iCARDEA Project: Personalized Adaptive Care Planner

Software Detailed Design Document
Version 1.0.0

ANTIQUE COWS

- Cihan Çimen - 1560689
- Elif Eryılmaz - 1560200
- Emine Karaduman - 1560317
- Ozan Çağrı Tonkal - 1560598

Sponsored by Software Research and Development Consultancy (SRDC)

iCARDEA

An Intelligent Platform for Personalized Remote Monitoring of the Cardiac Patients with Electronic Implant Devices
6. User Interface Design

6.1. Overview of User Interface

6.1.1. User Interface of Careplan Editor Subsystem

6.1.1.1. Login View ............................................................... 33
6.1.1.2. Medical Careplan Repository View ................................. 33
6.1.1.3. Medical Careplan Editing View ...................................... 33
6.1.1.4. Decision Step Editing View ......................................... 33
6.1.1.5. Consult Step Editing View .......................................... 34
6.1.1.6. Recommendation Step Editing View ............................... 34
6.1.1.7. HTML Editing View ................................................... 34
6.1.1.8. CIED/EHR/PHR Data Provider View ......................... 34
6.1.2. User Interface of Careplan Engine Subsystem ..................... 34
6.1.2.1 Assignment View .......................................................... 34
6.1.2.2 Execution View ........................................................... 34
6.1.3. User Interface of Alarm Manager Subsystem ...................... 35
6.1.3.1 Login View ............................................................... 35
6.1.3.2 Alarm Logs View ......................................................... 35
6.1.3.3 Configuration View ..................................................... 35
6.1.4. User Interface of Monitoring Tool ..................................... 35
6.1.4.1 Login View ............................................................... 35
6.1.4.2 Monitoring View .......................................................... 35

6.2. Screen Images ..................................................................... 36
6.2.1. Careplan Editor Subsystem .............................................. 36
6.2.1.1. Login View ............................................................... 36
6.2.1.2. Medical Careplan Repository View ................................. 36
6.2.1.3. Medical Careplan Editing View ...................................... 37
6.2.1.4. Decision Step Editing View ......................................... 38
6.2.1.5. Consult Step Editing View .......................................... 39
6.2.1.6. Recommendation Step Editing View ............................... 39
6.2.1.7. HTML Editing View ................................................... 40
6.2.1.8 CIED/EHR/PHR Data Provider View ............................ 42

Figure 33: CIED/EHR/PHR Data Provider View ............................ 43

6.2.2 Careplan Engine Subsystem .............................................. 43
6.2.2.1 Assignment View .......................................................... 43
6.2.2.2 Execution View ........................................................... 44

6.3. Screen Objects and Actions ................................................. 44
6.3.1. Careplan Editor Subsystem .............................................. 44
6.3.1.1. Login View Object and Actions ...................................... 44
6.3.1.2. Medical Careplan Repository View ................................................................. 45
6.3.1.3. Medical Careplan Editing View ................................................................. 45
6.3.1.4. Decision Step Editing View ........................................................................... 45
6.3.1.5. Consult Step Editing View ............................................................................. 46
6.3.1.6. Recommendation Step Editing View .......................................................... 46
6.3.1.7. HTML Editing View ....................................................................................... 46
6.3.1.8. CIED/EHR/PHR Data Provider View ......................................................... 46
6.3.2 Careplan Engine Subsystem ................................................................................ 47
  6.3.2.1 Assignment View .......................................................................................... 47
  6.3.2.2 Execution View ............................................................................................ 47
6.3.3 Alarm Manager Subsystem .................................................................................. 47
  6.3.3.1 Login View .................................................................................................... 47
  6.3.3.2 Alarm Logs View .......................................................................................... 47
  6.3.3.3 Configuration View ....................................................................................... 48
6.3.4. Monitoring Tool Subsystem ............................................................................. 48
  6.3.4.1 Login View .................................................................................................... 48
  6.3.4.2 Monitoring View .......................................................................................... 48
7. Detailed Design ......................................................................................................... 48
  7.1. Careplan Editor Subsystem .................................................................................. 48
    7.1.1. Structural Design .......................................................................................... 48
    7.1.2. Behavioral Diagrams .................................................................................... 50
      Figure 38: State Transition Diagram for Careplan Editor Subsystem .................. 51
    7.1.3. Interaction Diagrams .................................................................................... 51
      Figure 40: Interaction Diagram for Management of Existing Medical Careplan Templates 53
      Figure 42: Interaction Diagram for Generation of Personalized Medical Careplan (PMCP) 54
      Figure 43: Interaction Diagram for Modification of Personalized Medical Careplan (PMCP) 55
  7.2. Careplan Engine Subsystem ................................................................................ 55
    7.2.1. Behavioral Diagram ...................................................................................... 55
      Figure 44: State Transition Diagram for Careplan Engine Subsystem .................. 56
    7.2.2. Interaction Diagrams .................................................................................... 56
      Figure 45: Interaction Diagram for Provision of Medical services specified in Medical Careplan Definitions 57
      Figure 46: Interaction Diagram for Subscription to related Clinical Data Sources and bind Medical Operation Services 58
  7.3. Monitoring Tool Subsystem .............................................................................. 59
    7.3.1. Behavioral Diagram ...................................................................................... 59
    7.3.2. Interaction Diagrams .................................................................................... 60
      Figure 49: Interaction Diagram for Retrieval of Monitoring Messages .................. 60
      Figure 50: Interaction Diagram for Monitoring of Personalized Medical Careplans (PMCPs) assigned to Patients 61
  7.4. Alarm Manager Subsystem .............................................................................. 61
    7.4.1. Structural Design .......................................................................................... 62
    7.4.2. Behavioral Diagram ...................................................................................... 63
      Figure 54: State Transition Diagram for Alarm Manager Subsystem .................. 64
    7.4.3. Interaction Diagrams .................................................................................... 64
      Figure 57: Interaction Diagram for Management of Personal & Contact Information 66
8. Libraries and Tools .................................................................................................. 66
9. Time Planning (Gantt Chart) .................................................................................. 68
9.1. Term 1 Gantt Chart ........................................................................................................70
9.2. Term 2 Gantt Chart ........................................................................................................70
10. Conclusion ........................................................................................................................70
1. Introduction

This document describes the conceptual design of the iCARDEA project according to the document guidelines presented in the IEEE-1016 1998 Recommended Practice for Software Design Descriptions (SDD).

1.1. Problem Definition

The iCARDEA Project will develop an intelligent platform to automate the follow-up of CIED patients with adaptable computer interpretable clinical guideline models which access data seamlessly in EHR data resources, CIED data and PHRs using standard interfaces. The overall architecture of the project is shown in Figure 1. The computer interpretable guideline models to be developed will be adaptable, designed from re-usable building blocks to easily personalize the patient and device follow-up. Then, these guideline models will be converted to executable clinical work flows which will perform the follow-up activities and automate the risk assessment via integrative models and hence support medical professionals by automatically assessing the situations and generating alarms.
SRDC (Software Research & Development and Consultancy) is the administrative and scientific coordinator of the project. SRDC will issue action plans of six month periods for effective management of the project tasks, deliverables and deadlines. SRDC Team has extensive experience in project management and technically in the area of eHealth. In the iCARDEA project, SRDC will first coordinate the conceptual architecture of iCARDEA. The major development activity will be the CIED based Personalized Adaptive Care Planner. Additionally, SRDC is involved in the development of consent management system for PHR, interoperability infrastructures for EHRs and PHRs, code mapping API and security and privacy mechanisms for the mentioned components. Apart from being the coordinator, SRDC has to be well-informed of all the components developed by other partners since the Adaptive Care Planner will act as the integrated iCARDEA Platform. SRDC will have significant relationships directly with the end-users as the Adaptive Care Planner GUI and Engine will be used by them. End-users will share their expertise in clinical guidelines and help SRDC in interpreting them; in return SRDC will share its expertise in modeling of the guidelines for computer processing.

