CardioNotifier

A smartphone-based notification system for the cardiologist

Tamir Tsedenjav
September 2016
CardioNotifier
A smartphone-based notification system for the cardiologist

Eindhoven University of Technology
Stan Ackermans Institute / Software Technology

Tamir Tsedenjav

The design described in this report has been carried out in accordance with the TU/e Code of Scientific Conduct
Abstract

Philips provides cardiology solutions for medical data handling which are Cardiovascular Information Management Systems (CVIS). Recently, with the introduction of IntelliSpace CardioVascular (ISCV), Philips is shifting towards web-based solutions that can be used anytime and anywhere. However, a mobile solution that can notify the cardiologist of updates in patient data without direct access to the web-based CVIS would improve the efficiency of the system.

Consequently, smartphone-based notification system for the cardiologist was developed and demonstrated. For this, a research about cardiologist’s workflow was conducted to find out how smartphone can support cardiologist’s daily work. Also, various technical investigations were taken to achieve most optimum design solutions.

In the project FHIR was selected as the standard as this is an emerging and modern standard for healthcare information. The project proposes a prototype to demonstrate cooperation between a FHIR server and a mobile app. The report describes the process of design and implementation of a notification system. With help of the mobile app, cardiologist can be notified with context of information which is changed in the server.

Keywords
Cardiology, Smartphone, Notification system, FHIR (Fast Healthcare Interoperability Resources), cardiologist, cardiologist workflow

Preferred reference

Partnership
This project was supported by Eindhoven University of Technology and Philips Research.

Disclaimer and Endorsement
Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the Eindhoven University of Technology or Philips Research. The views and opinions of authors expressed herein do not necessarily state or reflect those of the Eindhoven University of Technology or Philips Research, and shall not be used for advertising or product endorsement purposes.

Disclaimer Liability
While every effort will be made to ensure that the information contained within this report is accurate and up to date, Eindhoven University of Technology makes no warranty, representation or undertaking whether expressed or implied, nor does it assume any legal liability, whether direct or indirect, or responsibility for the accuracy, completeness, or usefulness of any information.

Trademarks
Product and company names mentioned herein may be trademarks and/or service marks of their respective owners. We use these names without any particular endorsement or with the intent to infringe the copyright of the respective owners.

Copyright
Copyright © 2016. Eindhoven University of Technology. All rights reserved.
No part of the material protected by this copyright notice may be reproduced, modified, or
Foreword

Tamir’s project “CardioNotifier: A smartphone-based notification system for the cardiologist” has delivered a prototype mobile app for notifying a cardiologist of changes in the health records of his/her patients.

In the healthcare domain, large amounts of data are generated per patient, and the cardiology department is no exception. Although some information hardly changes, other information may be updated frequently – during a visit to the outpatient clinic, and especially when a patient is admitted to the hospital. New data is generated in the form of scans, blood analysis, blood pressure readings, ECGs, updated medication history, etc. Some of this new information is urgent, in which the staff pages the cardiologist and the information is shared person to person. Other information can be shared less urgently, but must still be communicated efficiently and correctly. Every piece of data adds to the overall picture of the patient, and may change the treatment plan of the cardiologist.

With the introduction of smartphones and tablets to the hospital, new methods to inform healthcare professionals have become possible – for instance via text messages, sharing of photos, and accessing of patient records from mobile devices. The assignment that was given to Tamir was in the area of using smartphones to notify a cardiologist that the record of a patient had been updated. Tamir was to explore the possibilities and restrictions of such an application, and to find out what would work best in clinical practice.

The assignment posed several challenges. Tamir had to gain insight in the healthcare and cardiology domain. A cardiologist typically manages a high number of patients, and the health record of each of these which may be updated multiple times per day. Thus, it is very simple to generate such a high number of notifications that the app will severely decrease the efficiency of the cardiologist, rather than improve it. However, identifying the situations in which a notification will be useful requires a thorough understanding of the daily practice of the cardiologist. Secondly, these clinical insights are usually in the minds of non-IT people. Therefore, Tamir had to translate user input into technical requirements, update the prototype, re-translate the technical improvements in user advantages, and then obtain feedback. Also, technologies were used he had no experience with, so Tamir had to acquaint himself with new types of data, standards and environments. Finally, the assignment was an exploratory study executed within Philips Research rather than an integral part of the ongoing activities in Cardiology Informatics systems at Philips. At the beginning of the project it was yet undefined what exactly had to be created. Tamir had to grow from a software developer into a software engineer, conducting himself more in a senior role. He also had to work much more self-managing and self-reliant than in his previous jobs.

Tamir has picked up these challenges, hesitant at first, successful in the end. He has worked with clinical experts at Philips to understand and identify the most suitable use cases. He has created a series of prototypes, each with increased functionality. He leaves us with a working prototype of a notification app and a web application, which automatically create notifications for the cardiologist upon updates to patient records, if specific and configurable conditions have been met. We can use this prototype to approach the end-users, and gain feedback to further develop and refine the application, so that we can finally develop it into a module on our Cardiology Informatics systems.

Dr. Charles Sio
Philips Research, Professional Healthcare
Eindhoven, September 2016
Preface

This report was written to provide a detailed summary of a nine-month project carried out at Philips Research, Eindhoven. The project serves as a final project for the Software Technology (ST) program for the Eindhoven University of Technology (TU/e), towards the Professional Doctorate in Engineering (PDEng) degree. Therefore, the project serves two main purposes which are to allow the author to demonstrate his capabilities in independently carrying out a design project and to provide a sufficient added value for Philips Research. Consequently, the report contains an account of the design process, and at the same time it dives deep into the problem, domain, and requirement analysis and provides various design decisions with the explanation of the rationale behind them.

Date: September, 2016
Acknowledgements

I would like to express my deepest appreciation to several people who provided me the possibility to complete this project.

First of all, I am sincerely grateful to my company supervisors, Roel Truyen, Charles Sio, and Iwo Serlie for great opportunity to carry out this project at Philips Research. They dedicated continuous support to this project. I could not have accomplished this project without their priceless suggestions, constructive comments, and kind encouragement.

I would like to thank Dirk Willy-an for providing me indispensable information, and enlightening ideas on the different aspects of cardiology field.

I am genuinely thankful for my university supervisor, Ad Aerts, for giving me the opportunity to be a part of the OOTI program and also for his time and effort to give me precise guidance and valuable feedbacks and enlightening ideas throughout this project.

I would like to thank my all OOTI colleagues with whom I spent wonderful time together during the last two years. I learned a lot from them.

I am grateful for Yanja Dajasuren and Razvan Dinu for their great advice, guidance, support, and help.

I am thankful for icons made by Freepik from www.flaticon.com.

I am thankful for Philips Research colleagues who made my time at Philips Research very enjoyable.

I am thankful for Judith Strother for reviewing my report.

Also, I must acknowledge my entire family for their support, love, and encouragement. Specially, I am heartily grateful to our mothers, Tuya Yondon and Altanzaya Gunsen for their time and dedication to support my study.

Finally, I would like to acknowledge my wife, Bayasgalan and the only person who can cheer me up in any condition, my son Orgil. I would not have finished this project without their love and encouragement.

Date: September, 2016
Tamir Tsedenjav
Executive Summary

Philips provides cardiology application solutions in the Cardiovascular Information Management System (CVIS) domain, such as Xcelera and IntelliSpace CardioVascular (ISCV).

Xcelera is a traditional cardiology desktop-based application, an integrated multi-modality image management system for cardiovascular information. This provides a single access point for multi-modality image viewing, specialized clinical applications and reporting.

More recently, with the introduction of ISCV, Philips is shifting towards web-based solutions that can be used anytime and anywhere. However, without a notification system a cardiologist is required to stay close to workstation to not miss any important information update. Staying at one place is not possible for a cardiologist: they have to be at many places, including those where there is no access to the workstations, throughout the day. All hospitals have efficient systems implemented for relaying emergency information, but for sub-emergency information a balance must be found between speed of relaying updates and interruption of daily activities. Because of that, crucial time may be lost by not having fast access to important information at any location in the hospital. Therefore, a mobile solution that can notify the cardiologist would be very helpful.

As a solution for above problem, in the CardioNotifier project, a smartphone-based notification system for the cardiologist, was developed and demonstrated. For this, a research about cardiologist’s workflow was conducted to find out how a smartphone can support the cardiologist’s daily work. Also, various technical investigations were done to achieve most optimal design solutions.

In this project we selected FHIR as a standard to work with, since it is an emerging and modern standard, endorsed by HL7. Therefore, the project also covered the detailed investigation and possibilities of FHIR standard usage in mobile application solution.

On top of that, a usage of business rule engine was proposed to provide the client more flexibility to define the logic of the subscription.

As a result of the project, the CardioNotifier, a smartphone-based notification system with integrated FHIR server for the cardiologist was implemented and demonstrated as a support of the concept.
Table of Contents

Foreword .................................................................................................................. i
Preface ..................................................................................................................... iii
Acknowledgements ................................................................................................. v
Executive Summary ................................................................................................. vii
Table of Contents .................................................................................................. ix
List of Figures ......................................................................................................... xiii
List of Tables .......................................................................................................... xv

1. Introduction ........................................................................................................ 1
   1.1 Context .......................................................................................................... 1
       1.1.1 Philips Research ..................................................................................... 1
       1.1.2 Philips contribution to cardiology ......................................................... 1
   1.2 The Project .................................................................................................... 1
       1.2.1 The Starting Point .................................................................................. 1
       1.2.2 The Objective ........................................................................................ 2
       1.2.3 The CardioNotifier Application ........................................................... 2
       1.2.4 The Scope .............................................................................................. 2
   1.3 Outline .......................................................................................................... 2

2. Stakeholder Analysis ......................................................................................... 5
   2.1 Potential End Users ..................................................................................... 5
   2.2 Clinical expert .............................................................................................. 5
   2.3 Philips Research: Project owner ................................................................. 6
   2.4 Philips Research: Project Manager .............................................................. 6
   2.5 TU/e ............................................................................................................. 7
   2.6 Philips Healthcare – Business Unit, Healthcare IT, Cardiology Informatics ......................................................... 7
   2.7 Business people ........................................................................................... 8
   2.8 The stakeholder mapping ............................................................................ 8
   2.9 Conclusion ................................................................................................... 9

3. Domain Analysis ............................................................................................... 11
   3.1 Cardiology .................................................................................................... 11
       3.1.1 Non-invasive workflow ......................................................................... 12
   3.2 Philips Cardiovascular Information Management System (CVIS) .......... 13
       3.2.1 IntelliSpace Cardiovascular (ISCV) ....................................................... 14
3.3 Fast Healthcare Interoperability Resources (FHIR)..........................14
  3.3.1. Collaboration with FHIR ..............................................14
  3.3.2. FHIR Technologies..................................................14
3.4 SMART on FHIR ..................................................................15
  3.4.1. Who is involved.........................................................16
  3.4.2. Why use SMART on FHIR .........................................17
  3.4.3. How is the support? ..................................................17
3.5 Conclusion...........................................................................17

4. Problem Analysis ..................................................................19
  4.1 Problem descriptions......................................................19
    4.1.1. Mobile cardiologist and a fixed workstation ..................19
    4.1.2. Filtering important information ..................................20
    4.1.3. Limitations of already available tools..........................20
  4.2 Design Opportunities .....................................................20

5. Feasibility Analysis ..............................................................23
  5.1 Technical challenges ......................................................23
    5.1.1. Smartphone network instability ..................................23
    5.1.2. New technologies ...................................................23
    5.1.3. Immature technologies .............................................23
    5.1.4. Unfamiliar domain: Healthcare IT ..............................23
  5.2 Operational challenges and risks ......................................24
    5.2.1. Communication with stakeholders who have different
            backgrounds ............................................................24
    5.2.2. Integration with ISCV ..............................................24
    5.2.3. New requirement or requirement change at the last moment..24
    5.2.4. Demo failure .........................................................24
    5.2.5. Inability to contact the clinical expert ..........................24

6. System Requirements ...........................................................25
  6.1 Functional requirements ..................................................25
    6.1.1. Subscribe to the information to be notified ....................25
    6.1.2. Receive a notification ..............................................25
  6.2 Non-functional requirements..............................................25
    6.2.1. Availability ............................................................25
    6.2.2. Reliability .............................................................25
    6.2.3. Performance efficiency ............................................26
    6.2.4. Usability ..............................................................26
    6.2.5. Security and privacy ...............................................26
    6.2.6. Standard compliance ..............................................26

