7.5.
Inputs	Nutrition Data
Outputs	SMSs, flags, alerts, GUI for the carer to access the data.

Table 6-29. Patient Nutrition Monitoring service description.

The Nutrition monitoring will be performed through the client application, where the user can mark if he has eaten and liked a meal.
The data storage and the personalized classes are already described in section 6.2.3

6.4.8.2 Operations

There are two operations used to monitor the nutrition of the user. The first one is used by the user to mark meals as liked and eaten, while the second one is the one used by the doctor to retrieve a menu week for a patient and be able to check if the patient has eaten and liked the meals.

Name	Description	Inputs	Outputs
`updateMealUserLikedEaten`	Stores in the DB if the patient has eated and/or liked one meal.	String token	True/False
`getMenuWeek`	Returns a menu week	String token Int menuID	nutritionMenu

Table 6-30. 6.4.10 Patient Nutrition Monitoring operations.

6.4.9 Patient Medication Monitoring

6.4.9.1 Service Description

Name	Patient Medication Monitoring
Description	Monitoring and storage of patient medical patterns data (collected from UC2.6) for use by his/her carer. Suggestions of recipes and meals will be provided by UC2.3. Selection of key parameters to communicate and store data according to the patient's medical patterns under the carer choice. Automatic weight monitoring. Algorithms for carer alerts raising (mainly carer defined).
Actors	Primary actors: Patient Secondary actors: Formal care-givers (both inpatient and outpatient), Service Centre, Health care and emergency support service providers, Telematic service providers
Related UCs	Input from: • UC2.2 • UC2.5. Interacts with: • UC7.5.
Inputs	Medical Data
Outputs	SMSs, flags, alerts, GUI for the carer to access the data.

Table 6-31. Patient Medication Monitoring service description.

The Medical monitoring will be performed through marking the calendar tasks as done. Once the user marks a task as done, the Calendar will call a service to store the task in the Medical data base, so the doctor can consult when the patient has taken his medications.

6.4.9.2 Operations

The operations related with the medication monitoring are the following:

Name	Description	Inputs	Outputs
`isMedicationTake`	Returns if a medication task has already been mark	Long taskID	True/False

	as taken today.		
markMedicationasTake	Stores in the data base that a medication task has been done.	Long taskID	True/False
getTakenMedications	Returns all medication tasks marked as done from a given date to today	Int patientID Date from	Medicament[] meds

Table 6-32. Patient Medication Monitoring operations.

6.4.10 Patient Records Monitoring

6.4.10.1 Service Description

Name	Patient Records Monitoring
Description	Monitoring and storage of patient health (from UC1.1) records for use by his/her MD. This use case includes the storage and view the patient monitoring results, which determines the evolution of the patient. Also authorized personnel could remotely collect data for analysis.
Actors	Primary actors: Patient Secondary actors: Family members, Informal care-givers (i.e. volunteers, neighbours, etc.), Formal care-givers (both inpatient and outpatient), Home automation service providers
Related UCs	• UC1.1 • UC 1.3
Inputs	Patient Record Data
Outputs	The result of health monitoring is stored and then the information is able to be consulted by authorized people as patient, doctors and carers.

Table 6-33. Patient Records Monitoring service description.

6.4.10.2 Data Storage

This service will use a database which will contain the thresholds for all kinds of vital signs being monitored for every patient, as well as the measurements taken by the patients, that is shown in 6.1.3.1 vital sign monitoring.

6.4.10.3 Classes

The classes of this service are shown in 6.1.3.3.

6.4.10.4 Operations

A description of these service operations is shown in 6.1.3.4.

6.5 INFORMAL TELE-ASSISTANCE

6.5.1 Functionalities

Applications to help informal caregivers and relatives giving information about the elderly, tools for observing and communicating directly with the elderly, access to home automation system, etc.

6.5.2 Services Candidates

Services Candidates List:

- Care Planning Assistance Service

6.5.3 Care Planning Assistance

6.5.3.1 Service Description

Name	Care Planning Assistance Service
Description	Tool for the caregiver, to manage the information collected about the patient through different ways.
Actors	Primary actors: Secondary actors: Formal care-givers (both inpatient and outpatient)
Related UCs	• UC8.1 Informal assistant – patient dialogue support. • UC1.4 Personal calendar construction and maintenance. • UC2.1 Activity advisor (related to health and nutrition). • UC2.2 Medical advisor (related to activity nutrition). • UC2.3 Nutritional advisor (related to health and activity). • UC2.4 Monitoring of user compliance to activity plans. • UC2.5 Monitoring of user compliance to medical treatment. • UC2.6 Monitoring of user compliance to nutritional plans. • UC7.5 Decision support tool for patient and treatment administration.

	• UC7.6 Treatment planning assistant. • UC10.1 System Administration. • UC10.2 Patient subscription. • UC10.4 Informal career subscription. • UC8.3. Automated alerts and periodic health status reporting.
Inputs	All data patient related
Outputs	The caregiver will receive a form with updated information about the patient, like the last data collected from the sensors, the MD orders and any other additional information provided about the patient

Table 6-34. Care Planning Assistance Service service description.

6.5.3.2 Data Storage

The database schema is detailed in section 6.6.3.2 Data storage (Emergency management service).

6.5.3.3 Classes

The service for the informal caregiver has multiple personalized classes used to work with all information of the data base.