Our team “Antique Cows” is responsible for the CIED based Personalized Adaptive Care Planner which is the major development activity part of the project.

Personalized Adaptive Care Planner for the CIED Recipients: In the iCARDEA project, the personalized follow-up of CIED patients is coordinated through a “care plan” which is an executable definition of computer interpretable clinical guideline models. The care plans are represented in GLIF (Guideline Interchange Format), and the Care Plan Engine is capable of semi-automatically executing the care plan by processing its machine processable definition. The control flow of the care plan is dynamically adapted based on the patient’s context derived from
the data coming from CIEDs and the medical context obtained from the EHRs. Through a graphical monitoring tool, the physicians are allowed to follow the execution of the care plan in detail, and coordinate the flow of actions when consultations to physicians are required.

The subsystems of Adaptive Care Planner Engine Module of the iCARDEA platform are as follows:

- **Careplan Editor Subsystem**: This subsystem is responsible for the management of the Medical Careplan Template and Personalized Medical Careplan Definitions.

- **Careplan Engine Subsystem**: The subsystem is responsible to execute Personalized Medical Careplans. During the executions, the subsystem retrieves data from EHR, PHR and CIED and send alert messages to corresponding parties through Alarm Manager Subsystem. Furthermore, it also sends monitoring messages to Monitoring Tool Subsystem.

- **Alarm Manager Subsystem**: The subsystem delivers the alert messages generated by the Careplan Engine to intended persons.

- **Monitoring Tool Subsystem**: This subsystem provides information about the execution status of Personalized Medical Careplans.

**1.2. Purpose**

The purpose of this document is to stress the initial design process about the project. The requirements specified in the requirements analysis report will be explained in detail as the structural components which will be used in the implementation phase. It is intended to be used by the members of the Antique Cows team that will design, implement and verify the correct functioning of the system.

**1.3. Scope**

The SDD shows how the software system will be structured to satisfy the requirements identified in the software requirements specification. It is a translation of requirements into a description of the software structure, software components, interfaces, and data necessary for the implementation phase. In essence, the SDD becomes a detailed blueprint for the implementation activity. In a complete SDD, each requirement must be traceable to one or more design entities.

**1.4. Overview**
This document lists the initial design issues of the iCARDEA system. It mainly includes design considerations, data design, system architecture, user interface design and detailed design. This document also covers the libraries and tools used during the project. At the end, time planning of the team is described.

1.5. Definitions, Acronyms and Abbreviations

**CIED**: (Cardiovascular Implantable Electronic Device) is an implantable device used to treat a number of conditions for patients with cardiac arrhythmia abnormalities. CIED family consists of implantable pacemakers, defibrillators, devices for cardiac resynchronization therapy and cardiac monitors.

**EHR**: (Electronic Health Record) “a longitudinal electronic record of patient health information generated by one or more encounters in any care delivery setting. Included in this information are patient demographics, progress notes, problems, medications, vital signs, past medical history, immunizations, laboratory data and radiology reports”.

**IHE**: (Integrating the Healthcare Enterprise) initiative by healthcare professionals and industry vendors to improve the way healthcare computer systems share information. IHE promotes the coordinated use of established standards, such as DICOM and HL7, to address specific clinical needs in support of optimal patient care. Systems developed in accordance with IHE communicate with one another better, are easier to implement, and enable care providers to use information more effectively.

**PHR**: (Personal Health Record) “a complete and accurate summary of the health and medical history of an individual by gathering data from many sources and making this information accessible online to anyone who has the necessary electronic credentials to view the information” (Wikipedia).

1.5.1. Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation/Acronym</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ATNA</td>
<td>IHE Audit Trail and Node Authentication Profile</td>
</tr>
<tr>
<td>CIED</td>
<td>Cardiovascular Implementable Electronic Device</td>
</tr>
<tr>
<td>CIS</td>
<td>CIED Information System</td>
</tr>
</tbody>
</table>
CTS | Common Terminology Services  
--- | ---  
CUI | Concept Unique Identifier  
EHR | Electronic Health Record  
HL7 | Health Level Seven  
ICD | International Statistical Classification of Diseases and Related Health Problems  
IHE | Integrating Healthcare Enterprise  
IHE-CM | IHE Care Management Profile  
IHE-IDCO | IHE Implantable Device Cardiac Observation Profile  
MCP | Medical Careplan  
PHR | Personal Health Record  
PHRS | Personal Health Record System  
PMCP | Personalized Medical Careplan  
PPM | Patient Parameter Monitor  
UMLS | Unified Medical Language System

Table 1: List of Abbreviations and Acronyms

1.6. References

[1]  
Council of Europe – Committee of Ministers, Recommendation No. R(97)5 of The Committee Of Ministers to Member States on the Protection Of Medical Data, Council of Europe Publishing, Strasbourg, 12 February 1997

IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications


2. System Overview

The iCARDEA Project will develop an intelligent platform to automate the follow-up of the CIED patients with adaptable computer interpretable clinical guideline models which access data seamlessly in EHR data resources, CIED data and PHRs using standard interfaces. The overall architecture of the project is shown in Figure 1. The computer interpretable guideline models to be developed will be adaptable, designed from re-usable building blocks to easily personalize the patient and device follow-up. Then these guideline models will be converted to executable clinical work flows which will perform the follow-up activities and automate the risk assessment via integrative models and hence support medical professionals by automatically assessing the situations and generating alarms.

Figure 2: Overall vision of the iCARDEA Project
Figure 3 gives an overview of the iCARDEA components and their interaction. To achieve the objectives and to implement the services required by the pilot application, the following basic components will be provided by iCARDEA:

- An interoperability layer that exposes the sensor data and medical information system functionality as semantically harmonized and standardized data to tackle the problem of interoperability between different CIED-vendors, medical information systems and the patient empowerment framework.

- An intelligent personalized Adaptive Care Planner for CIED recipients based on semantically enriched, computer-interpretable guidelines.
• A Framework for patient empowerment, where patients can view or update their PHR data, access educative materials related to their problems, get involved in the treatment process by providing feedback about their health status and finally manage their healthcare through electronic privacy consents.

• A Tool for Context Awareness and Clinically Useful Information Derivation, which helps medical professionals to easily identify crucial parameters of a patient and makes suggestions based on data obtained from medical knowledge bases.

• A comprehensive infrastructure for security and privacy.

3. Design Considerations

The first goal we want to achieve before devising a complete design solution is portability. The system should be portable to Windows and Linux platforms. Therefore we decided to use Java since we need a highly platform-independent programming language. Also user interfaces of the all subsystems should be accessible with minimal burden to user. Thus all user interfaces of the tools are web-based and does not require any additional setup or installation but an HTML browser capable of running Adobe Flash Player[3].

3.1. Design Assumptions, Dependencies and Constraints

A database should include single data point information, graphs and trended data facilitate CIED and patient disease and comorbidity management. Additionally, a database should include both registration data and follow-up clinical data.

Crucial to the usefulness of a database is the accuracy of the data and manual data entry should be discouraged as it creates the opportunity for error. A database that directly communicates with the registration system would automatically update patient demographics and contact information, as well as correctly identify the CIED follow-up physician.

State of the art security is mandatory to maintain confidentiality for patients with facilitated access for health care providers to expedite patient care.