7. System Architecture .............................................................27
  7.1 Introduction.........................................................................27
  7.2 Use case.............................................................................27
    7.2.1. Use case 1: Receive notification .................................27
    7.2.2. Use case 2: Check notification ...................................28
    7.2.3. Use case 3: Configure subscription ............................28
    7.2.4. Use case 3: Send notification manually .......................28
    7.2.5. Use case 3: Check notification .................................28
7.3 Logical View .............................................................. 29
7.3.1. Context ............................................................. 29
7.3.2. CardioNotifier with FHIR ......................................... 29
7.3.3. Security and privacy ............................................... 29

7.4 Development view .................................................... 30
7.4.1. FHIR Server .......................................................... 30
7.4.2. FHIR RESTful API ................................................... 30
7.4.3. Interface between Mobile App and FHIR service ............... 31
7.4.4. Push notification service .......................................... 31
7.4.5. Polling or Push Notification service? ............................. 32

7.5 Process view ............................................................. 32
7.5.1. Activities between components ................................. 33
7.5.2. Activities in Web App .............................................. 33
7.5.3. Activities in FHIR server .......................................... 33
7.5.4. Activities in Mobile App ........................................... 33
7.5.5. Activities in Message Generator ................................. 33

7.6 Physical view ............................................................ 36

7.7 The overall system architecture .................................... 36

8. System Design .................................................................. 37
8.1 Mobile App .................................................................... 37
8.1.1. Mobile application platform .................................... 37
8.1.2. Mobile application type ........................................... 37
8.1.3. Logical View: Factory and Observer Design Patterns ......... 38
8.2 FHIR Server .................................................................. 39
8.2.1. Some available FHIR Servers .................................. 39
8.2.2. Web App using FHIR subscription .............................. 40

8.3 Message Generator ....................................................... 41
8.3.1. Business rule engine ................................................. 41
8.3.2. Design of the Message Generator ............................... 42

8.4 Conclusion ..................................................................... 42

9. Implementation ................................................................ 43
9.1 Mobile App GUI ........................................................... 43

9.2 Web App ...................................................................... 44
9.2.1. Subscription ........................................................... 44
9.2.2. Create/update patient data ...................................... 45

9.3 FHIR service ............................................................... 46
9.3.1. Custom notification service ...................................... 46

9.4 Data repository ............................................................ 46

10. Verification & Validation ................................................ 49
10.1 Verification ................................................................. 49
10.1.1. Functional Test ...................................................... 49
10.1.2. Non-functional Requirements Evaluation ..................... 50

10.2 Validation ................................................................. 51
10.2.1. Validation with Cardiologist .................................... 51
10.2.2. Feedback .............................................................. 51
10.2.3. Conclusion ........................................................................53

11. Conclusions ........................................................................55
  11.1 Results.................................................................................55
  11.2 Future works.................................................................55

12. Project Management ..........................................................57
  12.1 Introduction ......................................................................57
  12.2 Work-Breakdown Structure (WBS) ....................................57
  12.3 Project planning ..............................................................58
    12.3.1. Initial ........................................................................58
    12.3.2. Final ..........................................................................59
  12.4 Conclusion .......................................................................59

13. Project Retrospective .........................................................61
  13.1 Reflection .................................................................61
    13.1.1. Lessons Learned ..................................................61
  13.2 Design Opportunities revisited .......................................62

Glossary .................................................................................63

Bibliography ............................................................................65

About the Author .................................................................67
List of Figures

Figure 1 The stakeholder mapping................................................................. 9
Figure 2 Non-invasive cardiology workflow.................................................. 12
Figure 3 Overall architecture of FHIR..........................................................15
Figure 4 SMART on FHIR..............................................................................16
Figure 5 Cardiologist's activities at a hospital................................................ 19
Figure 6 Use case view....................................................................................27
Figure 7 CardioNotifier with FHIR Server....................................................29
Figure 8 The Architecture of FHIR Server......................................................30
Figure 9 Activity diagram of CardioNotifier..................................................35
Figure 10 Deployment View of CardioNotifier...............................................36
Figure 11 Overall Architecture of CardioNotifier..........................................36
Figure 12 Class diagram of Mobile App..........................................................39
Figure 13 Sequence Diagram of Subscription...............................................40
Figure 14 Class diagram of the Message Generator.......................................42
Figure 15 Overall System Design of the CardioNotifier.................................42
Figure 16 Mobile app main GUI.....................................................................43
Figure 17 Mobile app detailed GUI.................................................................44
Figure 18 Web App GUI..................................................................................45
Figure 19 Create/update patient data window...............................................45
Figure 20 Subscription UML Diagram.............................................................46
Figure 21 MessageHeader UML Diagram......................................................46
Figure 22 Observation UML Diagram..............................................................47
Figure 23 The work-breakdown structure.......................................................57
Figure 24 The Project Planning.......................................................................58
Figure 25 The initial agile product backlog in Scrum.....................................58
Figure 26 The final agile product backlog in Scrum.......................................59
List of Tables

Table 1 Potential end user stakeholder analysis (Cardiologist) ........................................ 5
Table 2 Clinical expert stakeholder analysis ...................................................................... 5
Table 3 Project owner stakeholder analysis ....................................................................... 6
Table 4 Project manager stakeholder analysis .................................................................... 6
Table 5 TU/e supervisor stakeholder analysis .................................................................... 7
Table 6 PDEng trainee stakeholder analysis ...................................................................... 7
Table 7 Philips Healthcare - Business Unit Cardiology Informatics Section stakeholder analysis ................................................................. 8
Table 8 Business People .................................................................................................... 8
Table 9 Comparison between SMART on FHIR and FHIR server ................................ 17
Table 10 FHIR resources as a set of operations .................................................................. 30
Table 11 Comparison between three main push notification services ......................... 31
Table 12 Comparison between Push and Polling ................................................................. 32
Table 13 Activities in Web App .......................................................................................... 33
Table 14 Activities in FHIR Service .................................................................................. 33
Table 15 Activities in Mobile App ...................................................................................... 33
Table 16 Activities in Message Generator ......................................................................... 34
Table 17 Comparison between Android and iOS ................................................................. 37
Table 18 Publicly Available FHIR Servers for Testing ..................................................... 39
Table 19 Comparison between rule engines ...................................................................... 41
Table 20 Subscribe resource ............................................................................................... 49
Table 21 Update resource ................................................................................................ 49
Table 22 Receive notification ............................................................................................. 49
Table 23 Test case for not receiving notification ................................................................. 50
Table 24 Non-functional requirement verification ............................................................... 50
Table 25 Survey template for validation ........................................................................... 51
Table 26 Survey to validate the CardioNotifier ................................................................. 52
Table 27 Survey to validate the CardioNotifier (notification with summary) ............... 52
1. Introduction

In this chapter, a short introduction of this project including the relevant background information, the objective and scope of the project are given. Also, an outline of the entire report is provided in this chapter.

1.1 Context

1.1.1. Philips Research

Philips Research is one of the largest private research organizations in the world. It has many branches all over the world including USA, Brazil, Germany, UK, Kenya, India, China, and Netherlands (Eindhoven).

Philips Research Eindhoven works on innovations in Professional Healthcare and Personal Health. Philips Research combines research activities with prototype product development based on advanced engineering competencies [1].

Within Professional Healthcare, the main important topics are diagnostic imaging and minimally invasive image guided intervention, patient care, and clinical decision support.

In Personal Health, Philips is investigating technologies and solutions that stimulate a healthy life, offering optimized personal care, and exploring ways to enhance a fulfilling home and interactive lifestyle.

1.1.2. Philips contribution to cardiology

Philips provides many solutions for non-interventional and interventional cardiology such as ultrasound sound system, interventional X-ray system, ECG Algorithms, 3D ultrasound imaging, and Cardiovascular Information Management System (CVIS).

The CVIS is a software solution that provides complete patient’s cardiovascular electronic record and covers entire cardiovascular activities in cardiology department of hospital. Philips aims to offer a complete CVIS solution that will provide access to the complete patient data (electronic medical record, imaging, discrete data, and reports).

IntelliSpace Cardiovascular (ISCV) and Xcelera provide access to cardiovascular clinical information. ISCV is the next generation cardiovascular image and information management solution [2]. This software solution, providing access to relevant images and information anytime, virtually anywhere via a browser using a workstation, laptop, tablet, etc.

1.2 The Project

1.2.1. The Starting Point

Traditional cardiology applications are desktop based. Philips offers Xcelera, an integrated multi-modality image management system for cardiovascular information [3]. This provides a single access point for multi-modality image viewing, specialized clinical applications and reporting. However, during his workday the cardiologist does not have continuous access to a desktop computer: crucial time can be lost by not having fast access to the information that matters at any location in the hospital.

More recently, with the introduction of ISCV, Philips is shifting towards web-based solutions that can be used anytime and anywhere. ISCV allows the cardiologist to access important information from anywhere within the hospital and not only using a
web browser. This is still poll based: there is currently no notification mechanism to alert the cardiologist that important information is available and requires his/her attention. Therefore, a mobile solution that can notify the cardiologist of specific updates would be very valuable.

1.2.2. The Objective
The goal of this assignment is to investigate the possibility to support the cardiologist’s workflow with help of a smartphone application, as well as to discover which functionality (e.g. subscribe to a patient, subscribe to a type of information, etc.) is most valuable to get a notification to the end users. A proof of concept implementation will support the results of the investigation.

1.2.3. The CardioNotifier Application
The CardioNotifier application is designed and developed as a proof of concept in this project. The produced product should eventually be deployed on top of the ISCV infrastructure, but this is not within the scope of this assignment. This application is a smartphone application which can be used to support cardiologist’s workflow by providing important notifications anywhere and anytime from the back-end system.

1.2.4. The Scope
The project is focused on non-emergency cases at the cardiology department. A smartphone solution is not convenient for the emergency cases because of the current smartphones’ limitations (battery life, signal coverage within the hospitals, interferences with medical devices, costs, etc.). Therefore, emergency notification will be sent primarily via the traditional pager solution.

1.3 Outline
This report is organized as follows:

Chapter 2 describes and identifies the main stakeholders of the project and explanation of their interest in, influence on, and concern with the project.

Chapter 3 provides a domain analysis which gives the explanation of cardiology and non-invasive cardiology workflow, ISCV, FHIR standard and FHIR technology solutions, and a detailed explanation about SMART on FHIR solution.

Chapter 4 describes the detailed analysis of the actual problems which have to be solved in the scope of this project is given.

Chapter 5 gives feasibility analysis that discussed about issues and challenges, and risks with the mitigation strategy in the project.

Chapter 6 gives a comprehensive description of the requirements in this project. High level functional requirements and non-functional requirements are listed.

Chapter 7 focuses on the overall architecture detailed by 4+1 views. The chapter also includes alternatives for different architecture patterns and their justifications.

Chapter 8 describes detailed design including number of design alternatives and their evaluations.

Chapter 9 describes implementation details of Mobile app and Web App component and Subscription part of the FHIR service component.
Chapter 10 verifies and validates the result against the requirements provided in Chapter 6 and concludes with the validation by answering the question “Is this a useful tool for cardiologist?”

Chapter 11 concludes the result of the project and proposes the future work list for the further improvement of the project.

Chapter 12 discusses project planning and scheduling, schedule change, and its impact on the project.

Chapter 13 gives the author’s reflection on this project which includes the lessons learned throughout the project. Also, design opportunities described in Section 3.3 are revisited in here to check whether they have been carried out successfully.
2. Stakeholder Analysis

In this chapter, the main stakeholders of the project and explanation of their interest in, influence on, and concern with the project is described.

2.1 Potential End Users

At this moment, the highest potential end users are the cardiologists who are working at a hospital. Table 1 explains a stakeholder analysis of a cardiologist with whom I could validate with final result.

Table 1 Potential end user stakeholder analysis (Cardiologist)

<table>
<thead>
<tr>
<th>Interest</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Influence</td>
<td>Medium</td>
</tr>
<tr>
<td>What is important to the stakeholder?</td>
<td>Receive a notification on important patient update when he/she is not able to access into the system</td>
</tr>
<tr>
<td>How could the stakeholder contribute to the project?</td>
<td>Ask feedback and validation based on the final result</td>
</tr>
<tr>
<td>How could the stakeholder block the project?</td>
<td>By being hard to access</td>
</tr>
<tr>
<td>Strategy for engaging the stakeholder</td>
<td>Collaborate with project manager to arrange a meeting with physician to validate the final result</td>
</tr>
</tbody>
</table>

2.2 Clinical expert

An internal clinical expert is a Philips employee who knows the domain and can act as an end user proxy. This person gives input on the project when it is hard to make an appointment with cardiologist directly. Also, the clinical expert verifies a result before sending the result to the cardiologist and his main contribution is to represent the end user to the development team while developing the cardiology products like ISCV. Table 2 explains a stakeholder analysis of the clinical expert. The stakeholder’s availability has a big risk on this project.