- **RemoteMedicalCenterCarerWebService:** This class is used as main entry points to the WS.
- **MedicalCenterCarerMgmt:** This class is used as intermediate class between main entry point class and data model classes.
- **BDRemoteMedicalCenterCarer:** Main class in the data model. It is used in order to retrieve informal caregiver related data from the data base:
  - BDServiceConnector connector.
- **BDServiceConnector:** Helper class in the data model. It contains primitive operations that are used to retrieve data directly from data base.
  - String url.
  - String schema.
  - String user.
  - String pssw.
  - Connection conn.
  - Resultset rset.
  - PreparedStatement pstmt.
- **Alert:** Getter&Setter class which is used to store information about alerts.
  - String idUser.
  - String firstname.
  - String lastname.
  - String code.

- String activationType.
- String status.
- String measureName.
- String measureValue.
- String measureUnit.
- Boolean isFall.
- String messageForCarer.
- long timestamp.

6.5.3.4 Operations

Name	Description	Inputs	Outputs
getPatientAlarms_forCarer	Using this operation the Informal carer's applications retrieve information about the alerts stored on the Emergency service center.	String caregiverID, String patientID, Calendar from, Calendar to	Alert[]

Table 6-35. Care Planning Assistance service operations

6.6 MEDICAL CONTACT CENTRE

6.6.1 Functionalities

A connection service with the centre will be provided in order to allow work group activity management and coordination of patients living at home

6.6.2 Services Candidates

Services Candidates List:

- Emergency Management Service
- Users Management Service

6.6.3 Users Management Service

6.6.3.1 Service Description

Name	Users Management Service
Description	Medical Contact Centre (MCC) administrates all the involved end users of REMOTE. It registers new patients, their services, equipment and cost, assigns their healthcare professionals and informal caregivers.
Actors	Health service provider, Health care professionals, informal carers,
Related UCs	▪ UC 9.1 Medical Centre Administration tool ▪ UC 9.2: Emergency Management Service
Inputs	Patient data, Health care professionals data, Carers' data, services and equipment associated with the patient
Outputs	Data regarding the patient, health care professionals and carers, information regarding services and equipment for the patient, user profile data

Table 6-36. Users Management Service description.

6.6.3.2 Data Storage

The MCC Users Database contains information about the REMOTE users along with details about the services provided to each patient. Specifically it contains:

- Patient contact data and other types of information that maybe required (language, education, occupation, marital status, health insurance information).
- Healthcare professionals contact data and specialization (doctor, nurse, nutritionist, physiotherapist)
- Informal carers contact data, relationship to patient and other types of information that maybe required (language, education, occupation).
- Patient's healthcare professionals and informal carers
- Patient's services including equipment and costs
- Emergency services and contact details.

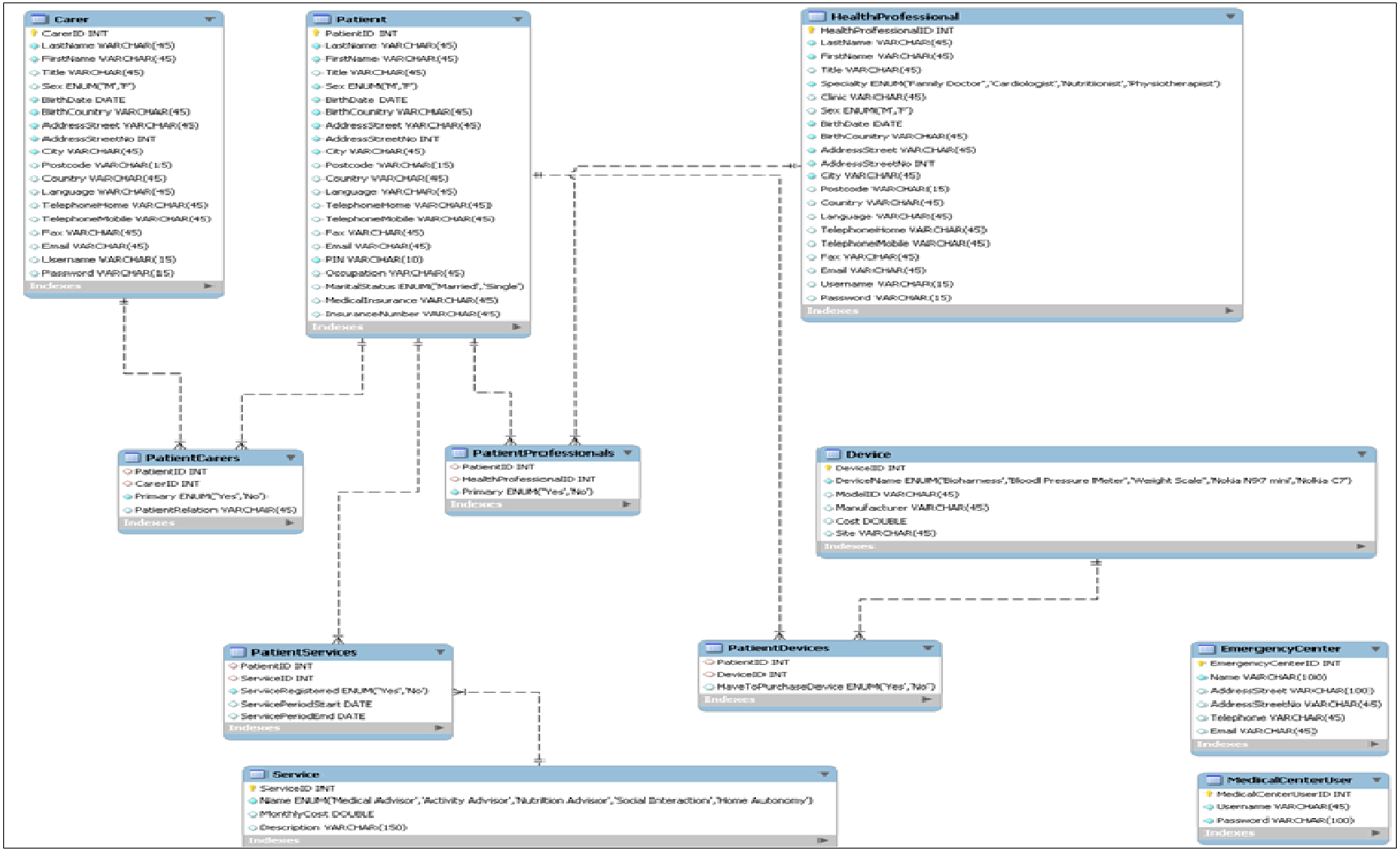


Figure 6-10. Users Management Service DB diagram

6.6.3.3 Classes

The Medical Contact Centre uses the following remote services:

- User Profile service
- Calendar service
- Nutritional advisor service

6.6.3.4 Operations

In the Medical Contact Centre the operator can perform the following operations for patients, health care professionals and for the carers:

- Add a new registration with all the corresponding associated data
- Edit and existing registration
- Delete an existing registration
- List an existing registration

When a new patient is added to the Medical contact centre his user profile is initialized and is updated by.

6.6.4 Emergency Management Service

6.6.4.1 Service Description

Name	Emergency Management Service
Description	Visualization and management of automatically driven or patient driven emergency calls and alerts.
Actors	Primary actors: Secondary actors: Informal care-givers (i.e. volunteers, neighbours, etc.), Formal care-givers (both inpatient and outpatient), Service Centre, Health care and emergency support service providers
Related UCs	• UC8.3 • UC6.1 • UC5.4 • UC6.2 • UC 9.2
Inputs	Patient, device and sensor alert
Outputs	The system asks the user to specify the type of emergency and whether to send user's medical data to a Centre. If the user does not respond within a small time limit, the system sends the data to the medical centre and notifies a generic

	emergency centre for support, providing also the user location. The user is notified that help is on the way, with approximate response time. The system notifies an emergency centre, sending the user's location, user group and medical file. It provides feedback to the user on actions performed and that help is on the way, giving estimated help arrival time. It opens a communication line between user and centre.

Table 6-37. Emergency Management Service description.

6.6.4.2 Data Storage

Alerts database. This database is used to store all the information related to emergency situations. It will be physically located in the Medical Contact Centre. All the alerts triggered automatically or manually by elderly users are stored in this DB. Also, the operator of the medical centre can add information about how the alert was managed, and write messages for the informal caregiver or the health professional. This information is stored in this database as well.