3.2. Design Goals and Guidelines

Personalized Adaptive Care Planner is suitable to every platform, the only need is an internet connection. Moreover, not only its response time, but also the use of memory is
important considering the performance. Actually, the system is time-critical with respect to the monitoring tool, but it is not that much time-critical with respect to the care planner engine.

Some of the design goals are the following:
- To maintain patient records and institutional data-bases.
- To receive the data information from PHR and EHR services in an accurate and efficient way.
- To monitor cardiac arrhythmias and physiologic parameters, communicating information related to CIED monitoring to involved physicians and other health care providers.
- To provide patient and family education and reassurance.

4. Data Design

4.1. Data Description

Data of the system can be partitioned in three groups:
- First part is the persistent data like Alarm Entity or Medical Careplan Entity which will be stored in a relational SQL database.
- Second part of the data is non-persistent data which will be retrieved by EHR, PHR or CIED data services when needed in careplan definition or execution phases.
- Since we have a rich client application. Server-side and client-side contains the same data at a time. To synchronize these two data models there is a messaging system between client and server that is triggered when a model in client side changes that is needed to be stored in server-side. Also there are some diagram representations that is created and modified on the fly and converted to some other format, but never stored in a database directly. All these models are utilized to overcome architectural constraints or constraints imposed by libraries we use. So they constitute third part of our data model.

4.1.1 Persistent Data Model

Figure 4 shows which entities are persisted in the database.

- MedicalCareplanTemplate: This entity represents a medical careplan template. It is not personalized and not instantiated.
- PersonalizedMedicalCareplan: A medical careplan template which is personalized for a specific patient by a healthcare actor.
- PersonalizedMedicalCareplanInstance: A medical careplan instance is an entity which contains data of a personalized medical careplan execution. Each execution of a personalized medical careplan results in a personalized medical careplan instance.
- Contact: Contact information of a Person or Organization.
- Organization: Entity that represents an organization.
- Person: Any user of the system.
- PatientHealthCareActorAssignment: This entity represents the relation between a patient and a health care actor. Each patient is assigned to a health care actor who supervises personalization or execution of a medical careplan.
- Patient: Represents a patient for whom a medical careplan is personalized and executed.
- Alarm: This entity is sent from patient to health care actor when any of the predefined criterion is satisfied.

Figure 4: Entity Relationship Diagram for persistent data model
4.2. Data Dictionary

In this section, the OO descriptions of the data objects used in the project are given.

There are 4 data types. These are indicated in the figures in between guillemets. The explanations of these types are as follow:

- Entity class models the information handled by the system, and sometimes the behaviour associated with the information.
- Boundary class is used to model the interaction between a system and its surroundings, i.e., its actors.
- ORM-Persistable class is capable of manipulating the persistent data with the relational database.
- Control class controls the transfer.

4.2.1. Acknowledgement Entity

Acknowledgement entity keeps the acknowledgement that is sent by the Healthcare Actor, who received an Alarm Message, through Alarm Manager GUI. It is an entity data object.

![Figure 5: Acknowledgement Entity Class Diagram](image)

4.2.2. Alarm Entity

Alarm entity describes an Alarm instance sent from a Patient to a Healthcare Actor. It is an entity data object.

![Figure 6: Alarm Entity Class Diagram](image)
4.2.3. AlarmManager

AlarmManager is used for delivering the Alert Messages generated by the CareplanEngine to the intended Persons. It is a control data object.

```
<<control>>
AlarmManager

+sendAlarm(alarm : Alarm)
+listPatients() : Patient[]
+listHealthcareActors() : Person[]
+assignHealthcareActorToPatient(assignment : PatientHealthcareActorAssignment)
+addHealthcareActor(actor : Person)
+login()
+updateHealthcareActor(actor : Person)
+retrievePMCPs(criteria : String) : PersonalizedMedicalCareplan[]
+updatePMCPConfiguration(pmcp : PersonalizedMedicalCareplan)
+sendAcknowledgement(ack : Acknowledgement)
+waitForAcknowledgement(ack : Acknowledgement)
```

Figure 7: AlarmManager Class Diagram

4.2.4. AlarmManagerGUI

AlarmManagerGUI is the graphical user interface through which the Healthcare Actor, who received an Alarm Message, sends an Acknowledgement. It is a boundary data object.

```
<<boundary>>
AlarmManagerGUI

+listPatients() : Patient[]
+listHealthcareActors() : Person[]
+assignHealthcareActorToPatient(assignment : PatientHealthcareActorAssignment)
+addHealthcareActor(actor : Person)
+login()
+editPersonalInformation(actor : Person)
+savePersonalInformation(actor : Person)
+retrieveLogs(criteria : String) : Log[]
+retrievePMCPs(criteria : String) : PersonalizedMedicalCareplan[]
+updatePMCPConfiguration(pmcp : PersonalizedMedicalCareplan)
+selectPMCP() : PersonalizedMedicalCareplan
+sendAcknowledgement(ack : Acknowledgement)
```

Figure 8: AlarmManagerGUI Class Diagram
4.2.5. CareplanEditor

CareplanEditor is used for both editing and managing the Medical Careplans. It is a control data object.

| <<control>> |
| CareplanEditor |

+constructTemplate() : MedicalCareplanTemplate  
+validateTemplate(mcp : MedicalCareplanTemplate)  
+askForApproval()  
+retrieveTemplates(criteria : String) : MedicalCareplanTemplate []  
+modifyTemplate() : MedicalCareplanTemplate  
+approveTemplate(approve : boolean, mcpID : String)  
+retrievePatients(criteria : String) : Patient []  
+linkEHREndpoint()  
+linkPHREndpoint()  
+linkCIEDEndpoint()  
+retrievePatientData(patientIdentifier : String) : Patient  
+constructPMCP()  
+validatePMCP()  
+retrievePMCPs(patientIdentifier : String) : PersonalizedMedicalCare...  
+finalizePMCP(pmcpid : String)

Figure 9: CareplanEditor Class Diagram
4.2.6. CareplanEditorGUI

CareplanEditorGUI is the graphical user interface of the CareplanEditor, as understood from its name. It is a boundary data object.

```
<<boundary>>
CareplanEditorGUI

+createNewTemplate()
+defineEligibilityCriteria()
+defineAlgorithm()
+addConsultStep()
+addDecisionStep()
+addRecommendationStep()
+saveTemplate()
+askForApproval()
+modifyTemplate()
+modifyEligibilityCriteria()
+modifyAlgorithm()
+modifyConsultStep()
+modifyDecisionStep()
+modifyRecommendationStep()
+selectTemplate() : MedicalCareplanTemplate
+retrieveTemplates(criteria : String) : MedicalCareplanTemplate []
+approveTemplate(approve : boolean)
+retrievePatients(criteria : String) : Patient []
+linkEHREndpoint()
+linkPHREndpoint()
+linkCDEEndpoint()
+retrievePatientData(patientIdentifier : String) : Patient
+personalizeMCP()
+saveMCP()
+retrieveMCPs(patientIdentifier : String) : PersonalizedMedicalCareplan []
+modifyMCP()
+finalizeMCP(pmcpid : String)
```

Figure 10: CareplanEditorGUI Class Diagram

4.2.7. Contact

Contact is the data object that holds the contact information of the user. It is an ORM-Persitent data object.
4.2.8. Log Entity

Log entity describes the log records of the Careplan Engine execution. It is an entity data object.

4.2.9. MCPRepository

MCPRepository keeps the MCPs and PMCPs. It is a control data object.
4.2.10. MCPRepositoryInterface

CareplanEditor constructs the PMCP and after validation it sends the PMCP to MCP Repository through MCPRepositoryInterface. It is a boundary data object.