Table 2 Clinical expert stakeholder analysis

<table>
<thead>
<tr>
<th>Interest</th>
<th>Medium</th>
</tr>
</thead>
<tbody>
<tr>
<td>Influence</td>
<td>High</td>
</tr>
<tr>
<td>What is important to the stakeholder?</td>
<td>Present the final result of the project to the end-users</td>
</tr>
<tr>
<td>How could the stakeholder contribute to the project?</td>
<td>Show innovative power of Philips to end users Provide domain knowledge of cardiology department for clinical aspects Validate intermediate/final results of the project for clinical aspect (end-users point of view)</td>
</tr>
<tr>
<td>How could the stakeholder block the project?</td>
<td>Availability of clinical expert Lack of the communication because of the different background between PDEng trainee and clinical expert</td>
</tr>
<tr>
<td>Strategy for engaging the stakeholder</td>
<td>Information and feedback meetings every two or three weeks according to his schedule Prepare very well for the meetings and improve meeting efficiency because of limited time Ask for feedback based on intermediate/final result via email if it is necessary</td>
</tr>
</tbody>
</table>
2.3 Philips Research: Project owner

The project owner represents Philips Research. The project owner has a high interest and high power in this project but does not want to be overwhelmed with too much data. Therefore, it is important to keep this stakeholder satisfied with only relevant information.

The project owner also acts as a facilitator of meetings/feedback sessions with important stakeholders, such as making an appointment with difficult-to-reach stakeholders and taking care of the blocking items. Table 3 explains a stakeholder analysis of project owner.

Table 3 Project owner stakeholder analysis

<table>
<thead>
<tr>
<th>Interest</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Influence</td>
<td>Medium</td>
</tr>
<tr>
<td>What is important to the stakeholder?</td>
<td>Explore a new marketable mobile app for cardiologist to support their clinical workflow  Working prototype that can further analysis to attract the end-users  Show innovative power of Philips Research to Philips business</td>
</tr>
<tr>
<td>How could the stakeholder contribute to the project?</td>
<td>Validate scope of the project, planning and acceptance criteria  Give feedback based on project progress from project development point of view  Give feedback based on intermediate/final result of the project from end-user point of view</td>
</tr>
<tr>
<td>How could the stakeholder block the project?</td>
<td>Availability of project owner  Shift of business interest</td>
</tr>
<tr>
<td>Strategy for engaging the stakeholder</td>
<td>Monthly project steering group meeting to discuss the project progress  Update project progress via email every week</td>
</tr>
</tbody>
</table>

2.4 Philips Research: Project Manager

The project manager represents Philips Research. He has a high interest and high influence in this project. Also, it is important for him to track project progress according to the schedule. Therefore, it is necessary to arrange a meeting with project manager every week for updating project progress. Table 4 explains a stakeholder analysis of Philips Research.

Table 4 Project manager stakeholder analysis

<table>
<thead>
<tr>
<th>Interest</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Influence</td>
<td>High</td>
</tr>
<tr>
<td>What is important to the stakeholder?</td>
<td>Successful completion of the project  Complete the project on time, no extra budget</td>
</tr>
<tr>
<td>How could the stakeholder contribute to the project?</td>
<td>Discuss scope of the project, planning and acceptance criteria  Give feedback for progress of the project from project development point of view</td>
</tr>
<tr>
<td>How could the stakeholder block the project?</td>
<td>Availability of project manager  Cannot give technical feedback</td>
</tr>
<tr>
<td>Strategy for engaging the stakeholder</td>
<td>Plan regular meeting every week to update project progress</td>
</tr>
</tbody>
</table>
2.5 **TU/e**

The Eindhoven University of Technology is responsible for the educational aspect of the project. The university supervisor is the main stakeholder in this category who should make sure that the design and documentation meet the standards of a PDEng project. Also, PDEng trainee is a main executor of the project. Table 5 and Table 6 explain a stakeholder analysis of TU/e supervisor and PDEng trainee.

**Table 5 TU/e supervisor stakeholder analysis**

<table>
<thead>
<tr>
<th>Interest</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Influence</td>
<td>Medium</td>
</tr>
<tr>
<td>What is important to the stakeholder?</td>
<td>Validate quality of the project for educational aspect Ensure a good software design/project development experience for PDEng trainee Complete the project on time Keep good relationship with Philips</td>
</tr>
<tr>
<td>How could the stakeholder contribute to the project?</td>
<td>Validate that design and documentation meet the standards of a PDEng project Give feedback on the project progress from the managerial side</td>
</tr>
<tr>
<td>How could the stakeholder block the project?</td>
<td>Availability of TU/e supervisor</td>
</tr>
<tr>
<td>Strategy for engaging the stakeholder</td>
<td>Monthly project steering group meeting to discuss the project progress Update project progress via email every week Arrange a meeting for reviewing documents/project development</td>
</tr>
</tbody>
</table>

**Table 6 PDEng trainee stakeholder analysis**

<table>
<thead>
<tr>
<th>Interest</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Influence</td>
<td>High</td>
</tr>
<tr>
<td>What is important to the stakeholder?</td>
<td>Successful completion of the project Gain software design/project development experience Complete the project on time Make a good impression at Philips Research</td>
</tr>
<tr>
<td>How could the stakeholder contribute to the project?</td>
<td>Make sure to deliver important design documentation and prototype of the project on time Organize meetings with key stakeholders to discuss the project progress (e.g., PSGM, weekly meeting) Update key stockholders with project progress</td>
</tr>
<tr>
<td>How could the stakeholder block the project?</td>
<td>Delay delivery of documentation and prototypes Lack of communication Not proactive</td>
</tr>
<tr>
<td>Strategy for engaging the stakeholder</td>
<td>NA</td>
</tr>
</tbody>
</table>

2.6 **Philips Healthcare – Business Unit, Healthcare IT, Cardiology Informatics**

This stakeholder has low interest and medium influence on the project because the mobile app would eventually be deployed on top of the ISCV infrastructure as developed by Philips Healthcare IT, but is currently in an exploratory phase. Table 7 shows the detailed analysis of this stakeholder.
Table 7 Philips Healthcare - Business Unit Cardiology Informatics Section stakeholder analysis

| Interest | Low |
| Influence | Medium |
| What is important to the stakeholder? | No or minimal infrastructure (like database) changes required; potential to eventually integrate and deploy on systems such as ISCV |
| Require less time and effort |
| How could the stakeholder contribute to the project? | Provide knowledge of ISCV platform |
| How could the stakeholder block the project? | Not able to provide ISCV knowledge |
| Strategy for engaging the stakeholder | Availability of administrator of ISCV |
| Get the contact details of ISCV architect/expert from the project manager |

2.7 Business people

This stakeholder has low interest and medium influence on the project because the mobile app should potentially be able to be integrated with commercial systems such as ISCV. Table 8 shows the detailed analysis of this stakeholder.

Table 8 Business People

| Interest | Low |
| Influence | Medium |
| What is important to the stakeholder? | Improve current project further |
| Look for a new business opportunity |
| How could the stakeholder contribute to the project? | Give feedback based on the final results of the project |
| Provide financial support for the further development of the project |
| How could the stakeholder block the project? | NA |
| Strategy for engaging the stakeholder | Collaborate with Roel Truyen to arrange a meeting with business people when the final result of the project is ready |

2.8 The stakeholder mapping

Figure 1 shows stakeholders who are affected by this project and classifies them by their influence and interest. There are many stakeholders included in this project. The stakeholder mapping (Figure 1) is very important to identify key stakeholders at early stage of the project. Also, it is possible to analyze which stakeholder can improve the quality, support or block the project.

Furthermore, the project manager has a great interested and also high influence in this project. Therefore, it is important to be fully engaged with him.
Figure 1 The stakeholder mapping

2.9 Conclusion

Stakeholder analysis helped to understand whom to discuss about cardiology and whom I can ask support on Philips CVIS. Also, clinical expert and project manager are good contact point to discuss about project domain. The next chapter explains the detailed domain analysis which was defined with help of the key stakeholders.
3. Domain Analysis

In this chapter, the introduction of basic terms that are needed to get a better understanding of the report is given. This is composed of a short introduction of Cardiology and non-invasive cardiology workflow, ICSV, FHIR standard and FHIR technology solutions, and a detailed explanation about SMART on FHIR solution.

3.1 Cardiology

Cardiology is the department of the medicine that studies about heart disease and its treatment. Cardiologists are doctors who specialize in diagnosing and treating diseases or conditions of the heart and blood vessels the cardiovascular system [4].

During the diagnosis, the cardiologist has to consider many aspects of the patient cardiovascular system such as heart, arteries and blood vessels. To structure the patient’s health condition, the cardiologist checks up patient’s medical records, tests blood pressure, vessels, lung functions, also other specific tests concentrating on the health problems.

There are three types of cardiology: invasive, non-invasive, and interventional.

Invasive cardiology causes to fully or minimally open surgery that helps to identify the heart abnormalities or treat them [5]. Cardiovascular surgery is surgery on the heart or great vessels performed by cardiac surgeons. Also, coronary artery bypass graft (CABG) surgery is a surgical procedure to restore normal blood flow to an obstructed coronary artery.

Non-invasive cardiology, in the other hand, focuses on the detection and treatment of heart disease, using external tests rather than instruments inserted into the body to evaluate and diagnose cardiac disorders [6].

- Nuclear cardiology: A non-invasive study of cardiovascular disorders may use radioactive elements in order to create different types of imaging.
- Echocardiography: To identify how condition of the heart pumps blood, infections, and structural abnormalities are, the echocardiography uses ultrasound waves to model out images of the heart and surrounding structures.
- Cardiac electrophysiology: Study and testing of the electrical currents which generate heartbeats.
- Stress tests: The stress testing is usually the exercise under cardiologist monitoring. With the result of this test, it provides the cardiologist how patient’s heart act under physical stress.
- Heart monitors: Heart monitors can also be named as a Holter monitor or cardiac event recorder. Heart monitors particularly record patient’s heart electrical activity for selected amount of time.
- CT scans: Computed tomography scan creates images that helps the cardiologist identify for the heart disease and atherosclerosis.

Interventional cardiology is a non-surgical option which uses a small, flexible tube called catheter to repair the heart problems which can be damaged vessels, narrowed arteries [7].

- Coronary artery disease: Arteries become narrow which affects to the supply of the heart muscle with blood and oxygen.
- Heart valve disease: This disease occurs as the valves that manage blood flow into the heart’s chambers are functioning abnormally.
Peripheral vascular disease: Peripheral vascular disease is caused by clogged or hardened veins and arteries.

In the cardiology department, there are several other specialists except cardiologist: cardiology specialist registrars, cardiology nurses and physiologists. Specialist Cardiac Nurses can be specialized in heart failure, catheter lab and cardiac rehabilitation. These specialists provide the cardiac patients with the diagnosis and the treatment. After the invasive, non-invasive or other procedures (pacemaker fittings), some cardiac patients are urgent to have rehabilitation programmers and advices of lifestyle and diet.

This project focuses on non-invasive cardiology because of the following discussion with clinical expert.

- For the physician who ordered the test, it is difficult to know when the patient’s preliminary or finalized echocardiogram report will be ready. The sooner they get the information, the earlier they can make a decision for patient’s medical treatment. The way to know if the information is ready is either to log into the system and check the status or ask the nurses, which affect the work efficiency.

3.1.1. Non-invasive workflow

A non-invasive cardiology workflow has six phases: pre-acquisition, cardiology consultation, data acquisition, reporting and treatment and after care (Figure 2).

![Non-invasive cardiology workflow](image-url)
• Pre-acquisition

The pre-acquisition phase begins when the patient visits the general practitioner (GP) with a complaint such as chest pains, fatigue, or difficulty in breathing. If the GP physician determines that cardiac testing is required, the patient is referred to the non-invasive Cardiology Department of a hospital.

• Cardiology consultation

Cardiologist review patient’s medical history and perform a physical examination which may include checking blood pressure, weight, heart, lungs, and blood vessels. Some problems may be diagnosed by patient’s symptoms and the cardiologist’s findings when patient is examined.