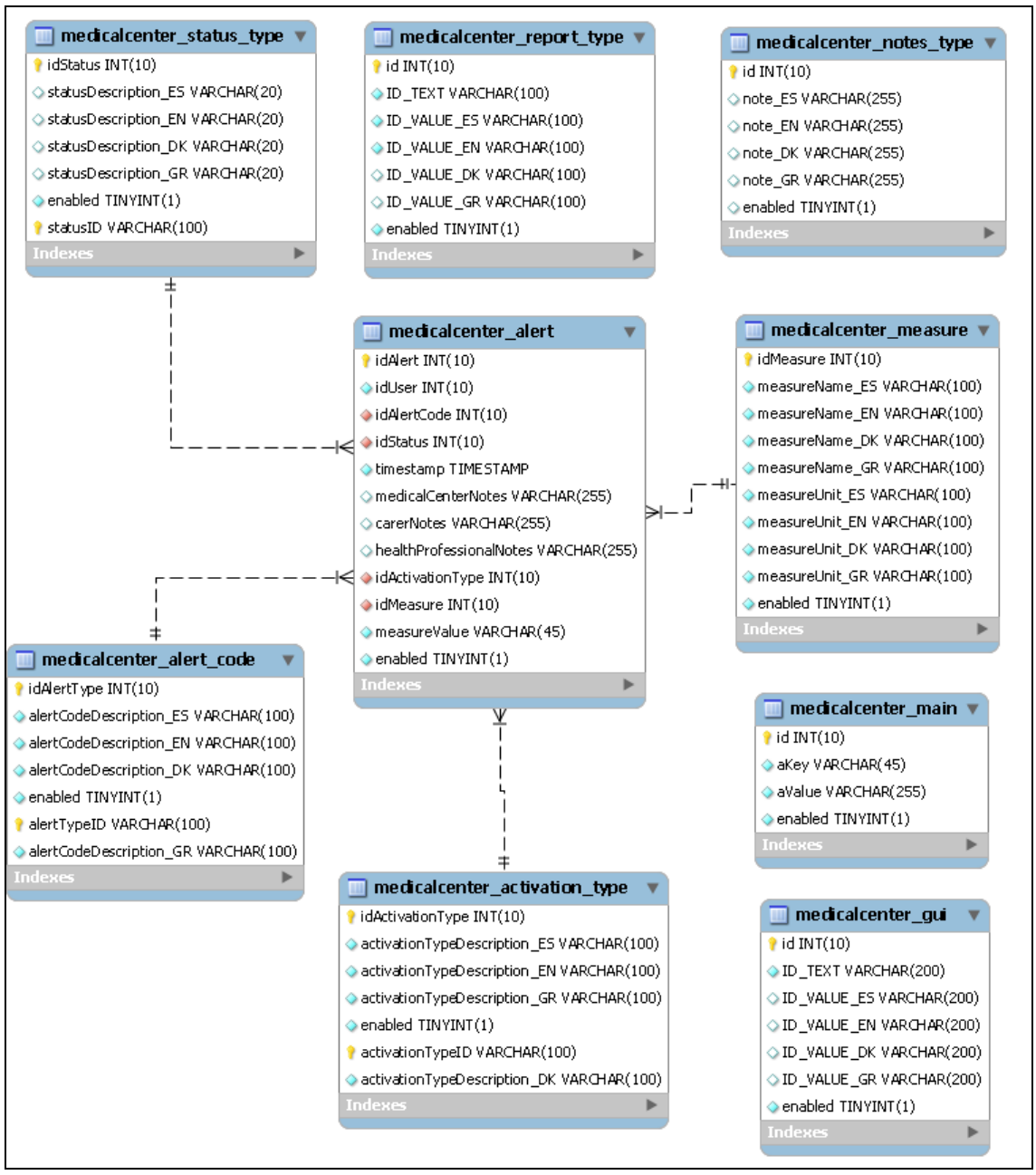


Figure 6-11. Emergency service DDBB

6.6.4.3 Classes

The emergency management service has two main components. The first one is the Web service that is used by the user's application, launching the alerts (automatically triggered by sensors or manually using the panic button) and sending the information to the Medical Centre application. The second module is the application of the Medical Centre itself, which receives the alerts and manages them according their priority.

The module in charge of the web service for the user's application manages the following classes:

- **MedicalCenterWebService:** This class is used as main entry point to the WS.

- **MedicalCenterEmergencyMgmt:** This class is used as an intermediate class between above class and the below classes.
- **BDRemoteMedicalCenterPatient:** This class contains several operations in order to register, into DB, any kind of alert (and their data)
- **BloodPressure:** This getter&setter class maps the BloodPressure object:
  - Int highPressure.
  - Int lowPressure.
- **VitalSign:** This getter&setter class maps the VitalSign object:
  - HeartRate heartRate.
  - BreathingRate breathingRate.
  - ActivityLevel activityLevel.
  - Hydration values.
- **HeartRate:** This getter&setter class maps the HeartRarte object inside the VitalSign class:
  - Int hr.
- **BreathingRate:** This getter&setter class maps the BreathingRate object inside the VitalSign class:
  - int bpm.
- **ActivityLevel:** This getter&setter class maps the ActivityLevel object inside the VitalSign class:
  - float activityLevel.

The module in charge of the Medical center itself has multiple personalized classes used to work with all information of the data base

- **RemoteMedicalCenterEmergencyMgmt:** Main class.
- **RemoteMedicalCenterGUIMgmt:** This class contains those methods which are used to provide more functionality to the GUI class.
- **RemoteMedicalCenterGUI:** This class is intended to show the GUI.
- **RemoteMedicalCenterDB:** This class is a DB Management class. It is used to retrieve/store information of alerts.
- **RemoteMedicalCenterDBManagement:** This class is a DB Management class. It is used to retrieve information about the health professional, the informal caregiver and patients general data.
- **Alert:** Getter&Setter class which is used to store information about alerts.
  - String idUser.
  - String firstname.
  - String lastname.
  - String code.
  - String activationType.
  - String status.
  - String measureName.
  - String measureValue.
  - String measureUnit.
  - Boolean isFall.
  - String messageForCarer.
  - long timestamp

- **EmergencyServiceCenter:** Getter&Setter class which is used to store information about the Emergency Service Center:
  - String name.
  - String address.
  - String addressNo.
  - String zipcode.
  - String city.
  - String telephone.
  - String email.
- **HealthProfessional:** Getter&Setter class which is used to store information about the health professional:
  - String title.
  - String firstName.
  - String lastName.
  - String specialty.
  - String clinic.
  - String telephoneHome.
  - String mobileHome.
  - String email

- **InformalCaregiver:** Getter&Setter class which is used to store information about the informal caregiver:
  - String title.
  - String firstName.
  - String lastName.
  - String relationship.
  - String telephoneHome.
  - String mobileHome.
  - String email
- **User:** Getter&Setter class which is used to store information about the patient:
  - String title.
  - String firstName.
  - String lastName.
  - String sex.
  - String sexDescription.
  - Int yearsOld.
  - String telephoneNumber.
  - String addressStreet.
  - String addressStreetNo.
  - String city.
  - String postcode.
  - String country.

6.6.4.4 Operations

Name	Description	Inputs	Outputs
sendManualAlert	This operation is used to send Manual alerts from the user application	String idUser, int code, long timestamp	boolean
sendAutomaticAlertHR_BR_AL	Automatic Alert related to the Heart rate, respiration rate and/or activity level	String idUser, VitalSign vitalSign, int code, long timestamp	Boolean
sendAutomaticAlertBloodPressure	Automatic Alert related to the Blood Pressure	String idUser, BloodPressure bloodPressure,int code, long timestamp	Boolean
sendAutomaticAlertTemperature	Automatic Alert related to the Temperature	String idUser, Float temperature, int code, long timestamp	Boolean
sendAutomaticAlertBMI	Automatic Alert related to the BMI	String idUser, float bmi, int code, long timestamp	Boolean
sendAutomaticAlertFall	Automatic Alert related to a fall	String idUser, int code, long timestamp	Boolean

Table 6-38. Emergency Management service operations.

6.7 COF SERVICE

The Common Ontological Framework (COF) is a core component of the REMOTE architecture that provides a semantics-aware infrastructure for the discovery and invocation of Web services. It also allows smooth integration of applications by means of software agents. It serves as an ontology-driven middleware infrastructure which receives requests for services and returns the required content. A COF service has been developed in order to be mainly invoked by the software agents through the AmI API. In this way, COF makes sure that service invocation from within the different integrated applications

occurs in an application-transparent way. The following functionalities are supported by the COF service.