4.2.11. MedicalCareplanTemplate

MedicalCareplanTemplate keeps a template for a Medican Careplan, as understood from its name. It is an ORM-Persitable data object.
4.2.12. MonitoringMessage Entity

MonitoringMessage entity holds the Monitoring Message which is sent by the Care Plan Engine to Careplan Monitoring Tool through Careplan Monitoring Tool Interface, and in return Careplan Monitoring Tool processes this Monitoring Message and displays the effects on its Careplan Monitoring Tool GUI. It is an entity data object.

4.2.13. Organization

Organization is the ORM Persistable data object that holds the organization information of the user.
4.2.14. Person

Person holds the all information about the present Person. It is an ORM-Persistable data object.
4.2.15. PersonalizedMedicalCareplan

PersonalizedMedicalCareplan describes a Medical Careplan which is personalized by a Healthcare Actor. It is an ORM-Persistable data object.

```
<<ORM Persistable>>
PersonalizedMedicalCareplan
-ID : int
-content : String
-identifier : String
-creationDate : Timestamp
-name : String
-patientIdentifier : String
```

Figure 19: PersonalizedMedicalCareplan Class Diagram

5. System Architecture

The system architecture of Personalized Adaptive Care Planner is analysed in this part of the document.

5.1. Architectural Design

This section describes the decomposition of iCARDEA modules into subsystems and packages. It describes the way the system has been structured and the purpose and function of each entity. The subsystems that are constructed as a result of the Architectural Design of iCARDEA are described with their responsibilities and design classes in this section, but actually the detailed analysis of the subsystems is given in the next section. The interfaces provided by subsystems are also presented in the sixth part of the document.
5.1.1. Adaptive Care Planner Engine

Figure 20: Adaptive Care Planner Engine Subsystems

Figure 20 displays the subsystems identified for the Adaptive Care Planner Engine Module of the iCARDEA platform. The subsystems are as follows:

- **Careplan Editor Subsystem**: This subsystem is responsible for the management of the Medical Careplan Template and Personalized Medical Careplan Definitions.

- **Careplan Engine Subsystem**: The subsystem is responsible to execute Personalized Medical Careplans. During the executions, the subsystem retrieves data from EHR, PHR and CIED and send alert messages to corresponding parties through Alarm Manager Subsystem. Furthermore, it also sends monitoring messages to Monitoring Tool Subsystem.

- **Alarm Manager Subsystem**: The subsystem delivers the alert messages generated by the Careplan Engine to intended persons.

- **Monitoring Tool Subsystem**: This subsystem provides information about the execution status of Personalized Medical Careplans.
5.2. Description of Components

5.2.1. Careplan Editor Subsystem

5.2.1.1. Processing Narrative

Careplan Editor subsystem is responsible for the management of the medical careplans (MCPs). It is responsible for providing functionalities to create, modify and personalize medical careplan templates.
5.2.1.2. Interface Description

The interface of the subsystem is a graphical interface and the following functionalities are provided by the Careplan Editor Subsystem to the users (Figure 22):
- Creation/Modification of new templates
  - Creation/Modification of eligibility criteria
  - Creation/Modification of careplan algorithm
  - Creation/Modification of action step
  - Creation/Modification of decision step
  - Creation/Modification of patient state step
  - Creation/Modification of branch step
  - Creation/Modification of synchronization step
- Saving/obtaining the template to/from MCP Repository
- Asking for approval
- Retrieval/Display of the patient medical data
- Linking the EHR, PHR and CIED endpoints in a template
- Personalizing a MCP template and generate a PCMP
- Modifying an existing PCMP

![Component Diagram of CareplanEditorSubsystem](image)

Figure 22: Component Diagram of CareplanEditorSubsystem

5.2.1.3. Processing Detail

The Careplan Editor subsystem creates new medical careplan template, saves it in medical careplan repository or retrieves medical careplan from the medical careplan repository and perform the necessary operations on them when requested.

5.2.1.4. Dynamic Behaviour

As shown in Figure 22, the interface of the Careplan Editor Subsystem is fulfilled by the CareplanEditorGUI class. The subsystem generates/consumes the classes in the Careplan Editor Model package and uses the MCP Repository Package for storing these model classes persistently. There are two main classes in the Careplan Editor Model package: MedicalCareplan for MCPs and PCMP for personalized medical care plans.
5.2.2. Careplan Engine Subsystem

The diagram is presented in Figure 23.

5.2.2.1. Processing Narrative

The Careplan Engine subsystem is responsible for executing personalized medical careplans.

5.2.2.2. Interface Description

The interface of the subsystem is realized by the CareplanEngineGUI class and it is a graphical user interface for human users. The functionalities provided by the subsystem is as follows (Figure 24):
   a. Retrieval/Display of PMCPs
   b. Retrieval/Display of the patients
   c. Initiation of the execution of PMCPs
   d. Execution of PMCPs
   e. Sending Monitoring Messages to Monitoring Tool Subsystem

5.2.2.3. Processing Detail

The Careplan Engine subsystem retrieves patients, select a specific patient, executes the PMCP (Personalized Medical Careplan). In return, a PMCP Instance is created.

5.2.2.4. Dynamic Behaviour

During the course of the execution necessary Alarm and Monitoring Messages are sent to Alarm Manager and Care Plan Monitoring Tool, respectively. Furthermore, required clinical data about the Patient is retrieved from Care Management DB and Web services of HIS is invoked by using Medical Service Interface.

5.2.3. Monitoring Tool Subsystem

The diagram is presented in Figure 23.

5.2.3.1. Processing Narrative

The Monitoring Tool subsystem is responsible for monitoring the execution of a personalized medical careplan. It also interacts with the users and directs the execution according to the user decisions (Consult messages).
5.2.3.2. Interface Description

The interface of the subsystem is realized through CareplanMonitoringToolInterface. Furthermore, CareplanMonitoringGUI displays the monitoring messages to the users. The functionalities provided by the subsystem is as follows (Figure 24):
   a. Retrieval of monitoring messages
   b. Displaying monitoring messages
   c. Tracing the logs of a executed/terminated PMCP instance
   d. Monitoring the execution of PMCP instance

5.2.3.3. Processing Detail

The Careplan Engine sends Monitoring Messages to Careplan Monitoring Tool through Careplan Monitoring Tool Interface and in return Careplan Monitoring Tool processes the Monitoring Message and displays the effects on its Careplan Monitoring Tool GUI.

5.2.3.4. Dynamic Behaviour

- The Careplan Engine generates/consumes classes from Model package and uses MCP Repository Package to obtain PMCP Instances.
- The Careplan Engine uses CareplanMonitoringToolInterface to realize the interface of the subsystem.
- The CareplanMonitoringToolInterface accepts monitoring messages from Careplan Engine Subsystem and the CareplanMonitoringGUI.
5.2.4. Alarm Manager Subsystem

The Alarm Manager subsystem is responsible for sending alarm/alert messages to related/responsible persons in predefined cases.

5.2.4.2. Interface Description

The subsystem’s interface is handled by the AlarmManagerInterface. Moreover, AlarmManagerGUI presents a graphical user interface to the user in order them to mainly specify alert conditions. The functionalities provided by the subsystem is as follows:

a. Retrieval of alarm messages
b. Assign patients to healthcare actors
c. Inspecting the logs
5.2.4.3. Processing Detail

Alarm Manager receives the alarm message from the executing Personalized Medical Careplan (PMCP) instance. Then it gathers data and rules for the received alarm message, and runs the rules to structure the outgoing messages. Finally it sends out the alert/alarm messages.