• Data acquisition

During the data acquisition phase, the patient arrives at the hospital and take some physical tests for non-invasive diagnostics.

• Reporting

During the reporting phase, the echo-cardiologist studies the images, measurements, and ECG and dictates, summarizes a report. Then the report is sent to the cardiologist.

• Treatment

After receiving the summarized reports, the cardiologist or cardiac surgeon makes a treatment plan and treats the patient according to the plan.

• After care

During the aftercare period, cardiologist or general practitioner follow up with the patient and monitor patient’s condition.

Cardiologists are involved with many other activities at the hospital such as visiting patient rooms, meetings with nurses/another physician, working in the emergency room, and performing medical examinations.

In most cases, they work outside of their office. Currently, if cardiologist needs to check patient information he/she would have to find workstation and login to the system in the hospital. They receive important information using phone call in some demanding cases. For example: when patient diagnosis result is ready, they either get notification from nurse by phone call or check by themselves from Cardiovascular Information Management System (CVIS).

Furthermore, all other activities in cardiology department of the hospital are covered by the CVIS. The next sections explain the Philips CVIS solutions and alternatives for exploratory research activities such as SMART on FHIR, and FHIR servers.

3.2 Philips Cardiovascular Information Management System (CVIS)

Philips Cardiovascular Information Management System (CVIS) provides a cardiologist access to all relevant patient data: patient medical record, imaging data (patient medical history), discrete data (measurements, observation results) and reports.

The CVIS receives patient’s data from integrated sub systems such as echo machines, EKG/ECG carts and third-party integrated monitoring systems. Also, all remaining measurements and nurse charting captured in the course procedures are stored within the CVIS database for patient record and reporting.

Currently, Philips offers software solutions which are ISCV and Xcelera to provide access to cardiovascular clinical information. Comparing to Xcelera, ISCV is a web based cardiovascular image and information management solution which makes it next generation technology. In the following section ISCV is described.
3.2.1. **IntelliSpace Cardiovascular (ISCV)**

ISCV is a comprehensive cardiac image and information management solution designed to provide clinicians with convenient access to the detailed records of cardiac patients across their complete cardiovascular care continuum [8].

Currently, ISCV offers access to the patient imaging history, the reports, and other relevant information via different plugins/add-ons. In order to offer access to more and more patient discrete data coming from various sources and in the move towards a complete CVIS solution, Philips is looking into possibilities for standards to use.

In this project we selected FHIR as a standard to work with, since it is an emerging and modern standard, endorsed by HL7. This selection enables us to use a standard off the shelf FHIR server for testing purposes, which reduces dependency on the current product, will allow independent rapid prototyping and is expected to be future-proof.

### 3.3 Fast Healthcare Interoperability Resources (FHIR)

Fast Health Interoperable Resources (FHIR, pronounced "Fire") is a next generation standards framework created by Health Level Seven International (HL7) organization. FHIR combines the best features of HL7’s Version 2, Version 3 and CDA product lines while leveraging the latest web standards and applying a tight focus on implementability [9].

FHIR solutions are built from a set of modular components called “Resources” [2]. These resources can easily be assembled into working systems that solve real world clinical and administrative problems at a fraction of the price of existing alternatives. FHIR is suitable for use in a wide variety of contexts such as mobile phone apps, cloud communications, EHR-based data sharing, and server communication in large institutional healthcare providers [10].

#### 3.3.1. Collaboration with FHIR

The FHIR specification is owned by HL7 which making the FHIR specification freely available to the FHIR community. The FHIR community is very large and distinguished. There are representatives from institutes and companies such as athenaHealth, The Mayo Clinic, McKesson, Partners HealthCare Systems and many more leaders in the healthcare industry.

One of the main tenets of the FHIR specification is that it is to be free for use with no restrictions to ensure the broad adoption, increased interoperability, and fast and easy implementation as everyone uses and contributes to the specification.

#### 3.3.2. FHIR Technologies

FHIR defines a set of "Resources" that represent granular clinical concepts. The resources can be managed in isolation, or aggregated into complex documents. Technically, FHIR is designed for the web; the resources are based on simple XML or JSON structures, with an http-based RESTful protocol where each resource has predictable URL. Where possible, open internet standards are used for data representation [11].

---

2 [https://www.hl7.org/fhir/resourcelist.html](https://www.hl7.org/fhir/resourcelist.html)
The FHIR Service will check the standard compliance and transfer the message between front-end and back-end. FHIR data access framework will also check the standard compliance of the data coming from back-end service and update or retrieve the database.

- **Advantages:**
  - The integration with future versions of commercial systems such as ISCV which follow standards such as FHIR will be easier.
  - FHIR push mechanism can be used instead of developing a new one.

- **Disadvantages:**
  - The demonstration for end user cannot be as realistic as a demonstrator showing real clinical data examples connected to current commercial systems such as ISCV.

There are many free to use versions of FHIR such as HAPI-FHIR and SPARK-FHIR which are implemented on the languages like Java and C#. These versions share information between client and server in XML and JSON formats and allow the definition of custom formats on top which gives much more flexibility.

### 3.4 SMART on FHIR

SMART Health IT is an open, standards based technology platform that enables innovators to create apps that seamlessly and securely run across the healthcare system. Using an Electronic Health Record (EHR) system or data warehouse that supports the SMART standard, patients, doctors, and healthcare practitioners can draw on this library of apps to improve clinical care, research, and public health [12].

SMART on FHIR defined a way for health apps to connect to EHR systems with appropriate security guarantees. In addition to FHIR models and API, components include authorization, authentication, and app integration [13].

The physical point of view there are three servers working according to the SMART on FHIR solution which are API Server, Authorization Server, and App Server.

- **API Server**
  The API server supports create, read, update, and delete operations for all FHIR resources and implements the FHIR search API.

- **Authorization Server**
  Web standard Authorization that OpenID and OAuth2 were used for authorization in this server. Its key function is to enable an end user to approve a third party app to access a specific set of data from API Server.
16

- **App service**
  The app server exposes an EHR-like environment for developers to browse a patient list and launch apps on a given record. The environment is a SMART on FHIR web app written in HTML5 and JavaScript [14].

This solution is using SMART on FHIR. This is an extension of Solution 2 having a SMART container in front-end which helps to arrange the new and existing applications. Also, this solution provides authorization service between back and front ends (Figure 4).

The advantage of using this solution is:
- It gives more flexibility for application developers. They do not need to worry about back-end and simply develop their application which can work with diverse EHR systems.

The disadvantage of the solution is:
- Using SMART on FHIR will not be easy for developers because of the poor documentation.
- It cannot give a convincing result for the end user.

Furthermore, it is possible to deploy the above servers on MAC, Windows and Ubuntu 14.04 Server machines. Also, the SMART on FHIR team used Vagrant and Ansible tools for the installation by doing all the configuration automatically.

### 3.4.1. Who is involved

The SMART Health IT Project is run out of not-for-profit institutions, Boston Children’s Hospital Computational Health Informatics Program and the Harvard Medical School Department for Biomedical Informatics.

Additionally, through a project called Argonaut, some of the largest EHR vendors (Cerner, Intermountain Healthcare, Harris Corporation, and Hewlett-Packard) have joined forces with the SMART team and the HL7 organization to build SMART into the releases of their products, and to standardize the SMART API in HL7 specifications.
3.4.2. Why use SMART on FHIR

The key focus of SMART is to develop a SMART platform architecture to achieve two major goals:

- Develop a user interface to allow substitutability for medical apps based on basic shared components
- Create a set of services to enable efficient data capture, storage, retrieval and analytics.

In another words, the SMART is an ecosystem of substitutable apps that can run on any EHR system.

3.4.3. How is the support?

There are plenty of tutorials and documentations online. Also, more support is provided via email, google group, and GitHub configuration or development issues. Josh Mandel is a lead architect for SMART Platforms. He is very supportive and responds to the questions posted on GitHub or google email group really fast.

3.5 Conclusion

There are three possible infrastructures, ISCV, SMART on FHIR, and FHIR server, for this project. ISCV is the most compatible back-end solution because it is provided by Philips which makes it consistent for future extension. However, commercial systems are not ideal for making demonstrators in exploratory research activities. Therefore, we needed to select one of the remaining two infrastructures for proof of concept. Table 9 shows comparison between SMART on FHIR and FHIR server.

<table>
<thead>
<tr>
<th></th>
<th>SMART on FHIR</th>
<th>FHIR servers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pros</strong></td>
<td>Authorization Service</td>
<td>Easy to deploy</td>
</tr>
<tr>
<td></td>
<td>Easy Integration using Smart Container</td>
<td>Easy integration</td>
</tr>
<tr>
<td></td>
<td>Use of different data repository</td>
<td>Open test servers are available</td>
</tr>
<tr>
<td></td>
<td>Open source</td>
<td>Good documentation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Easy to test</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Multiple choices for server technologies such as Java, C#</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Open source</td>
</tr>
<tr>
<td><strong>Cons</strong></td>
<td>Lack of documentation</td>
<td>No authorization implemented</td>
</tr>
<tr>
<td></td>
<td>More complex</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No client library for Android</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not easy to test</td>
<td></td>
</tr>
</tbody>
</table>

In conclusion, FHIR server is chosen for this project because it was suitable for creating proof of concept with its advantages such as easy to deploy, integrate, and test.
4. Problem Analysis

In this chapter, the detailed analysis of the actual problems which have to be solved in the scope of this project is given. Finally, the design opportunities to solve those problems are described.

4.1 Problem descriptions

There are four major problems: mobile cardiologist versus a fixed workstation, filtering important information, cumbersome information sharing, and limitations of already existing tools. There are described in detail in the following sections.

4.1.1. Mobile cardiologist and a fixed workstation

The cardiologists at the hospital have their own fixed working place with a workstation connected to the hospital information system. He/she can also use any other workstation located in different places and access the information system by logging in. Using the system, the cardiologist can check or update the patients’ status. However, the cardiologists are involved with many other activities at the hospital such as visiting patient rooms, meetings with nurses/another physician, working in the emergency room, and performing medical examinations (Figure 5).

Most of the time, the physicians are not in a fixed location which makes difficult for them to access the patient status (e.g., a patient’s diagnosis and status of a report). In order to check the latest patient’s status, they can either ask the nurses or find the closest workstation and login to the hospital information system, which makes their work less efficient.

Currently, to retrieve information updates, the cardiologist needs to poll the system for information by asking or checking the information. Depending on the timing of
the update and the request, this may sometimes be successful, and sometimes not. Ideally, important notifications are pushed by the information system to cardiologist because it is not known to the user when the information becomes available in the system; therefore, he/she does not know when to poll the information.

Furthermore, cardiologists still carry pagers for receiving messages from a hospital’s pager system. However, a pager solution is designed for transmitting critical short textual messages. This solution is only available to use in a hospital area.

In conclusion, smartphone can be suitable device to receive notification because cardiologist can keep it everywhere and anytime instead of tablet or laptop. Comparing with pager, smartphone can get more information (e.g., more text, picture) and also possible for cardiologist to receive a notification out of hospital area if device is connected to the internet.

4.1.2. Filtering important information
Every day a cardiologist needs to manage up to 10 patients in a big hospital and each patient’s data is updated approximately 10 times per day. If cardiologist get a notification when patient data has been updated, cardiologist will receive at least 100 notifications. The cardiologist does not want to be overwhelmed with all that information. Therefore, it is important to find out exactly which information is valuable for them and when, and only notify them about updates of this information.

One of the valuable kinds of information can be a patient’s medical test result (e.g., echocardiogram), this is needed for decision on treatment. Therefore, knowing test result as soon as possible is needed to improve the working efficiency of the cardiologist and quality of the patient management.

In this project, we chose the echocardiogram testing procedure as an example case because it is one of the most common examinations for a patient to diagnose heart disease. In an echocardiogram procedure ultrasound is used to create a visual image of a patient heart. This helps the physician to see the internal structure of the heart and to know how well the blood is able to flow through it. Poor flow has to be treated as early as possible.

4.1.3. Limitations of already available tools
The above problems could be solved using the already existing powerful information sharing tools such as Snapchat, Viber, and WhatsApp. However, that is not allowed at the hospitals because of confidentiality issues. Since the information exchange refers to actual patient data, privacy and confidentiality are important concerns and are strictly regulated within the hospital environment. Therefore, an application specifically created for the hospital that can provide enough security and privacy is needed.