- ***View ideal operations in the ontology***: Application developers may request a list of all available ideal operations that are defined in the ontologies, as well as relevant metadata that describe the various interconnected web services.
- ***Upload ontologies***: Service providers may create and upload their ontologies on COF.
- ***Invoke web services***: The COF service may receive requests for invocation of aligned web services.

The COF web service has been developed to allow access to the basic functionalities provided by COF. The service details are presented in what follows.

6.7.1 Services Description

Name	COF service
Description	Management of COF related functionalities mainly accessed by the AmI API. These functionalities include: view and upload ontologies, view aligned services and invoke operations of the aligned services.
Actors	AmI API (agents) who request particular information from COF
Related UCs	All UCs which involve the invocation of aligned web services.
Inputs	Requests for service invocation or for viewing the contents of the ontology
Outputs	Lists of ideal or real aligned web service operations. List of output objects when an aligned operations is invoked.

Table 6-39. COF service description.

6.7.1.1 Data Storage

The COF database contains information about the aligned and ideal operations that are stored in the ontology. The db schema is illustrated in the following figure.

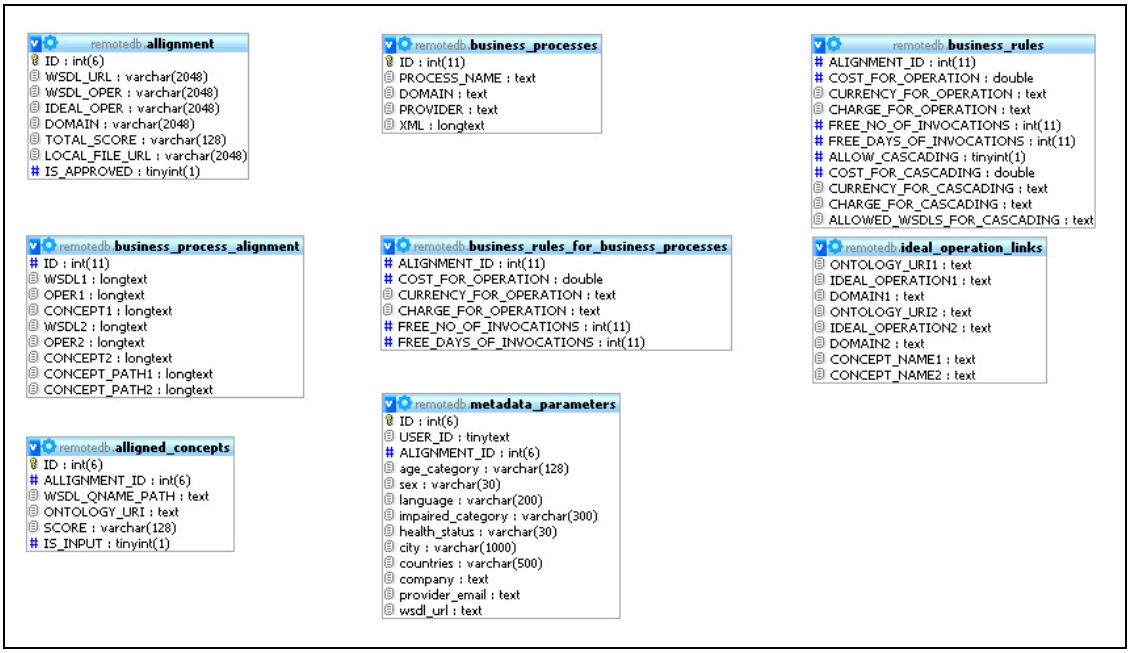


Figure 6-12. COF data base schema

6.7.1.2 Classes

In order to support the different COF functionalities, the following new classes are defined (complex objects):

- **WSOperationInput:** this class is used to store the input parameters of a web service operation. It includes the following fields:
  - *Vector hasNativeOrComplexObjects*: A vector containing ComplexObject or NativeObject objects used for storing the inputs of an operation.
  - *String hasUse*: String with two possible values, encoded or literal, according to the wsdl's specification
  - *Vector hasSoapHeaders*: A vector containing ComplexObject or NativeObject objects used as soap headers in the web service operation
- **WSOperationOutput**, this class is used to store the output parameters of a web service operation.
  - *Vector hasNativeOrComplexObjects*: A vector containing ComplexObject or NativeObject objects used for storing the outputs of an operation.
  - *Vector hasFaultMessageObjects*: A vector containing possible errors occurred during the invocation of the operation
  - *Vector hasSoapHeaders*: A vector containing ComplexObject or NativeObject objects used as soap headers in the web service operation
- **commonMetadata**, this class is used to store the metadata information of a web service operation.
  - *String USER_ID*: The ID of the service developer that aligned the operation
  - *String ALIGNMENT_ID*: The ID of the alignment

- *String age_category*: The age category that the operation is intended to.
- *String sex*: The sex that the operation is intended to
- *String language*: The languages that are supported by the current operation
- *String impaired_category*: The impaired category that the operation is intended to
- *String health_status*: The health status of elderly that the operation is intended to
- *String city*: The cities that the operation is designed to
- *String countries*: The countries that the operation is designed to
- *BusinessRules businessRules*: The business rules of the current operation
- *String company*: The company that has provided the service
- *String providerEmail*: The email of the service developer that aligned the service
- *String wsdlUrl*: The wsdl url of the online web service