5.2.4.4. Dynamic Behaviour

- AlarmManagerInterface accepts alarm messages from Careplan Engine Subsystem and sends the messages to related persons and AlarmManagerGUI.
- Alarm Manager Subsystem generates/consumes classes from Model package.

5.3. Design Rationale

Main criterion used in decomposition of the system is in which aspect the task of the subsystem is critical. For example, tasks of Alarm Manager subsystem is time critical in the sense that any alarm/alert generated by the system must be delivered in a certain time and with 100% accuracy. However Careplan Editor and Monitoring Tool is more user interface focused subsystems. Subsystems which offer a graphical user interface are aimed to be a rich Internet applications, there is an overhead to synchronise the data model on the client side and on the server side. However this overhead can be tolerated considering that designing user friendly HTML based graphical user interfaces is a cumbersome work and mostly requires a web development framework which eventually results in framework-dependent code that catastrophically affects maintainability and extensibility of the code. Also these subsystems are designed to handle large amount of data so we plan to use frameworks that simplify complexity on server side and provides seamless object relational mapping integration.
6. User Interface Design

6.1. Overview of User Interface

All of the subsystems of Adaptive Care Planner Engine have a graphical user interface.

6.1.1. User Interface of Careplan Editor Subsystem

User interface of Careplan Editor subsystem offers different functionalities for Medical Domain Expert, Careplan Reviewer and Healthcare Actor. Following section explains actions that can be performed by each user type in each view.

6.1.1.1. Login View

Each user regardless of its type needs to login the system before utilizing any of the services provided by Adaptive Care Planner Engine.

6.1.1.2. Medical Careplan Repository View

This view provides creation, deletion, modification, approval of medical careplans. Medical Domain Expert can create, delete, modify and deactivate medical careplans. Careplan reviewer can view, approve and reject medical careplans.

6.1.1.3. Medical Careplan Editing View

This view provides an easy and user-friendly way to view and edit medical careplans. The algorithm that will be used in the execution of the medical careplan is presented as a diagram and very easy to edit by simple drag and drop operations. Also this view allows user to export a medical careplan in a special format that contains both medical careplan and visualization information. Also previously exported medical careplans can be imported. Another feature of the view is providing collapsible diagram blocks to present diagram as simple as possible to the user. Medical Domain Expert can edit, Careplan Reviewer can view and Healthcare Actor can personalize the medical careplans.

6.1.1.4. Decision Step Editing View

Decision step is used in the execution of the personalized medical careplan when a branch of the execution is to be decided by evaluating the evaluation script using real values for variables retrieved from CIED, PHR, EHR data services. Medical Domain Expert is able to edit evaluation script. Also variables in the evaluation script can also be edited through this view.
6.1.1.5. Consult Step Editing View

Consult step is used in the execution of the personalized medical careplan when a branch of the execution is to be decided by retrieving input from Healthcare Actor and using this input as a variable in evaluation script. Medical Domain Expert is able to edit evaluation script and variables. Also this view is allows editing of HTML form which will be displayed to the Healthcare Actor, through HTML Editing View.

6.1.1.6. Recommendation Step Editing View

Medical Domain Expert is able to edit urgency and receiver of the recommendation step. Also this view allows editing HTML form displayed to the Healthcare Actor through HTML Editing View.

6.1.1.7. HTML Editing View

Medical Domain Expert is able edit, HTML forms displayed in recommendation and consult which consists of different blocks such as Patient Block and CIED block. Each block has a HTML table template that user can edit by dragging and dropping HTML elements such as text area, radio button etc.

6.1.1.8. CIED/EHR/PHR Data Provider View

This interface is utilized by Medical Domain Expert to define retrieval rules for variables used in evaluation scripts in algorithm steps. CIED/EHR/PHR data schema is presented to Medical Domain Expert so that she/he is able to select any data provided by CIED/EHR/PHR data service.

6.1.2. User Interface of Careplan Engine Subsystem

Careplan Engine user interfaces are used to retrieve and display personalized medical careplans, retrieve and display patient records, initiate execution of personalized medical careplans and sending monitoring messages to Monitoring Tool subsystem.

6.1.2.1 Assignment View

User is able to retrieve PMCP instances and select any of them to change or remove Healthcare Actor assignments to patients.

6.1.2.2 Execution View

User can start or stop each PMCP instance’s execution also during the execution user is displayed execution logs and decisions made at each step.
6.1.3. User Interface of Alarm Manager Subsystem

Alarm Manager user interfaces are used to retrieve alarm messages, assign patients to healthcare actors, inspect the logs, send alarm to related persons and send acknowledgements.

6.1.3.1 Login View
User needs to login through this view to utilize functionalities of Alarm Manager Subsystem.

6.1.3.2 Alarm Logs View
User can view all the alarms and notifications sent by Alarm Manager through this view. User can view these logs sorted by date, patient or healthcare actor.

6.1.3.3 Configuration View
Through this view user configures when an alarm is send.

6.1.4. User Interface of Monitoring Tool

Monitoring Tool subsystem has only one user type, Healthcare Actor. Monitoring Tool user interfaces are used to view logs of execution of a personalized medical careplan instance such as when execution started, when eligibility criteria satisfied, current state of the execution and alarm generated. Healthcare Actor is able to examine each step of execution through logs even if the execution is terminated.

6.1.4.1 Login View
User needs to login through this view to utilize functionalities of Monitoring Tool Subsystem.

6.1.4.2 Monitoring View
All monitoring logs are viewed by user through this view as soon as a PMCP instance is selected.
6.2. Screen Images

6.2.1. Careplan Editor Subsystem

6.2.1.1. Login View

![Login View](image1)

**Figure 25: Login View**

6.2.1.2. Medical Careplan Repository View
6.2.1.3. Medical Careplan Editing View
6.2.1.4. Decision Step Editing View

Figure 28: Decision Step Editing View
6.2.1.5. Consult Step Editing View

![Consult Step Properties Diagram]

**Figure 29: Consult Step Editing View**

6.2.1.6. Recommendation Step Editing View

![Recommendation Step Properties Diagram]
6.2.1.7. HTML Editing View

An example Choice Block view is shown below. The table shown in the right will be converted to XHTML format and then will be displayed by a web browser.
Figure 32: Choice Block Editing View
6.2.1.8 CIED/EHR/PHR Data Provider View
6.2.2 Careplan Engine Subsystem

6.2.2.1 Assignment View

Figure 33: CIED/EHR/PHR Data Provider View
6.3. Screen Objects and Actions

6.3.1. Careplan Editor Subsystem

Following section explains screen objects and actions in Careplan Editor Subsystem.

6.3.1.1. Login View Object and Actions

At left side of the view important links of iCardea project is located. In the middle of the screen there is a text containing brief summary of the iCardea project. At the right, a log-in from is provided through which each user should log in to system. Following are available actions of the login view. Following actions are available:

- Log in : Healthcare Actor, Medical Domain Expert, Careplan Reviewer can log in to the system by filling username and password areas and clicking “Log In” button.
● Passwords Recovery: Healthcare Actor, Medical Domain Expert, Careplan Reviewer can request an e-mail containing their passwords by filling username area and clicking “Forgot your password?” label.

6.3.1.2. Medical Careplan Repository View

Existing medical careplan templates are shown in a data grid with id, name, version and approval status. Following actions are available.

● Create New MCP: Medical Domain Expert can create a new medical careplan by clicking “NEW” button.

● Edit MCP: Medical Domain Expert can edit an existing medical careplan by clicking edit image in the row of a medical careplan. As a result careplan is displayed in Medical Careplan Editing View.

● Ask For Approval of a MCP: Medical Domain Expert can ask for approval after he/she made any changes by clicking “Ask For Approval” column for any careplan.