4.2 Design Opportunities
Based on the problem statement described in the previous section, the assessment criteria for the technological design that is most appropriate for this project can be identified. According to the “New criteria for assessing a technological design” by Kees van Hee and Kees van Overveld [15], the following design opportunities were picked out as the important ones for this project. These criteria are revisited and reflected on at the end of this report.
Important design criteria for this project:

Functionality
- Ease of use: the final product has to be easy to use by end-user (e.g., cardiologist). Therefore, it requires user friendly interface design without using manual.
- Maintainability: easy to maintain the functionalities by IT specialist

Construction
- Convincingness: The result of the project has to provide a concrete demonstrator of the idea of using a smart phone application to improve cardiologist’s work efficiency. The end user is generally very practical, hard to get valuable feedback from abstract idea.
- Genericity: Is it reusable in another context, best practices.

Realizability
- Technical realizability: The investigation of the technical possibility to use FHIR and SMART on FHIR on the developed product is needed.

However, the rest of the design criteria listed below are less important in this project:

Construction
- Inventivity

Realizability
- Economical realizability

Security
- Security is one of the important aspects for providing clinical information using mobile devices. Therefore, security and privacy are the highest priority in future development. However, it is not essential at the current stage in this project because of time frame.
5. Feasibility Analysis

In this chapter, the introduction of the three possible solutions of implementing CardioNotifier application and their feasibilities are detailed. Based on the result of the feasibility analysis, the initial design decision was made.

5.1 Technical challenges

5.1.1. Smartphone network instability
Since smartphones are light and portable, it always changes its location which makes the network connection instable. Therefore, it is difficult to maintain stable connection with a server or other applications. Because of this, the critical information should not be lost somewhere. Therefore, design solutions should be designed that are robust against this network instability.

5.1.2. New technologies
The solution leverages new technologies and frameworks with which the author has no experience. Becoming familiar with them as well acquiring the broad understanding that this project requires is a challenge. Significant effort is needed to make up for those shortcomings.

5.1.3. Immature technologies
Some of the technologies to use in this project such as FHIR and SMART on FHIR are currently not mature enough to provide what they aim to offer. However, Philips is willing to know how these concepts of these technologies can help to solve the problems in healthcare information system development. Therefore, during the architecture and design phase, the project focused on the concepts of the technologies to give the impression about how this technologies can be used to aid our goal when it is fully functioning. After having an ideal overall architecture and design, the implementation of the non-feasible parts was done using any other alternative methods to achieve final demonstrator prototype very quickly.

5.1.4. Unfamiliar domain: Healthcare IT
The author was not familiar with the domain, medical cardiology, of this project. Having sufficient domain knowledge is important to analyze the requirement correctly. Therefore, it was important to learn the domain quickly by communicating with the stakeholders frequently.
  
  o Research on medical domain
The biggest challenge in this project was to find out how and in which cases the smart phone can support the cardiologist’s workflow while he/she is working out of the office. Since the author does not have any clinical domain knowledge, organizing enough discussions with a clinical expert is needed.

  o Restricted contact with end user
In this project, the chance to contact a cardiologist was zero and a contact with a clinical expert is limited. Therefore, determining the end user’s wish was challenging. Therefore, careful planning with clinical expert was very crucial.
5.2 Operational challenges and risks

The following is a short overview of the organizational challenges and risks that were identified early. Also, mitigation strategies to limit their impact on the project are described.

5.2.1. Communication with stakeholders who have different backgrounds

Some of the stakeholders have no IT background people. Therefore, it is challenging to communicate with them efficiently. Extremely detailed discussion should happen in a very easy to understand way.

5.2.2. Integration with ISCV

The produced product will eventually be deployed on top of the ISCV infrastructure. However, there is high risk in relying on support for commercial products like ISCV during exploratory research activities. Thus, a backup plan is needed. To reduce dependency toward the existing product platform, an off the shelf FHIR server can be used instead of the ISCV backend.

5.2.3. New requirement or requirement change at the last moment

Prototyping makes more problems clear, which leads to new requirements that the customer was not aware of at the beginning. If this happens very late in the project, better to make a feasibility analysis by prioritizing all the remaining tasks and allocating them to the remaining period. If the new requirement is feasible within the timeframe, it is possible to accept it. Otherwise, offer to put it in the future work list.

5.2.4. Demo failure

A successful demonstrator is critical to satisfy customer’s needs. To achieve that, it is better to avoid last moment changes in the system. Also, repeated demonstration rehearsal is needed. Furthermore, a backup demonstration environment/laptop is needed in case of any hardware or software failure.

5.2.5. Inability to contact the clinical expert

Clinical experts have busy agendas; free access to them is not guaranteed. Therefore, early scheduling with them is extremely crucial. If the clinical expert becomes unavailable, then a replacement should be found.
6. System Requirements

In this chapter, the detailed system requirements are discussed. The chapter is divided into two parts; functional and non-functional. The section for functional requirements covers high-level functional requirements in this system. The section for non-functional requirements explains all the quality attributes.

6.1 Functional requirements

The following are the high level functional requirements of this project. The detailed design of these requirements is discussed in the architecture and design section.

6.1.1. Subscribe to the information to be notified
- The user should be able to subscribe to the following information:
  - Patients
  - Type of resources (e.g. Observations (Weight measure, Blood pressure measurement, Temperature, EKG etc.), Diagnostic report, or Diagnostic order)
- Subscription should be done by workstation because it is much easier to see detailed information on a workstation monitor than on a mobile phone screen.
- Subscription for a group of patients should be possible in one action
- The subscription should be stored in a database

6.1.2. Receive a notification
- The user should be able to receive the notification of the subscribed information on his/her smartphone anytime in both inside and outside the hospital area.
- User should be able to receive any text message as a notification from other hospital staffs.
- The notification should be retrieved from database.
- History of the notifications should be reachable for the user
- Context of a notification should be configurable
- Notification should be generated automatically when subscribed information has been updated in the system

6.2 Non-functional requirements

The terminologies used in the non-functional requirement are chosen based on the “ISO SQuaRe Product Quality” document [16].

6.2.1. Availability
- The application should be available 24 hours a day, seven days a week
- User should be able to access the application and receive notification anytime anywhere as long as the smartphone is up and running with available network connection

6.2.2. Reliability
- No data should be lost during the traffic
- Unread notifications should be shown with a highlight until it has been checked.
6.2.3. **Performance efficiency**

It will be extremely inconvenient for the busy cardiologist if the system performs very slowly. Therefore, a high performance solution is needed for the final product. User should receive the notification as soon as the information is updated.

6.2.4. **Usability**

In the smart phone application, the user needs to be able to receive the notification in an intuitive way. In general, this requirement can be judged from the following aspects:

- The users can accomplish their tasks with minimal effort and no redundant procedures are required.
- The application should be easily accessible and the notifications should be easily seen.
- The new notifications should be highlighted until it’s checked.
- The application should not distract the cardiologist with loud sound effects while he/she is being busy with a patient etc.
- The application should give a reminder for the user if he/she does not check the new notifications timely.
- Easy to use for non-IT people to receive a notification and process (Minimum time for introduction)
- Minimum steps to check content of a notification

6.2.5. **Security and privacy**

Since the application deals with very private information, the patient’s medical records, this should be handled in a secure way.

- Follow the CVIS security and privacy for the integration
- Follow the rules for mobile device and health information privacy and security such as HIPAA (The Health Insurance Portability and Accountability Act) Privacy Rule

6.2.6. **Standard compliance**

The system should follow FHIR standard.

- FHIR server should be used in back-end
- FHIR standard should be used transferring information between back-end and front-end

---

3 [http://www.hhs.gov/hipaa/for-professionals/privacy/](http://www.hhs.gov/hipaa/for-professionals/privacy/)
7. System Architecture

In this chapter, the overall architecture of CardioNotifier, along with the important decisions and their justifications is covered. The detailed design is discussed in next chapter.

7.1 Introduction

This chapter provides a comprehensive architectural overview of a CardioNotifier application of the Solution 2 which was chosen based on the feasibility analysis in Section 5.1.

The architecture of the solution is detailed according to 4+1 architectural views: logical, development, process, physical, and use case in order to depict different aspects of the design.

7.2 Use case

The use-case view captures the functionalities of the Mobile app as seen from the user’s point of view. Figure 6 provides an overview of the essential steps a user would follow to create.

![Figure 6 Use case view](image)

<table>
<thead>
<tr>
<th>7.2.1. Use case 1: Receive notification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Actor: Cardiologist</td>
</tr>
<tr>
<td>Context of use: A cardiologist wants to get a notification about patient updated information</td>
</tr>
<tr>
<td>Scope: Mobile app &amp; CardioNotifier</td>
</tr>
<tr>
<td>Precondition: Subscription is configured</td>
</tr>
<tr>
<td>Success Guarantees: Notification successfully received</td>
</tr>
<tr>
<td>Main Success Scenario</td>
</tr>
<tr>
<td>1: Mobile app: Receive all notifications</td>
</tr>
<tr>
<td>2: Mobile app: Show total number of notifications in popup window</td>
</tr>
</tbody>
</table>
### 7.2.2. Use case 2: Check notification

**Primary Actor:** Cardiologist  
**Context of use:** A cardiologist checks the received notification  
**Scope:** Mobile app & CardioNotifier  
**Precondition:** Notification has been delivered on Mobile app  
**Success Guarantees:** Notification successfully received  

**Main Success Scenario**
1: Cardiologist: Press the notification on popup window  
2: Mobile app: Display detailed information of notification  

### 7.2.3. Use case 3: Configure subscription

**Primary Actor:** Cardiologist  
**Context of use:** Subscribe patient information  
**Scope:** Web App  
**Precondition:**  

**Success Guarantees:**  

**Main Success Scenario**
1: Cardiologist: Select patient(s) from the patient list  
2: Cardiologist: Press the subscribe button  
3: CardioNotifier: Store data of selected patients  
4: Cardiologist: Select type of patient information from the list  
5: Cardiologist: Press the subscribe button  
6: CardioNotifier: Store data of selected type of patient information  
7: Cardiologist: Select threshold of notification number  
5: Cardiologist: Press the save button  
6: CardioNotifier: Store data of threshold of notification number  

### 7.2.4. Use case 3: Send notification manually

**Primary Actor:** Nurse  
**Context of use:** Send a notification manually from web  
**Scope:** Web App  
**Precondition:** Patients info page is opened  
**Success Guarantees:** Send a notification successfully  

**Main Success Scenario**
1: Nurse: Type the content of the notification  
2: Nurse: Press the send button  
3: CardioNotifier: Store content of the notification  
3: CardioNotifier: Send the notification  

### 7.2.5. Use case 3: Check rules

**Primary Actor:** IT Technical Supporter  
**Context of use:** Set rules  
**Scope:** Rule engine  
**Precondition:**  

**Success Guarantees:** Set rules successfully  

**Main Success Scenario**
1: IT Technical Supporter: Set rules on the editor
7.3  Logical View
The logical view by contains the logical components of the CardioNotifier and their interactions. This view explains more about how the components interact with each other in details which helps to specify the functional requirements of the system.

7.3.1. Context
Generally, FHIR is designed for the web technology with http-based RESTful protocol where each resource has predictable URL. Therefore, the RESTful API interface can be used to transfer the data between CardioNotifier and FHIR back-end.

7.3.2. CardioNotifier with FHIR
Several key requirements steer the decision for an architectural pattern and later design of the system. One of the main requirements in the project (see chapter 6) is for it to be a mobile application available for a cardiologist using FHIR back-end.

The mobile application and FHIR back-end service follow the client-server architectural style, where clients make requests to server. The client-server architectural style also describes as two-tier: front-end and back-end (Figure 7).

o  Front-end
The front-end tier consists of the user specific application logic and the user interface. In this project, web app and mobile are front-end tier.

o  Back-end
The back-end tier manages business logic and storage such as FHIR service, FHIR database, and message generator in this project.

Figure 7 CardioNotifier with FHIR Server

7.3.3. Security and privacy
Authorized access and confidentiality requirements can be implemented on top of the system. There are already existing technologies available for this purpose such as Geneva framework, SSL, OpenID, and OAuth. Geneva framework is currently used on ISCV and OAuth is recommended by FHIR. Therefore, OAuth is the best candidate at the moment considering the fact that the SMART on FHIR is already integrated with OpenID and OAuth2.
The using SMART on FHIR can be the best choice to go since we do not need to deploy and configure a server for authorization which is extremely time consuming. However, SMART on FHIR does not have the client library for Android native app yet which hinders the direct usage of the platform.

### 7.4 Development view

The development view focuses on the actual module of CardioNotifier within the software development environment. This CardioNotifier is expressed through packages or libraries organized in a way that they provide interfaces between main components.