- **BusinessRules**, this class is used to store the business rules of a web service operation.
  - *Cost hasCost*: The cost associated with this web service operation
  - *int freeAllowedNoOfInvocations*: The number of free invocations that the service provider has allowed for each user
  - *int freeAllowedDaysOfInvocation*: The number of free days for invocation that the service provider has allowed for each user
  - *boolean allowCascading*: A boolean indicating if the current operation can be included into a business process
  - *List allowedProvidersForCascading*: A list containing the user names of the service providers that can use this operation in a business process
  - *Cost hasCostForCascading*: The associated cost for using this operation in a business process
- **Cost**, this class is used to store associated costs for a web service operation.
  - *double cost*: The total amount of money that should charged
  - *enum currency*: Enumeration (with possible values EUR, USD, GBP) indicating the currency of the charge
  - *enum chargeEnum*: Enumeration (with possible values PER_INVOCATION, PER_MONTH, PER_YEAR, UNLIMITED) indicating how the charge will be performed

6.7.1.3 Operations

Name	Description	Inputs	Outputs
`GetDomains`	This web service operation returns a list with all the available domains along with their Preferred UI Names	-	List domains

GetIdealOperations	This web service method returns a list with the names of all the available ideal operations along with their Preferred UI Names, of a specific domain	String domain	List idealOperations
GetAllIdealOperations	This web service method returns a list of the names of all the available ideal operations along with their Preferred UI Names, of all available domains.	--	List idealOperations
getAlignedOperations	This web service method returns a list with all the aligned operations to an ideal operation of a specific domain.	- String idealOper - String domain	List with triples <id, Name, Alignment Score, isBusinessProcess>
getCommonMetadata	This web service method returns an object containing all the metadata parameters for a specific aligned operation.	- int id	common Metadata object
getCommonMetaDataOfBusinessProcess	This web service method returns all the metadata parameters of the operations involved in a business process	- int id - String name	List with common Metadata objects
getInputForOperation	This web service method returns an aligned web service/business process input object.	- String Oper - String Domain – int id	WSOperationInput object
getOutputForOperation	This web service method returns an aligned web service/business process output.	- String Oper - String Domain – int id	WSOperationOutput object
invokeOperation	This web service method invokes an aligned operation/business process and returns the invocation output.	- String Oper - String Domain – int id - WSOperationInput input	WSOperationOutput object

UpdateOntology	This web service method checks the COF for new ontologies, and downloads them.	-	True if the ontology updated, false otherwise

Table 6-40. COF operations.

7 COMPONENT IDENTIFICATION AND SPECIFICATION

REMOTE components are defined as modules that encapsulate a set functionalities or services. When a component offers services to the whole system it is usually carried out through the AmI, specifying the services and operations that other components can utilize, and how they can do so.

One of the most interesting attributes that offers the components is that they are substitutable, it means that a component can replace another without breaking the system in which the component operates.

This document is not going to show the strict meaning of a software component where it usually shows a declaration component in XML with all the properties: implementation, references, interfaces. The objective of this section is to translate the concept of components to the REMOTE platform philosophy and to show the most important components of the architecture.

The list of REMOTE components is the following:

- PC/Mobile Main Menu.
- PC applications.
- Mobile applications.
- AmI.
- User Profile System.
- DDBB servers.
- Medical Contact Centre.
- Professional/Carer Web Based application.
- Sensors.

7.1 PC/MOBILE MAIN MENU

7.1.1 Description

Integration framework where PC/Mobile application/services are installed and registered in a common user interface.
NOTE: this component is described in detail in D6.4, section 3.3 Main Menu.

7.1.2 Services

- Provided services:
  - Main Menu registration internal service
  - Emergency Management service (Panic Button)

- Requested services:
  - Medical contact service authentication
  - User profile

7.2 PC APPLICATIONS

7.2.1 Description

Java desktop applications based on OSGI and addressed to end user. PC applications are provided to the user through the Main Menu.

NOTE: PC Applications are described in detail in D4.1.

7.2.2 Services

- Provided services:
  - Nutritional Advisor (monitoring).
  - Activity Advisor (monitoring).
  - Brain Skills trainer (monitoring)
  - Calendar (monitoring)
  - Environmental control management.

- Requested services:
  - BPM-Activity Plan
  - BPM-Nutrition Plan
  - BPM-Patient Medication Plan
  - BPM-Training Plan
  - BPM-Trip Plan.
  - User Management Service information.

7.3 MOBILE APPLICATIONS

7.3.1 Description

Symbian applications based on OSGI and addressed to end user. Symbian applications are provided to the user through the Main Menu and they call external services through the AmI.

NOTE: Mobile Applications are described in detail in D4.2

7.3.2 Services

- Provided services:
  - Nutritional Advisor (monitoring).
  - Activity Advisor (monitoring).
  - Brain Skills trainer (monitoring)
  - Calendar (monitoring)
  - Environmental control management.
  - Vital sign monitoring
  - Trip Advisor / Guardian Angel / Fall protection

- Requested services:
  - BPM-Activity Plan
  - BPM-Nutrition Plan
  - BPM-Patient Medication Plan
  - BPM-Training Plan
  - BPM-Trip Plan
  - User Management Service information.

7.4 AmI

7.4.1 Description

Centralized system where all the REMOTE operations/services are aligned and ready to be provided to clients.

NOTE: AmI system is described in detail in D2.2.

7.4.2 Services

- Provided services:
  - Vital Sings Monitoring
  - User profile building
  - Health Advisor
  - Nutritional Advisor
  - Activity Advisor
  - Trip Advisor
  - Personal Calendar
  - Brain and Skills Trainer
  - Environmental Control at Home.
  - Fall protection.
  - Patient Activity Monitoring
  - Patient Nutrition Monitoring
  - Patient Records Monitoring
  - Care Planning Assistance Service
  - Emergency Management Service
  - Users Management Service

- Requested services:
  - Vital Sings Monitoring
  - User profile building
  - Health Advisor
  - Nutritional Advisor
  - Activity Advisor
  - Trip Advisor
  - Personal Calendar
  - Brain and Skills Trainer

- Environmental Control at Home.
- Fall protection.
- Patient Activity Monitoring
- Patient Nutrition Monitoring
- Patient Records Monitoring
- Care Planning Assistance Service
- Emergency Management Service
- Users Management Service

7.5 USER PROFILE SYSTEM

7.5.1 Description

XML user data base system fed by the Medical Contact Center and consumed by all the PC/Mobile applications.