● Approve/Reject MCP: Careplan Reviewer can approve/reject any of which approval is pending by clicking “Approve” and “Reject” button respectively.

● Import MCP: Medical Domain Expert may import a previously saved medical care plan by clicking “OPEN” button and choosing a file in the file open dialog.

This view provides creation, deletion, modification, approval of medical careplans.

6.3.1.3. Medical Careplan Editing View

This view consists of a diagram on the right and building blocks on the left. Following actions are available:

● Add Step: Authorized Healthcare Actor can add a new step by dragging and dropping an item from the “STEP” list to the diagram.

● Remove Step: Authorized Healthcare Actor can remove an existing step from diagram by clicking “X” button on the upper left corner of desired step.

● Open Edit Step Dialog: Authorized Healthcare Actor can a open edit dialog for any step by double clicking the step icon on the diagram. Appropriate dialog for each step type is displayed to the user.

● Linking Steps: Authorized Healthcare Actor can define next step of each step by simply clicking link icon that appears when mouse hovers on upper right corner of any step and dragging this icon to another step.

6.3.1.4. Decision Step Editing View

Healthcare Actor can edit a decision step through this view. List of variables are shown in “Variables” data grid. Following actions are available:

● Add Variable: Healthcare Actor can create a new variable to be used in the evaluation script by clicking “NEW” button.

● Edit Variable: Healthcare Actor can edit an existing variable by selecting a node at CIED/EHR/PHR Data Provider view which opens when clicks “EDIT” button.
● Remove Variable: Healthcare Actor can delete an existing variable by selecting a variable and clicking “DELETE” button.
● Edit Evaluation Script: Healthcare Actor can edit evaluation script simply by typing in the text area next to “Evaluation Script” label.

6.3.1.5. Consult Step Editing View

Healthcare Actor can edit a decision step through this view. List of variables are shown in “Variables” data grid. Following actions are available:

● Add Variable: Healthcare Actor can create a new variable to be used in the evaluation script by clicking “NEW” button.
● Edit Variable: Healthcare Actor can edit an existing variable by selecting a node at CIED/EHR/PHR Data Provider view which opens when clicks “EDIT” button.
● Remove Variable: Healthcare Actor can delete an existing variable by selecting a variable and clicking “DELETE” button.
● Edit Evaluation Script: Healthcare Actor can edit evaluation script simply by typing in the text area next to “Evaluation Script” label.
● Edit Display HTML: Healthcare Actor can edit HTML displayed during execution by clicking “DISPLAY(HTML)” button which opens HTML Editing View.

6.3.1.6. Recommendation Step Editing View

List of available actions are:

● Edit Display HTML: Healthcare Actor can edit HTML displayed during execution by clicking “DISPLAY(HTML)” button which opens HTML Editing View.
● Edit Urgency: Healthcare Actor can edit urgency simply by typing in text area next to “Urgency” label.
● Edit Receiver: Healthcare Actor can edit receiver simply by typing in text area next to “Receiver” label.

6.3.1.7. HTML Editing View

Healthcare Actor can perform following actions in this view:

● Selecting Block: Healthcare Actor can select a block after clicking “Block Editor” button and then selecting any of the Patient, EGM, Choice, CIED Blocks.
● Creating Custom Block: Healthcare Actor can create a new block by clicking “Custom Block” button and providing width and height of the table.
● Editing HTML View: Healthcare Actor can drag and drop HTML elements after selecting a block or creating a custom block.

6.3.1.8. CIED/EHR/PHR Data Provider View
CIED, EHR and PHR data schemas are represented in a tree model and user can select any node then click “SAVE” button to save that value as a variable in an algorithm step.

6.3.2 Careplan Engine Subsystem

6.3.2.1 Assignment View
Existing medical careplan templates are shown in a data grid with id, name, version and approval status and approval date also healthcare actors assigned to the patient. Following actions are available. Personalized medical careplan instances are displayed in another data grid.

- Assign Healthcare Actor: User can select a patient and a healthcare actor in order to assign the healthcare actor to the user.
- Remove Healthcare Actor Assignment: User can select a healthcare actor from a patient’s healthcare actor list and click “Remove Assignment” button to remove that assignment.
- View Execution: User can select any PMCP instance from the data grid and click “View Execution” button. Then user is displayed with an “Execution View” window.

6.3.2.2 Execution View
In this window all steps of an execution can be traced. Following actions are available. A simple log is displayed to the user that contains all history of the steps taken in execution if execution is started.

- Start Execution: User can start execution of the PMCP instance by clicking “Start Execution” button.
- Stop Execution: User can stop execution of the PMCP instance by clicking “Stop Execution” button.

6.3.3 Alarm Manager Subsystem

6.3.3.1 Login View
Following actions are available in this view:

- Log in: Healthcare Actor, Medical Domain Expert, Careplan Reviewer can log in to the system by filling username and password areas and clicking “Log In” button.
- Passwords Recovery: Healthcare Actor, Medical Domain Expert, Careplan Reviewer can request an e-mail containing their passwords by filling username area and clicking “Forgot your password?” label.

6.3.3.2 Alarm Logs View
In this view Alarm logs is displayed to user in data grid. This information contains date, patient and healthcare actor information, content of the alarm and actor’s response if available. User can sort the items in the data grid by clicking any of the columns. There are no further actions in this view.
6.3.3.3 Configuration View

In this view user can configure any alarm that will be send by Alarm Manager to a specific healthcare actor. Available actions are as follows:

- Add Alarm: User can select in any step in the careplan’s algorithm and associate that step with an alarm by clicking “Add Alarm” button.
- Remove Alarm: User can select any previously created alarm and delete by pressing “Delete Alarm” button.

6.3.4. Monitoring Tool Subsystem

6.3.4.1 Login View

Following actions are available in this view:

- Log in: Healthcare Actor, Medical Domain Expert, Careplan Reviewer can log in to the system by filling username and password areas and clicking “Log In” button.
- Passwords Recovery: Healthcare Actor, Medical Domain Expert, Careplan Reviewer can request an e-mail containing their passwords by filling username area and clicking “Forgot your password?” label.

6.3.4.2 Monitoring View

In this view all the evaluation scripts, external data retrieved from Care Manager DB Interface, alarms and forms sent to healthcare actors and their responses on each step of evaluation are shown. There are no actions for user since user is not editing any specific content.

7. Detailed Design

7.1. Careplan Editor Subsystem

It is a subsystem that is used for creation, modification, and approval of the medical careplan templates, and personalized medical careplan templates. It utilizes MCP Repository Interface to store and retrieve the medical careplans.

Careplan Editor GUI is a rich Internet application that is used via an Internet browser, which means that all the data model with the application itself will be loaded by browser and run on client side. Thus, the program needs to minimize consumption of memory and cpu cycles while executing, aiming to reduce the burden to user. To achieve this goal all entities that will be displayed to the user will be lazily loaded from server.