#### 7.4.1. FHIR Server

Table 8 shows FHIR resources which are mainly designed for the RESTful HTTP based implementation. RESTful API provides an exchange of resources using a service-based approach.

**Figure 8 The Architecture of FHIR Server**

FHIR API describes the FHIR resources as a set of operations (Table 10) on resources where individual resource instances are managed in collections by their type [17].

**Table 10 FHIR resources as a set of operations**

<table>
<thead>
<tr>
<th>Operations</th>
<th>Explanations</th>
</tr>
</thead>
<tbody>
<tr>
<td>create</td>
<td>creates a new resource in a server-assigned location</td>
</tr>
<tr>
<td>read</td>
<td>accesses the current contents of a resource.</td>
</tr>
<tr>
<td>update</td>
<td>updates for an existing resource for the given id</td>
</tr>
<tr>
<td>delete</td>
<td>removes existing resource</td>
</tr>
<tr>
<td>history</td>
<td>retrieves the history of either a particular resource</td>
</tr>
<tr>
<td>search</td>
<td>searches a set of resources based on some filter criteria</td>
</tr>
<tr>
<td>vread (read version)</td>
<td>reads the state of a specific version of the resource</td>
</tr>
<tr>
<td>validate</td>
<td>checks whether the attached content would be acceptable</td>
</tr>
</tbody>
</table>

#### 7.4.2. FHIR RESTful API

The RESTful API defines a set of interactions (e.g., read, update, create, delete) performed on a repository of typed resources. These interactions follow the RESTful of managing actions such as read, update, delete, and create on a set of identified resources.

The interactions are defined like this:
VERB [base]/[type]/[id] {?_format=[mime-type]}

- VERB: HTTP (POST, GET, PUT, and DELETE).
- base: service root URL
- mime-type: application/fhir+json or application/fhir+xml
- type: resource type (e.g. Patient, Observation, etc.)
- id: logical id of a resource
- vid: version id of a resource
- compartment: name of a compartment
- parameters: URL parameters as defined for the particular interaction

7.4.3. Interface between Mobile App and FHIR service

The interface between Mobile App and FHIR service can be implemented in either pushing or polling styles. Web socket is the base technologies for push mechanism and polling is an implementation underneath the Web service.

- **Polling** uses request-response protocol HTTP. The client submits an HTTP request message to the server and the server returns a response message to the client. If there is not anything to respond, the server just returns an empty message. However, each HTTP message contains a header part which has to be created on every request-response connection. This header creation process affects the performance of the polling mechanism tremendously [18].

- **Push** notification is the delivery of information from a software application to a computing device without a specific request from the client. The advantage of push notification is that the technology does not require an application on a mobile device to be running in order for receiving message [19].

7.4.4. Push notification service

Push notification service allow to send messages to users who have installed the mobile application from server. Most mobile applications are executed only a few times after installed. However, a push notification service provides relevant information to customers when the mobile app is closed. There are the following advantages:

- Engaging users: providing valuable and relevant updates to customers, even when mobile app is closed.
- Ease of use: messages that provide only the essential information, making user friendly and effective.
- Fast delivery: cloud vendors (e.g., Google Cloud Messaging, Apple Push Notification service) provide real-time notifications delivery.

Furthermore, there are three main technologies for pushing services:

- Google Cloud Messaging (GCM)/Firebase Cloud Messaging (FCM) for Android
- Apple Push Notification service (APNs) for iOS
- Microsoft Push Notification Service (MPNS) for Windows

Besides that there is also a lot of push notification services using these technologies such as Amazon Simple Notification Service, Parse.

<table>
<thead>
<tr>
<th>Table 11 Comparison between three main push notification services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feedback Service</td>
</tr>
<tr>
<td>---------------------</td>
</tr>
<tr>
<td>Message Expire</td>
</tr>
<tr>
<td>Bi-directional</td>
</tr>
<tr>
<td>messaging</td>
</tr>
</tbody>
</table>
Supporting mobile platforms

<table>
<thead>
<tr>
<th></th>
<th>iOS</th>
<th>Android</th>
<th>Windows Phone devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store and Forward</td>
<td>Stores only unsent message</td>
<td>Stores multiple based on collapse keys</td>
<td>Stores unsent message</td>
</tr>
<tr>
<td>Payload format</td>
<td>JSON</td>
<td>JSON</td>
<td>XML</td>
</tr>
<tr>
<td>Payload Size</td>
<td>2KB</td>
<td>4KB</td>
<td>3KB</td>
</tr>
</tbody>
</table>

It is not possible to send patient information, and notification using GCM because of the confidentiality. Therefore GCM used only for informing CardioNotifier there is a new notification. After that CardioNotifier request to get patient information and context of the notification.

### 7.4.5. Polling or Push Notification service?

Using Push Notification Service can give many benefits as it can qualify all the advantages detailed in above section and in Section 7.4.4, plus, it can give better performance and reliability. To find out which technology will suit better in our situation, we made a comparison between push and polling in Table 12.

**Table 12 Comparison between Push and Polling**

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Push</th>
<th>Polling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication direction</td>
<td>Bi-directional. Both a client and a server can start to talk independently once the connection is established.</td>
<td>Single-directional. A client needs to use polling to get the response, or the client provides a web service URL for calling back.</td>
</tr>
<tr>
<td>Base Technology Implementation</td>
<td>TCP, XMMP, HTTP</td>
<td>HTTP</td>
</tr>
<tr>
<td>Performance</td>
<td>Low latency</td>
<td>High latency</td>
</tr>
<tr>
<td>Implementation</td>
<td>More complex.</td>
<td>Simpler. Use URLs and wrap existing functionalities.</td>
</tr>
<tr>
<td>Network traffic</td>
<td>Only necessary messages are transferred.</td>
<td>High latency. If a client needs response from a server, there are many redundant messages due to the polling mechanism.</td>
</tr>
</tbody>
</table>

During interview with a clinical expert we realized that a cardiologist does not want to be overwhelmed with high number of notifications; instead, they want to receive only the relevant information. This means the cardiologist is not likely to receive more than one notification per day depending on his/her number of subscription in general cases. Therefore, using polling mechanism in this case will cost a lot of redundant traffic which will cause a network overload and a faster battery loss.

Since, the performance efficiency is an important requirement (Section 6.2.3), the push notification is chosen for this project as a better architectural solution. However, implementation wise, it is bit complicated requiring longer learning curves. Therefore, since our goals is fast prototyping, we decided to choose implement polling mechanism which is much simpler and faster to implement. For future improvement, we highly recommend to consider the Push Notification service.

### 7.5 Process view

After a general description of the five core components, the overall activity diagram is shown in Figure 9. It illustrates the action flows in the system as well as the interactions among the three components.
7.5.1. Activities between components

There are five main components which are Web App, FHIR Service, Message Generator, and Mobile App. The end-user can subscribe to patient information from the Web App. After that Message Generator checks every resource that is created or updated, and if the resource matches the subscribed information, it generates a notification to the database. At last, the Mobile App shows received notification to the end-user. The main components activities are explained in next section except database.

Figure 9 shows the activity diagram groups actions by the components. A block represents an action which is provided by one component. An arrow between two blocks points a flow direction.

7.5.2. Activities in Web App

In Web App, the user can subscribe patient and type of information. It by performing the following steps, shown in Table 13.

<table>
<thead>
<tr>
<th>STEP</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Select a patient from patient list</td>
</tr>
<tr>
<td>Step 2</td>
<td>Save subscription</td>
</tr>
<tr>
<td>Step 3</td>
<td>Subscribe a type of information from the list</td>
</tr>
<tr>
<td>Step 4</td>
<td>Save subscription</td>
</tr>
</tbody>
</table>

Before getting notification it should be subscribed from Web App. There are two type of lists which are the patient list and type of information list. User can subscribe from these two lists.

7.5.3. Activities in FHIR server

The FHIR server defines a set of common interactions (read, update, search, delete) performed on a repository of typed resources. It by performing the following steps, shown in Table 14.

<table>
<thead>
<tr>
<th>STEP</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Receive request from front-end</td>
</tr>
<tr>
<td>Step 2</td>
<td>Process request (read, delete, search, update)</td>
</tr>
<tr>
<td>Step 3</td>
<td>Send response</td>
</tr>
</tbody>
</table>

7.5.4. Activities in Mobile App

Mobile App checks a notification from the FHIR server via polling approach and shows a notification on mobile app. It by performing the following steps, shown in Table 15.

<table>
<thead>
<tr>
<th>STEP</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Polling notification</td>
</tr>
<tr>
<td>Step 2</td>
<td>Show notification</td>
</tr>
</tbody>
</table>

7.5.5. Activities in Message Generator

The message generator is a module which generates a content of notification. The message generator provides a notification message based on the updated/created patient data. It performs the following steps, shown in Table 16.

---

4 https://www.hl7.org/fhir/resourcelist.html
<table>
<thead>
<tr>
<th>STEP</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Get new created/updated resource in database</td>
</tr>
<tr>
<td>Step 2</td>
<td>Check the updated/created data is subscribed.</td>
</tr>
<tr>
<td>Step 3</td>
<td>If data is subscribed, generate a message based on created/updated data</td>
</tr>
<tr>
<td>Step 4</td>
<td>Save generated message</td>
</tr>
</tbody>
</table>
Figure 9 Activity diagram of CardioNotifier
7.6 Physical view

After the detailed description of core components and interaction flows, this section discusses how to deploy the entire system physically. Depending on technologies there are three ways to deploy the system.

In order to deploy the system, the FHIR requires a web server, a database server and an execution environment. The technologies applied for developing web servers and message generator the same.

![Deployment View of CardioNotifier](image)

**Figure 10 Deployment View of CardioNotifier**

7.7 The overall system architecture

Figure 11 shows the overall system architecture of CardioNotifier and a proposal solution how it can work together with a CVIS system using the FHIR standard. The grey blocks show already existing components and blue represents the part which has to be added but not in the scope of this project. This became out of scope as described in Section 5.2.1. The orange parts are added within the scope of this project. The Message Generator component is used to detect the information update in FHIR Database via polling and then generates a message to the database. The mobile app gets generated message through the FHIR Service via polling. The next section part of the CardioNotifier solution has been explained in details.

![Overall Architecture of CardioNotifier](image)

**Figure 11 Overall Architecture of CardioNotifier**
8. System Design

So far, the overall architecture choices are detailed, but not the design of each component itself. This section covers the detailed design of the components: Mobile App, and FHIR service and the design decisions. The other implementation details are detailed in next Chapter, Implementation.

8.1 Mobile App

The main function of the Mobile App is to show notification to the end-users. For that, the component receives a message from push notification service and sends a request to FHIR server to get a content of the notification. Before starting the implementation, there are two things to decide: which mobile platform to use and what application type to choose. The rationale behind the choices are detailed in Section 8.1.1 and 8.1.2.

8.1.1 Mobile application platform

There are many mobile platform options in the market such as Android, iOS, and Windows phone. Table 17 shows the comparison between two most popular platforms: Android and iOS.

Table 17 Comparison between Android and iOS

<table>
<thead>
<tr>
<th></th>
<th>Android</th>
<th>iOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development cost</td>
<td>Free</td>
<td>Tools are free but need publisher account</td>
</tr>
<tr>
<td>Publisher account needed for development</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Popularity</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Learning curve</td>
<td>Medium</td>
<td>Long</td>
</tr>
<tr>
<td>Device available</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Both Android and iOS are powerful and popular; however, Android will cost us less because it has an open source SDK and also we already have a device to test the result on. Also, the developer has previous experience and knowledge of Android application development; therefore, the learning curve will be shorter than iOS development. Furthermore, the iOS more closed system with many restrictions while Android is more open. Considering above reasons, Android was chosen as a mobile application platform in this project.

8.1.2 Mobile application type

There are following three types of mobile applications:

- **Native app** is a Smartphone application that works on particular platform or device. It is mostly coded on a specific programming language such as Objective C for iOS and Java for Android operating systems.
- **Hybrid app** uses a common code base to deploy native-like apps to a wide range of platforms.
- **Web app** is a regular mobile website behaves like native applications.

Table 14 shows the comparison between these three types.