NOTE: User profile system is described in detail in D2.1.

7.5.2 Services

- Provided services:
  - User profile building

- Requested services:
  - Medical contact service

7.6 DDBB SERVERS

7.6.1 Description

Servers that provide services to manage the information related to specific REMOTE services databases.

NOTE: DDBB servers system is described in detail in D5.1,D4.1 and D5.2.

7.6.2 Services

- Provided services:
  - Vital Sings Monitoring
  - Health Advisor
  - Nutritional Advisor
  - Activity Advisor
  - Trip Advisor
  - Personal Calendar
  - Brain and Skills Trainer

- Environmental Control at Home.
- Fall protection.
- Patient Activity Monitoring
- Patient Nutrition Monitoring
- Patient Records Monitoring
- Care Planning Assistance Service
- Emergency Management Service
- Users Management Service

- Requested services:
  - Vital Sings Monitoring
  - Health Advisor
  - Nutritional Advisor
  - Activity Advisor
  - Trip Advisor
  - Personal Calendar
  - Brain and Skills Trainer
  - Environmental Control at Home.
  - Fall protection.
  - Patient Activity Monitoring
  - Patient Nutrition Monitoring
  - Patient Records Monitoring
  - Care Planning Assistance Service
  - Emergency Management Service
  - Users Management Service

7.7 MEDICAL CONTACT CENTRE

7.7.1 Description

REMOTE management system which allows the user to manage REMOTE end users, professionals, carers and services information.

NOTE: Medical contact centre is described in detail in D5.2.

7.7.2 Services

- Provided services:
  - Medical contact service authentication
  - Vital Sings Monitoring
  - Health Advisor
  - Nutritional Advisor
  - Activity Advisor
  - Trip Advisor
  - Personal Calendar
  - Brain and Skills Trainer
  - Environmental Control at Home.

- Fall protection.
- Emergency Management Service
- Users Management Service

7.8 PROFESSIONAL/CARER WEB BASED APPLICATIONS

7.8.1 Description

Web based applications for professionals which retrieves information from its own database and from services across the platform.

7.8.2 Services

- Provided services:
  - BPM-Activity Plan
  - BPM-Nutrition Plan
  - BPM-Patient Medication Plan
  - BPM-Training Plan
  - BPM-Trip Plan
  - Cognitive Problem Prognosis
  - Decision Support Tool
  - Patient Activity Monitoring
  - Patient Nutrition Monitoring
  - Patient Records Monitoring

- Requested services:
  - Nutritional Advisor
  - Activity Advisor
  - Trip Advisor
  - Personal Calendar
  - Brain and Skills Trainer
  - Environmental Control at Home.
  - Fall protection.
  - Patient Activity Monitoring
  - Patient Nutrition Monitoring
  - Patient Records Monitoring
  - Care Planning Assistance Service
  - Emergency Management Service
  - Users Management Service

7.9 SENSORS

7.9.1 Description

Domotic and wearable sensors that usually provide information on demand.

NOTE: Sensors systems are described in detail in D3.1 and D4.2.

7.9.2 Services

- Provided services:
  - Fall Protection / Guardian Angel / Trip advisor
  - Vital Sings Monitoring

.

7.10 SUMMARY

The following table summarizes the most important information described before:

Components	Description
PC/Mobile Main Menu	• Integration framework where PC/Mobile application/services are installed and registered in a common user interface. • User Authentication service consumer. • User profile service consumer.
PC applications	• Java application based on OSGI. • Application is provided to the user through the Main Menu. • REMOTE philosophy requires consuming services from other components through the AmI.
Mobile applications	• Symbian application. • Application is provided to the user through the Main Menu. • REMOTE philosophy requires consuming services from other components through the AmI.
AmI	• Centralized system where all the REMOTE operations/services are aligned and ready to be provided to clients. • Web services are published through the AmI. • PC/Mobile applications use the AmI to consume services.
User Profile System	• XML user data base system fed by the Medical Contact Center and consumed by all the PC/Mobile applications.
DDBB servers	• Servers that provide some services to manage the information related to specific databases.

Medical Contact Centre	• REMOTE management system: users, professionals, carers, services. • User Profile system feeder. • Authentication mechanism. • REMOTE User information provider to the rest of application.
Professional/Carer Web Based application	• Web interface which retrieves information from its own database and from services across the platform.
Sensors	• Domotic and wearable sensors that usually provide information on demand.

7-1. REMOTE Components

Services that were described in the last section are provided to the platform through the components shown in the table above.

8 SERVICE MODEL

This section is a summary of the preceding sections it shows a model diagram with the full REMOTE services.