7.1.1. Structural Design

Careplan Editor presents a user-friendly interface and provides functionalities shown in Figure 36. It uses MCPR Repository for storage of careplans and Care Manager for retrieving and editing of patient related data.
Figure 36: Class Diagram for Careplan Editor Subsystem

Figure 37 shows the data model for a personalized medical careplan instance. This model is created after a medical careplan template is created, approved and personalized. The model is persisted by Medical Careplan Repository.
7.1.2. Behavioral Diagrams

The creation of a new Medical Careplan Template is the first step when using the Careplan Editor Subsystem. This Medical Careplan Template is modified by adding Eligibility Criteria, Algorithm, Consult Step, Recommendation Step and Decision Step. After the Medical Careplan Template is approved, Approved Medical Careplan Template can still be modified, or it is personalized. Personalized Medical Careplan can also be modified. Finally, the execution of Personalized Medical Careplan leads to Personalized Medical Careplan Instance.
7.1.3. Interaction Diagrams

The interaction diagram for “Creation of New Medical Careplan Template” is shown in Figure 39. As shown in the figure, the Medical Domain Expert creates the MCP graphically by specifying the eligibility criteria and the steps in the algorithm. When s/he saves the template, it is verified in the Careplan Editor and added to the MCP repository. Finally, Medical Domain Expert asks for the approval of the template.
The interaction diagram for “Management of Existing Medical Careplan Templates” is shown in Figure 40. As in “Creation of New Medical Careplan Template”, the Medical Domain Expert modifies an existing MCP template graphically by using the Careplan Editor GUI. The modification and verification of the GLIF definition of the MCP is performed by the Careplan Editor, and MCP Repository holds the modified MCP template for later approval.
Figure 40: Interaction Diagram for Management of Existing Medical Careplan Templates

The interaction diagram for “Approval of Medical Careplan Template” is displayed in Figure 41. As shown in the figure, the Careplan Reviewer is first displayed unapproved templates and then s/he selects one of them. After examining the details of the MCP, s/he approves/rejects the MCP.

Figure 41: Interaction Diagram for Approval of Medical Careplan Template

The interaction diagram for “Generation of Personalized Medical Careplan (PMCP)” is provided in Figure 42. The Healthcare Actor first selects the Patient for whom the MCP is personalized. After that s/he obtains the MCP Templates from MCP Repository and selects one of them for the patient. Having linked the template to the actual data sources, the Health Care
actor personalizes the MCP Template according to Patient Data. Finally, Careplan Editor constructs the PMCP and after validation it sends the PMCP to MCP Repository through MCP Repository Interface.

Figure 42: Interaction Diagram for Generation of Personalized Medical Careplan (PMCP)

The interaction diagram for “Modification of Personalized Medical Careplan (PMCP)” is presented in Figure 43, where the Healthcare Actor modifies a previously personalized PMCP. The sequence is similar to “Generation of Personalized Medical Careplan (PMCP)”; that is, when performing the modification, the Healthcare Actor first obtains the PMCP to be modified from the MCP repository. After that s/he obtains the data from Patient Parameter Monitor Interface. Having been modified by the Healthcare Actor, the PMCP is inserted back to the MCP Repository.
7.2. Careplan Engine Subsystem

It is the subsystem where the actual execution of the personalized medical careplan instances takes place. These instances are retrieved from MCP Repository Interface. Instances are executed according to following algorithm:
1. Start execution of a PMCP Instance.
2. Check if eligibility criteria is satisfied. If it is satisfied go to 3, else go to 1.
3. Go to Start step.
4. Decide the next step.
5. If the next step is Recommendation step, retrieve the variable values from Care Management DB Interface, and according to these values evaluate the evaluation script, and finally recommend an action.
6. If the next step is Decision step then a Recommendation step is chosen from a list by evaluating evaluation script.
7. If the next step is Consult step, Healthcare Actor is send an HTML form requesting the parameters which will be used in execution of evaluation function of this step, Healthcare Actor fills and sends the form back to the engine, and according to the outcome the next step is decided.
8. If the next step is the final step, then finalize execution.

7.2.1. Behavioral Diagram

The first action for the Careplan Engine Subsystem is the start of execution of a PMCP Instance. If eligibility criteria are satisfied, Start step is executed. Then, the next step is decided.
If the next step is Recommendation step, the variable values are retrieved from the Care Management DB Interface, and according to these values the evaluation script is evaluated, and finally an action is recommended. If the next step is Decision step then a choice between multiple recommendation steps must be made using parameters supplied by Care Management DB Interface. This step can be best interpreted as a branching step between multiple recommendation steps. If the next step is Consult step, Healthcare Actor is send an HTML form requesting the parameters which will be used in execution of evaluation function of this step, Healthcare Actor fills and sends the form back to the engine, and according to the outcome the next step is decided. If the next step is the final step, then the execution is finalized.

![State Transition Diagram for Careplan Engine Subsystem](image)

**Figure 44: State Transition Diagram for Careplan Engine Subsystem**

### 7.2.2. Interaction Diagrams
The interaction diagram for “Provision of Medical services specified in Medical Careplan Definitions” is shown in Figure 45. The Technical Domain Expert first retrieves the PMCP from the MCP Repository. Having examined the PCMP through Careplan Engine GUI, s/he identifies the points where an interaction with the Hospital Information System (HIS) is required. After that, s/he queries the Medical Service Registry Interface and obtains related available Web Services HIS. Finally, the Technical Domain Expert makes the assignments.

![Figure 45: Interaction Diagram for Provision of Medical services specified in Medical Careplan Definitions](image)

The interaction diagram for “Subscription to related Clinical Data Sources and bind Medical Operation Services” is shown in Figure 46. First, the Technical Domain Expert retrieves the PMCP from the MCP Repository. Having inspected the PMCP definition, s/he identifies the Service Definition of the data sources from the Medical Service Interface and sends necessary Subscription messages to those services through Clinical Data Source Interface. Finally, the Technical Domain Expert saves the PMCP definition back to the MCP Repository.
The interaction diagram for “Execution Personalized Medical Careplan (PMCP)” is presented in Figure 47. The Healthcare actor first identifies the PMCP and the Patient, and executes the PMCP. In return, a PMCP Instance is created. During the course of the execution necessary Alarm and Monitoring Messages are sent to Alarm Manager and Care Plan Monitoring Tool, respectively. Furthermore, required clinical data about the Patient is retrieve from Care Management DB and Web services of HIS is invoked by using Medical Service Interface. It should be noted that the sequence of actions shown in Figure 47 is not strict. It may change according to the execution of a particular PMCP Instance.
7.3. Monitoring Tool Subsystem

Monitoring Tool Subsystem is responsible for displaying terminated and on-going personalized medical careplan instances. Monitoring Tool Subsystem displays this information to the user through Monitoring Tool GUI. This interface is designed as Rich Internet Application, which can be run on client-side with a web browser and Adobe Flash Player. For this interface to function properly both Careplan Engine and MCP Repository must be up and running.

7.3.1. Behavioral Diagram

The first step for the Monitoring Tool Subsystem is selection of an instance of executing careplan. Then, messages are retrieved from the Careplan Engine. After that, Confirmation responses are obtained from the users. Finally, after displaying the execution status on GUI, the Careplan Engine is informed about the Confirmation Status.
7.3.2. Interaction Diagrams

The interaction diagram for “Retrieval of Monitoring Messages” is presented in Figure 49.

The interaction diagram for “Monitoring of Personalized Medical Careplans (PMCPs) assigned to Patients” is shown in Figure 50.
Figure 50: Interaction Diagram for Monitoring of Personalized Medical Careplans (PMCPs) assigned to Patients

The interaction diagram for “Tracing completed/terminated Personalized Medical Careplans (PMCPs)” is presented in Figure 51.