<table>
<thead>
<tr>
<th></th>
<th>Native</th>
<th>Hybrid</th>
<th>Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed performance</td>
<td>Very fast</td>
<td>Fast</td>
<td>Fast</td>
</tr>
<tr>
<td>Reliability</td>
<td>High</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Device access</td>
<td>Full</td>
<td>Full</td>
<td>Partial</td>
</tr>
<tr>
<td>Offline work</td>
<td>Possible</td>
<td>Possible</td>
<td>Impossible</td>
</tr>
</tbody>
</table>
Able to give push notification using native controls | Yes | Yes | No
---|---|---|---
Portability | Low | Medium | High
Extendibility | Easy | Difficult | Easy

Considering the fact that native apps perform fast and it is easy to start implementation without deploying any additional services.

### 8.1.3. Logical View: Factory and Observer Design Patterns

Figure 12 shows the class diagram of Mobile App, which reflects both of the Factory and Observer design patterns. There are following main four classes:

- **Fragment** represents different parts of the GUI. Each of the fragment has its own CustomList. In another word, Fragment is a container of the CustomList. This class is an abstract class which has following four concrete classes which are observer of the ViewAdapterPage:
  - **SelectedPatientFragment** class is a fragment for the SelectedPatientList class which shows list of selected patients.
  - **AllPatientFragment** is a fragment of AllPatientList class which shows the list of all patients.
  - **NotificationFragment** is a fragment of NotificationList class which show all notification list.
  - **PatientDetailsFragment** is a fragment of PatientDetailsList class which shows the list of notification for one specific patient.

- **CustomList** is used to show the list of information on its fragment. This is an abstract class of following concrete classes. Each of these concrete classes belong to one of the Fragment concrete classes described above.
  - **SelectedPatientList** class shows only selected patient list on the fragment SelectedPatientFragment.
  - **AllPatientList** class shows list of all the patients on its fragment, AllPatientFragment.
  - **NotificationList** class shows all the list of notifications on NotificationFragment.
  - **PatientDetailsList** class is used to show the details of the selected patient on PatientDetailsList fragment.

- **ViewPagerAdapter**: This class performs as a Factory which creates the concrete fragment classes. Also, the class acts as an Observable by registering all the created concrete fragments and notify them when its status changes.

- **MainActivity**: The main class of the application which connects the application to other components. The class is used to receive the notification from the server and invokes updateState function of ViewAdapterPage to notify all the observers (fragments).
Figure 12 Class diagram of Mobile App

8.2 FHIR Server

8.2.1. Some available FHIR Servers

There are many test servers available for testing purpose. Table 18 shows comparison between some available FHIR testing servers.

Table 18 Publicly Available FHIR Servers for Testing

<table>
<thead>
<tr>
<th>Name</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPARK FHIR</td>
<td>Easy to deploy</td>
<td>No Authorization</td>
</tr>
<tr>
<td></td>
<td>All resources are available</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Open Source</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Easy integration for 3rd party</td>
<td></td>
</tr>
<tr>
<td>Oridashi FHIR</td>
<td>No deployment</td>
<td>Read-only</td>
</tr>
<tr>
<td></td>
<td>All resources are available</td>
<td>No documentation</td>
</tr>
<tr>
<td></td>
<td>Easy integration for 3rd party applications</td>
<td>No Authorization</td>
</tr>
<tr>
<td>NProgram</td>
<td>No deployment</td>
<td>Read-only</td>
</tr>
<tr>
<td></td>
<td>All resources are available</td>
<td>No documentation</td>
</tr>
<tr>
<td></td>
<td>Some resources are available</td>
<td>No Authorization</td>
</tr>
<tr>
<td>SMART Platforms</td>
<td>Authorization(OAuth)</td>
<td>Complex to deploy</td>
</tr>
<tr>
<td></td>
<td>Open Source</td>
<td>Lack of documentation</td>
</tr>
<tr>
<td></td>
<td>Easy integration for 3rd party applications</td>
<td>No client library for Android</td>
</tr>
<tr>
<td>HAPI/University Health Network</td>
<td>All resources are available</td>
<td>Complex to test</td>
</tr>
<tr>
<td></td>
<td>Open source</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Good documentation</td>
<td></td>
</tr>
</tbody>
</table>
The important criteria to choose the test server is future compatibility with the CVIS back-end service. In this project we assume that both the CVIS and CardioNotifier will adopt future releases of the FHIR standard. Developing the CardioNotifier application compatible with the FHIR standard will provide seamless integration with the CVIS (Figure 8). Considering this, we chose SPARK FHIR test server for this project. SPARK FHIR test server is an open source options for C# health communities which gives a quick start in FHIR implementations and integration.

### 8.2.2. Web App using FHIR subscription

FHIR subscription functionality has two steps:
- Step1: Make a subscription
- Step2: Send a push notification to subscriber

Step1: Make subscription
The subscription resource in FHIR is used to define a push based subscription from a FHIR server to mobile app. How the overall subscription is done is explained in Figure 13. First, the cardiologist makes subscription via the Web App which sends a post message in a JSON format. Then FHIR server handles the received message and update the Database.

The following is an example rule for server push criteria:

```
“criteria”: Observation?code=http://loinc.org|1975-2,
```
However, this functionality is not implemented yet in any of the available FHIR test servers described in Table 18. Therefore, we found following two possible solutions: Polling from Mobile App and Push notification service as a substitute for FHIR subscription functionality. The details of the both solution are covered in next sections.

8.3 Message Generator

The message generator module generates the content of a notification message. This module was demanded by the Usability requirement described in the Section 6.2.4.

The message generator checks new updates in the FHIR database. When the patient data is updated, it generates the content of a notification and saves back to the database.

To avoid that frustration and make the module more scalable, we decided to apply business rule engine in this module. By using the business rule engine, a system maintainer can easily define the message generation logic using the user friendly GUI. It is also easy to change or edit the already defined logics. The details of the business rule engine is described in the following section.

8.3.1. Business rule engine

The Business Rule Engine (BRE) is a software component that is used to validate business rules at runtime. They are often used in order to easily change business rules without software recompile/redeploy.

The advantage using rule engine is that it separates logic and data. Therefore, it gives an opportunity to gain improved speed, scalability, and flexibility, as well as reduced complexity, reusability. Table 19 shows comparison between available BREs.

<table>
<thead>
<tr>
<th>Rule Engines</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drools</td>
<td>Good documentation&lt;br&gt;Eclipse IDE plugin&lt;br&gt;Web UI for rule editor&lt;br&gt;Familiar for Philips Research&lt;br&gt;Supported by RedHat</td>
<td>Complex to deploy</td>
</tr>
<tr>
<td>OpenRules</td>
<td>Interface to create for a web application&lt;br&gt;Rules can be loaded by MS Excel or OpenOffice</td>
<td>Poor documentation</td>
</tr>
<tr>
<td>JRuleEngine</td>
<td>Interface to create for a web application&lt;br&gt;Rules can be loaded by an XML file</td>
<td>Poor documentation&lt;br&gt;Not updated recently</td>
</tr>
<tr>
<td>Zilonis</td>
<td>Open source</td>
<td>Poor documentation&lt;br&gt;Not updated recently</td>
</tr>
</tbody>
</table>

As a result, Drools rule engine was chosen for this project because it was suitable for creating a proof of concept with its advantages such as good documentation, Eclipse IDE plugin, familiar for Philips Research, and Supported by RedHat.
In Drools rule engine, there is a rule file with a “.drl” extension which contains multiple rules, queries, and functions. The following is the example how to define rules:

```
rule "name"
  when
    LHS
  then
    RHS
end
```

The “LHS” is the conditional parts of the rule. The “RHS” is a block that allows specific function to be executed.

### 8.3.2. Design of the Message Generator

Figure 14 shows the class diagram of the Message Generator. There are following main classes:

- Notification class contains generated message.
- MongoDBConnection class contains connection between database and message generator.
- Resource class contains the parsed value of the updated patient data.
- DroolsMain class executes loaded rules and updated resources in order to generate messages.
- Message Generating Rules is a “.drl” file which contains the logic of the rules.

![Class diagram of the Message Generator](image)

**Figure 14 Class diagram of the Message Generator**

### 8.4 Conclusion

Figure 15 shows the overall system design of the CardioNotifier and the proposal of the technical solutions. Also, interfaces are defined between the components based on FHIR standard. In the next section, the detailed information of the implementation is introduced.

![Overall System Design of the CardioNotifier](image)

**Figure 15 Overall System Design of the CardioNotifier**
9. Implementation

In this chapter covers the implementation details of the Web App component and Subscription part of the FHIR service component which was implemented little bit differently than the designed solutions. The rationale behind that change is also given. Also, details of the Data repository is explained at the end.

9.1 Mobile App GUI

The Mobile App GUI which notifies cardiologist with the update of his/her desired information.

Figure 16 Mobile app main GUI
Main window shows how many new notifications are arrived (Figure 16 (a)). When user press on the notification, it will direct the user to the main window of the CardioNotifier Mobile App. In tab “All”, the number of the new notification for each patient is shown (Figure 16 (b)). Prioritized tab shows the number of new notification for only a subscribed patients (Figure 16 (c)). Notifications tab shows the entire list of notifications for subscribed resources (Figure 16 (d)).

![Figure 17 Mobile app detailed GUI](image)

Figure 17 displays how to check notification history on mobile app. In tab “All”, by pressing on highlighted patient name it will move to the patient detailed window. This window shows patient detailed information, diagnostic status, and history of the notifications. After checking an unseen notification, the highlight will disappear from the “All” tab list.

### 9.2 Web App

The Web App was implemented to support the Mobile App demonstration. That means the proper designing of the Web App is not a main concern of this project. The Web App has two main functionalities: subscribe and update resources.

#### 9.2.1 Subscription

To make a subscription, user has to do followings (Figure 18):

- Choose practitioner
- Select patients (Multiple selection is possible). After pressing Subscribe button, the selected patients will be linked to the selected practitioner.
- Select information type (Multiple selection is possible). After pressing the Subscribe button, the selected information types will be linked to the selected patients and selected practitioner.
Pressing on the details button next to patient (Figure 18), user can make a notification for that patient by making create or update resource (See Section 8.2.2)

### 9.2.2. Create/update patient data

In the Create/update patient data window, user can make two things (Figure 19):
- Send any text message: The defined message will be sent to the practitioner as a notification
- Create notification based on information types. The available information types can be chosen from the Observation dropdown list. After pressing Create button, the defined notification will be sent to the practitioner who made a subscription for this patient and information type.
- Update existing data and send it as a notification. This can be done by pressing the details button.

---

**Figure 19 Create/update patient data window**
9.3 FHIR service

9.3.1. Custom notification service

There are two ways to check registered subscription which are push based check and poll based check. The FHIR server provides push based subscription that means server checks every resource when created or updated and if the resource matches with defined criteria it sends notification.

In this project it is not possible to use because of the following reasons:

The poll based subscription, there is another service that is polling frequently from created and updated resources check it with criteria.

9.4 Data repository

SPARK test server uses MongoDB database server. In this demo implementation of the project, the server contains following information:

Subscription: Figure 20 shows data structure of the Subscription resource.

![Subscription UML Diagram](https://www.hl7.org/fhir/subscription.html)

Notifications: Notification information is stored in MessageHeader resource object in the database. The MessageHeader resource is defined in order to support Messaging using FHIR resources. The principle usage of the MessageHeader resource is when messages are exchanged. Figure 21 shows the data structure of the MessageHeader resource.

![MessageHeader UML Diagram](https://www.hl7.org/fhir/messageheader.html)

---

5 https://www.hl7.org/fhir/subscription.html
6 https://www.hl7.org/fhir/messageheader.html
Observations: Observations are a central element in healthcare, used to support diagnosis, monitor progress, determine baselines and patterns and even capture demographic characteristics. Observation resource include:
- Vital signs: temperature, blood pressure, respiration rate
- Laboratory Data
- Imaging results like bone density or fetal measurements
- Devices Measurements such as EKG data or Pulse Oximetry data
- Clinical assessment tools such as APGAR
- Personal characteristics: height, weight, eye-color
- Social history: tobacco use, family supports, cognitive status
- Core characteristics: pregnancy status, death assertion

In this project demo, Blood pressure measurement, EKG, and Weight measurement are used. Figure 22 shows the structure of FHIR observation data.

Figure 22 Observation UML Diagram

https://www.hl7.org/fhir/observation.html
10. Verification & Validation

The architecture and design presented in the previous chapter should fulfil all the requirements introduced in Chapter 6. To ensure that, the verification and validation were applied to the CardioNotifier application. This chapter describes various tests and results for verification and concludes with the validation by answering the question “Is this a useful tool for cardiologist?”

10.1 Verification

Since the project goal is a proof of concept, the main effort was not spent on the complete testing strategy. However, to provide a stable prototype, the functional test was conducted manually through the CardioNotifier application.