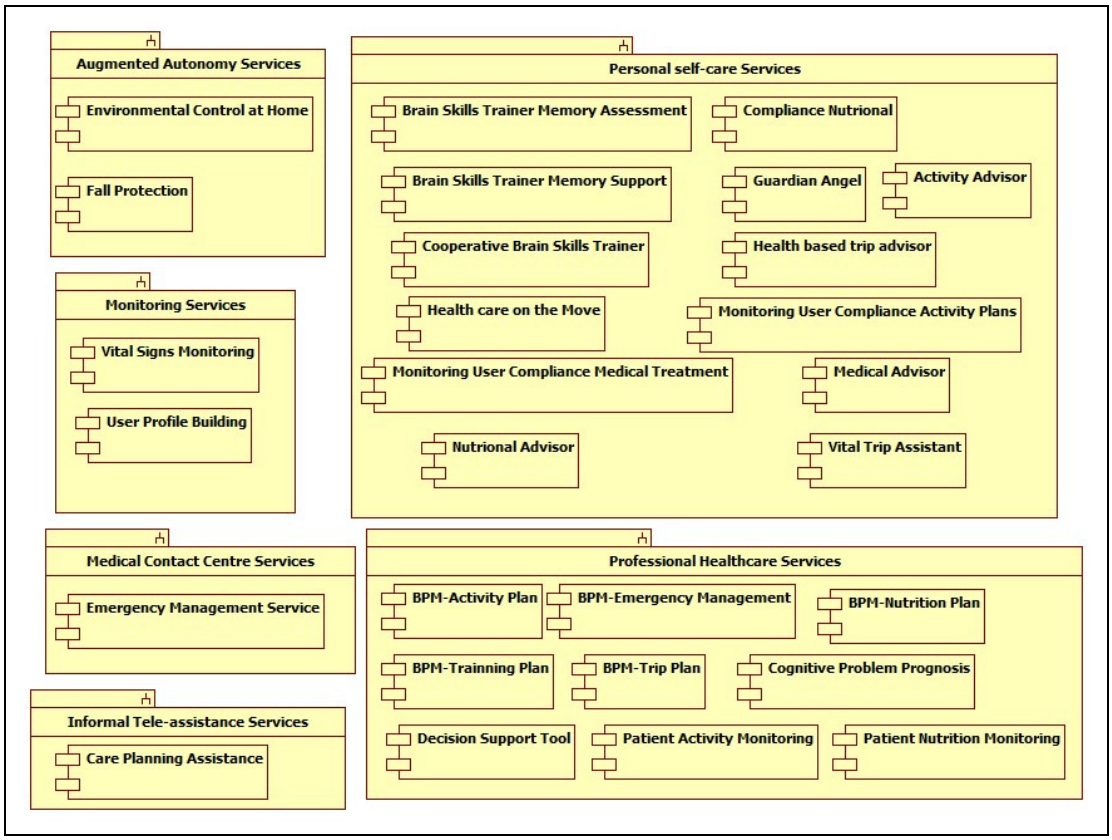


Figure 8-1 Service Model

9 CONCLUSIONS

This document described the REMOTE Service and component identification and specification, focusing on the supported services, user needs and requirements. Firstly, in Section 2, it provides a first insight on the Services Architecture. In Section 3 is described the domain decomposition (from one of three techniques for services identification of SOMA methodology). The Goal Service Modelling (other technique for services identification of SOMA methodology) is described in detail in section 4, while in Section 5 is described the existing asset analysis. Section 6 provides a detailed description about the Services Identification. In Section 7 the Service Model is described and in Section 8 is detailed the description about the Services Specification.

The Remote domain can be decomposed into functional areas across the value-net. This functional areas identified in the Remote architecture are: the End-user applications, the Device and Sensors, the Remote software modules, Ontology and Services.

The business processes are good candidates for services. The business processes are: elaborate Nutrition Plan, elaborate Training Plan, elaborate Trip Plan, elaborate Patient Medication Plan and elaborate Activity Plan.

The business use cases can be used to decompose the domain. In Remote, several use cases, in different functional areas, which are candidates to become Web Services can be identified: Monitoring Services, Personal self-care services, augmented autonomy services, Professional healthcare services, Informal tele-assistance services, Medical Contact Centre.

In SOMA methodology, another technique for services identification is through the goal service modelling. In this step we create a goal-service model through business process identification and business goals identification. Some businesses Goals are: to reduce health spending (reduce waiting in health care), and to improve the quality of life of elderly and physically disabled (access to rural areas, patients monitored 24/7 reducing response times in emergency patients).

Service identification:

- *Monitoring*: Vital Sings Monitoring and Vital Sings Monitoring.
- *Personal Self-care*: Health Advisor, Nutritional Advisor, Activity Advisor, Trip Advisor, Personal Calendar, Brain and Skills Trainer
- *Augmented Autonomy*: Environmental Control at Home, Fall Protection.
- *Professional Health Care*: BPM-Activity Plan, BPM-Nutrition Plan, BPM-Patient Medication Plan, BPM-Training Plan, BPM-Trip Plan, Cognitive Problem Prognosis, Decision Support Tool, Patient Activity Monitoring, Patient Nutrition Monitoring, Patient Records Monitoring.
- *Informal Tele-Assistance*: Care Planning Assistance Service.
- *Medical Contact Centre*: Emergency Management Service.

REFERENCES

[1]. REMOTE-DoW-July2010.doc. Work packages Description.
[2]. REMOTE_D1.1_v1.0.doc. Definition of REMOTE user requirements and use cases. (WP1).
[3]. REMOTE_D6.2_v1.0.doc. Integration Platform Architecture (WP6).
[4]. Service-oriented modelling and architecture. How to identify, specify, and realize services for your SOA (09 Nov 2004). IBM. Available at: http://www.ibm.com/developerworks/library/ws-soa-design1/
[5]. RUP Methodology. IBM. http://www-01.ibm.com/software/awdtools/rup/
[6]. R. Paradiso, G. Loriga, N. Taccini, "A Wearable Health Care System based on Knitted Integrated Sensors". IEEE Transaction Technology in Biomedicine, vol 9 (3), pp.337-345, 2005
[7]. G. Loriga, N. Taccini , M. Pacelli and R. Paradiso, From Mind to Market, Fall 2007, IEE Industrial Electronic Magazine