Figure 51: Interaction Diagram for Tracing completed/terminated Personalized Medical Careplans (PMCPs)

7.4. Alarm Manager Subsystem

Alarm Manager is designed to send alarms and notification to related patients or healthcare actors when previously defined situations occur. The highest priority of the Alarm Manager is to send alarms and notifications with 100% accuracy. Also it should send them in a timely manner since in certain conditions immediate actions needs to be taken to save a patient’s life. Alarm Manager provides both graphical user interface and a programming interface. GUI is used for retrieving alarm logs and updating Personalized Medical Careplan configuration for alarms while programming interface is utilized by Careplan Engine to send alarms. Figure 52 shows these interfaces in detail.
7.4.1. Structural Design

Care Manager component shown in Figure 53, is responsible for retrieving any external data related to patient. It can be CIED, EHR or PHR data. Care Manager component is capable of connecting any CIED, EHR, PHR endpoint in runtime. Also it is able to retrieve all Healthcare Actors assigned to a specific patient and also send notification about new assignments. Both Alarm Manager and Careplan Engine utilizing functionalities provided by Care Manager to carry out tasks such as assigning Healthcare Actor to a patient or executing personalized medical careplan instances.
Figure 53: Class Diagram for Care Management DB Interface

7.4.2. Behavioral Diagram

The first action for the Alarm Manager Subsystem is the retrieval of Alarm Message. Then, the recipient and the medium are decided in sequence. After sending GTalk Message, SMS Message or Email, acknowledgement (ACK) is waited for. Before the final step, if re-route message is retrieved, then again ACK is waited for.
7.4.3. Interaction Diagrams

The interaction diagram for “Giving Monitoring Authorization” is presented in Figure 55. First, the Healthcare Administrator obtains the PMCPs from MCP Repository and obtains the Healthcare Actors from Care Management DB and related HIS Web Service. After the Assignment is realized, the Alarm Manager sends it to MCP Repository for later use.
The interaction diagram for "Adding New Healthcare Actor" is shown in Figure 56.

The interaction diagram for "Management of Personal & Contact Information" is presented in Figure 57.
The interaction diagram for “Management of Alert Message Logs” presented in Figure 58.

First, the Healthcare Actor retrieves the PMCPs from the MCP Repository and selects one of them. After that s/he edits the configuration information related with the PMCP instance.

Following is a list of external software that we are planning to use for implementing the project:

- Source Code Management, Project Management and Build Automation Tools
  - **Apache Subversion[4]**: Apache Subversion (often abbreviated SVN, after the command name svn) is a software versioning and a revision control system
founded and sponsored in 2000 by CollabNet Inc. Developers use Subversion to maintain current and historical versions of files such as source code, web pages, and documentation.

- **Apache Maven[5]:** Maven is a software tool for project management and build automation. While primarily used for Java programming, it can also be used to build and manage projects written in C#, Ruby, Scala, and other languages. Maven serves a similar purpose as the Apache Ant tool, but it is based on different concepts and works in a profoundly different manner. Maven is hosted by the Apache Software Foundation, where it was formerly part of the Jakarta Project.

- **Google Code[6]:** Project Hosting gives users version control for open source code. We use Google Code as SVN repository, wiki and issue management tool.

### Frameworks

- **Adobe Flex SDK 3.5.0[7]:** Adobe Flex is a software development kit (SDK) released by Adobe Systems for the development and deployment of cross-platform rich Internet applications based on the Adobe Flash platform. Flex applications can be written using Adobe Flash Builder or by using the freely available Flex compiler from Adobe.

- **BlazeDS[8]:** BlazeDS is a server-based Java remoting and web messaging technology that allows you to connect to back-end distributed data and push data in real-time to Adobe Flex and Adobe Integrated Runtime (AIR) Rich Internet applications (RIA). Because of its open licensing, BlazeDS is not precluded from being used with other client platforms, such as JavaScript/Ajax.

- **Spring[9]:** The Spring Framework is an open source application framework for the Java platform. A particular reason for us to choose Spring Framework is its seamless Hibernate and BlazeDS integration.

- **Hibernate[10]:** Hibernate is an object-relational mapping (ORM) library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions.

### Libraries

- **Kap Lab Diagrammer[11]:** A customizable tool for drawing diagram in Adobe Flex applications.

- **FlexXB[12]:** An Action Script 3 Library for serializing/deserializing Action Script object to XML documents.

### Development Environment

- **Eclipse IDE for Java EE Developers[13]:** Eclipse is a multi-language software development environment comprising an integrated development environment (IDE) and an extensible plug-in system. It is written mostly in Java and can be used to develop applications in Java and, by means of various plug-ins.

- **Adobe Flash Builder[14]:** Adobe Flash Builder (previously known as Adobe Flex Builder) is an integrated development environment (IDE) built on the Eclipse platform and speeds development of rich Internet applications (RIAs) and cross-
platform desktop applications, particularly for the Adobe Flash platform. Support for cross-platform desktop applications was added in version 3 with the introduction of AIR.

9. Time Planning (Gantt Chart)
**Figure 60: Gantt Chart**

<table>
<thead>
<tr>
<th>Number</th>
<th>Task Description</th>
<th>Start Date</th>
<th>End Date</th>
<th>Duration</th>
<th>% Complete</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Project Management</td>
<td>1/11/2010</td>
<td>3/5/2011</td>
<td>2/7</td>
<td>10.0</td>
</tr>
<tr>
<td>4</td>
<td>Design Report</td>
<td>1/12/2010</td>
<td>4/12/2010</td>
<td>13</td>
<td>100.0</td>
</tr>
<tr>
<td>6</td>
<td>Web Page</td>
<td>2/12/2010</td>
<td>3/5/2011</td>
<td>3</td>
<td>10.0</td>
</tr>
<tr>
<td>7</td>
<td>Logic Management</td>
<td>2/13/2010</td>
<td>3/5/2011</td>
<td>6</td>
<td>5.0</td>
</tr>
<tr>
<td>8</td>
<td>Guideline Definition Tool</td>
<td>3/14/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>15.0</td>
</tr>
<tr>
<td>9</td>
<td>Setting up Development Environment</td>
<td>3/15/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>0.0</td>
</tr>
<tr>
<td>10</td>
<td>Implementing Basic Functionality</td>
<td>3/16/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>11</td>
<td>Implementing HTML Editor</td>
<td>3/17/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>12</td>
<td>Implementing User Management</td>
<td>3/18/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>13</td>
<td>Unit Testing of the Components</td>
<td>3/19/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>14</td>
<td>Integration of the Components</td>
<td>3/20/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>16</td>
<td>Testing and Debugging</td>
<td>3/22/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>18</td>
<td>Guideline Monitoring Tool</td>
<td>3/24/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>19</td>
<td>Messages from the Campus Engine</td>
<td>3/25/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>20</td>
<td>Messages from the Alarm and Notification Manager</td>
<td>3/26/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>21</td>
<td>Traceback to the Execution Logs</td>
<td>3/27/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>22</td>
<td>Unit Testing of the Components</td>
<td>3/28/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>23</td>
<td>Integration of the Components</td>
<td>3/29/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>26</td>
<td>Deployment and Maintenance</td>
<td>4/2/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
<tr>
<td>27</td>
<td>Finalization</td>
<td>4/3/2010</td>
<td>3/5/2011</td>
<td>10</td>
<td>100.0</td>
</tr>
</tbody>
</table>
9.1. Term 1 Gantt Chart

Based on the Figure 60 shown above, from November 2010 to the end of the January 2011 shows the Term 1 schedule of the project.

9.2. Term 2 Gantt Chart

Based on the Figure 60 shown above, from February 2011 to the end of the May 2011 shows the Term 2 schedule of the project.

10. Conclusion

This document states the design level approach taken by Antique Cows team for the iCardea project. In this document, a fair amount of elaboration has been done on the project scenario pointing out most of the important details. The goals for the final product has become more apparent as the scenario and the desired user interface is visually explained. Additionally, this document is the first document that explains somewhat deep technical details. The architecture of the system is discussed with an overview. Further information on the technical design is given with detailed explanations of the modules which are supported with diagrams. Finally, the progress made by the project team is summarized.