10.1.1. Functional Test

The test covered following test cases:

1. Subscribe resource (Table 20): This test case can be done on Web App by choosing a patient and information type and send subscription request. As a result, the subscription data in a database has to be updated.

2.

Table 20 Subscribe resource

<table>
<thead>
<tr>
<th>TC1: Subscribe resource</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Objective</strong></td>
<td>This test case is used to check if user can make a subscription for desired resource</td>
</tr>
<tr>
<td><strong>Precondition</strong></td>
<td>Web app is up and running</td>
</tr>
<tr>
<td><strong>Step</strong></td>
<td><strong>Expected result</strong></td>
</tr>
<tr>
<td>1. Select “Jones Patricia” patient from patient list</td>
<td></td>
</tr>
<tr>
<td>2. Press subscribe button</td>
<td>Web App: Shown message box: Selected patient list has been updated Mobile App: shown “Jones Patricia” information in “Prioritized” tab</td>
</tr>
<tr>
<td>3. Select “Weight Measured” information type and press subscribe button</td>
<td>Web app: Shown message box: Selected type of information has been updated</td>
</tr>
</tbody>
</table>

Conclusion

Table 21 Update resource

<table>
<thead>
<tr>
<th>TC2: Update resource</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Objective</strong></td>
<td>This test case is used to check if user can make a subscription for desired resource</td>
</tr>
<tr>
<td><strong>Precondition</strong></td>
<td>Web app is up and running</td>
</tr>
<tr>
<td><strong>Step</strong></td>
<td><strong>Expected result</strong></td>
</tr>
<tr>
<td>1. Create new observation “Weight Measured” from dropdown list and press create button</td>
<td>Shown created observation in the list view</td>
</tr>
</tbody>
</table>

Conclusion

Table 22 Receive notification

<table>
<thead>
<tr>
<th>TC3: Receive notification</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Objective</strong></td>
<td>This test case is used to check if user can receive notification about</td>
</tr>
</tbody>
</table>
the subscribed information from server

<table>
<thead>
<tr>
<th>Precondition</th>
<th>Mobile application is up and running, TC1 and TC2 were executed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step</td>
<td>Expected result</td>
</tr>
<tr>
<td>1. Check the mobile</td>
<td>Main window is shown “1 new message(s)”</td>
</tr>
<tr>
<td>2. Press arrived notification</td>
<td>In tab All, 1 new notification for “Jones Patricia” is shown</td>
</tr>
<tr>
<td>3. Press “Prioritized tab”</td>
<td>1 new notification for “Jones Patricia” is shown</td>
</tr>
<tr>
<td>4. Press “Notification tab”</td>
<td>1 new notification with context for “Jones Patricia” is shown</td>
</tr>
<tr>
<td>5. Press on the “Jones Patricia”</td>
<td>Shown the notification details window</td>
</tr>
<tr>
<td>6. Press back button</td>
<td>The highlight is removed after checking the new notification. The bell icon will be also removed if there is not any new notification left</td>
</tr>
</tbody>
</table>

**Conclusion**

**Table 23 Test case for not receiving notification**

<table>
<thead>
<tr>
<th>Objective</th>
<th>This test case is used to check if user cannot receive a notification when the not subscribed message has been updated.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precondition</td>
<td>Mobile application is up and running.</td>
</tr>
<tr>
<td>Step</td>
<td>Expected result</td>
</tr>
<tr>
<td>1. Uncheck “Jones Patricia” patient from patient list and press the subscribe button</td>
<td>Web App: Shown message box: Selected patient list has been updated</td>
</tr>
<tr>
<td>2. Create new observation for “Jones Patricia”</td>
<td>Web App: Shown message box: Patient data has been updated</td>
</tr>
<tr>
<td>3. Check the mobile</td>
<td>Main window is shown no notification from CardioNotifier</td>
</tr>
<tr>
<td>4. Select “Jones Patricia” patient from patient list and press subscribe button</td>
<td>Web App: Shown message box: Selected patient list has been updated</td>
</tr>
<tr>
<td>5. Unselect observation “Weight Measured” from dropdown list and press subscribe button</td>
<td>Web app: Shown message box: Selected type of information has been updated</td>
</tr>
<tr>
<td>6. Create new observation “Weight Measured” from dropdown list and press create button</td>
<td>Shown created observation in the list view</td>
</tr>
<tr>
<td>7. Check the mobile</td>
<td>Main window is shown no notification from CardioNotifier</td>
</tr>
</tbody>
</table>

**Conclusion**

**10.1.2. Non-functional Requirements Evaluation**

Non-functional requirements are important to check software quality. Table 24 shows the relation between non-functional requirements and the corresponding Architectural and design decision that helped to achieve that requirement. The details of the design decision can be found in specified sections.

**Table 24 Non-functional requirement verification**

<table>
<thead>
<tr>
<th>Non-functional Requirements</th>
<th>Design decisions which support the requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>Client-Server architectural pattern (Section 7.2.2)</td>
</tr>
</tbody>
</table>
### 10.2 Validation

#### 10.2.1. Validation with Cardiologist

Table 25 shows the survey created for validation purposes of CardioNotifier. The questions address the user interface (cardiologist) and functionality. The users can answer for each question with an answer scaling from flowing numbers [20]:

- **1 (bad)** – Feature is not easy to use or don’t understand it / User interface is bad;
- **2 (not good)** – Feature takes time to learn, can be made more intuitive / User interface is a bit difficult to use
- **3 (good)** – Feature is fairly easy to use / User interface is easy enough to use
- **4 (very good)** – Feature is very easy to use / User interface is easy to use
- **5 (excellent)** – Feature exceeds expectations / User interface is very easy and intuitive

*Table 25 Survey template for validation*

<table>
<thead>
<tr>
<th>Question (mark the answer with an X)</th>
<th>1 (bad)</th>
<th>2 (not good)</th>
<th>3 (good)</th>
<th>4 (very good)</th>
<th>5 (excellent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>How easy it is to use this?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Does the application performs as is should?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How clear are the results displayed?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Did the results provide you with meaningful information?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How helpful is the application in your daily work?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How did you like the overall design of the application?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How easy it is to learn the application?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Due to inability of a cardiologist, Geert Gijsbers helped to make a validation for CardioNotifier. He is principal scientist in In-Body system department, Philips Research and he has good experience in this field. The following section explains validation of the CardioNotifier.

#### 10.2.2. Feedback

The selected patient and selected type for notification need to be very easy to use. Cardiac patients do not often stay at the hospital to get treatment for a long period (maximum 3 days). Therefore, the list of the patients that the cardiologist manages after every day is needed to pay attention. Also, it is important to find a way how to present a patient list selection more efficiently.
Currently, CardioNotifier receives only notifications. The real value is in providing the update info or summary. The current CardioNotifier is less valuable because a cardiologist still needs to find a place to login in to the system and check updated information. However, if the cardiologist receives some related information with the notification such as summary or comment based on updated information, it would be more valuable.

Also, it would be nice if the cardiologist can see patient summary from the CardioNotifier, e.g., if the cardiologist needs to stand in for a colleague and encounters a patient he/she has never seen, it would be nice to be able to check quickly latest summary of patient condition, diagnosis, results.

The showing diagnostic status in notification details window was shown clear. However, it would be nice to show summary of diagnosis or real information when user press the diagnostic status.

Table 26 shows a result of validation with the end-user. This is current prototype is only showing notification about updated information such as “XXX patient’s blood pressure has been updated”.

Table 26 Survey to validate the CardioNotifier

<table>
<thead>
<tr>
<th>Question (mark the answer with an X)</th>
<th>1 (bad)</th>
<th>2 (not good)</th>
<th>3 (good)</th>
<th>4 (very good)</th>
<th>5 (excellent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>How easy it is to use this?</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Does the application performs as is should?</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How clear are the results displayed?</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Did the results provide you with meaningful information?</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How helpful is the application in your daily work?</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How did you like the overall design of the application?</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>How easy it is to learn the application?</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Table 27 shows a result of validation with end-user assuming an improve prototype which is only showing notification about updated information such as “XXX patient’s blood pressure has been updated - 107mm[Hg], 60mm[Hg]”.

Table 27 Survey to validate the CardioNotifier (notification with summary)

<table>
<thead>
<tr>
<th>Question (mark the answer with an X)</th>
<th>1 (bad)</th>
<th>2 (not good)</th>
<th>3 (good)</th>
<th>4 (very good)</th>
<th>5 (excellent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>How easy it is to use this?</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Does the application performs as is should?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How clear are the results displayed?</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Did the results provide you with meaningful information?</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
formation?

| How helpful is the application in your daily work? |   |   | X |
| How did you like the overall design of the application? |   |   | X |
| How easy it is to learn the application? |   |   | X |

10.2.3. Conclusion

Currently, receiving only notification is less valuable because cardiologist still needs to login in to the system for checking updated information. Therefore, a context of notification should be provided more valuable information for cardiologist such as summary of updated patient information. It is important to know about patient updated without login into the system.

In the project, a number of technological and project risks were identified as mentioned in section 5.2. It caused small changes in the direction of development for the product. These changes were clearly presented to the stakeholders and proposed mitigation strategy from the candidate. Some of these risks are mentioned as future work for the product.
11. Conclusions

In this chapter, the results obtained in the project are covered. Following this, the suggestions for future work, which will be needed to improve the product and make it more complete, are listed.

11.1 Results

Within the scope of CardioNotifier project, smartphone-based notification system for the cardiologist was developed and demonstrated. For this, a research about cardiologist’s workflow was conducted to find out how smartphone can support cardiologist’s daily work. Also, various technical investigations took place to achieve most optimum design solutions. The following are the achievement in this project:

- Research about how smartphone can support cardiologist’s workflow (Chapter 3).
- Problem and solution analysis
- Architecture and design
  - Available architecture Client-Server architectural pattern (Section 7.2.2)
  - Reliable architecture with help of Push notification Service (Section 8.5)
  - Fast and efficient architecture with help of the design decisions native application (Section 8.1.2) and polling (Section 7.3.2 and 7.3.3)
  - User friendly GUI (Section 9.1)
- Prototype demonstration for cardiologist which helped to further analyze the problem and wish of the end user.

11.2 Future works

The following is the list of the suggestions for future improvements:

- Mobile App
  - The Mobile App should show also details of the information received as a notification because of the reason summarized in Section 10.2
- Web App
  - Web App was developed to support the demonstration of the Mobile App in this project. Implementing it as a part of the CVIS system, specifically ISCV, would be desirable.
- Application service
  - Message generator component can be improved to provide more detailed information for the cardiologist.
- Security and privacy
  - Although security and privacy was not covered in the design and implementation of this project, it should be a main concern of the future work.
- SMART on FHIR
  - SMART on FHIR can be used when the client library for Android platform become available
- FHIR subscription
  - FHIR subscription can replace current implementation of message notification when it becomes functional in the future.
12. Project Management

In this chapter focuses on how the project was carried out, from a project management point of view. The focus is on the methods used to steer the planning and monitor the progress, which includes a description of work breakdown structure and project planning.

12.1 Introduction

The project was conducted for nine months and consisted of two main parts. The first part of the project focused on cardiology domain study, problem analysis, and user requirements. The second part of the project considered design and implementation based on the outcomes of the first part.

In the early stage of the project we planned to show a demo every meeting with clinical expert to validate. Therefore, the management of the project was conducted using elements of the agile methodology, Scrum approach. The agile scrum supports iterative and incremental approach to software development which called iterations. It uses the term Sprint for iterations which is short time span of usually two weeks. Priorities are regularly updated in which deliverable with clear value to user need to be produced.

The following section explains into the details of the project management such as project planning, features, and acceptance criteria.

12.2 Work-Breakdown Structure (WBS)

There are two top level packages are identified early stage of the project which cardiology domain analysis and design and implementation. Each package is further decomposed to smaller packages. The work-breakdown structure for project is presented in Figure 23.

![Figure 23 The work-breakdown structure](image)

The CardioNotifier domain analysis part consists of three packages:

- Domain study – research and study about cardiology and cardiology workflow in a hospital.
- CVIS – study for integration with mobile app
- FHIR – research and study the standard of the FHIR for sharing data.
- SMART on FHIR - research and study the SMART on FHIR platform for the integration with mobile apps.

The design and implementation part consist two main parts which are Deploy back-end system and Application. It includes the following sub parts:
- FHIR server – deploy and setting up the FHIR server and database for testing.
- Mobile app – design and implement prototype of mobile app.
- Web app – design and implement prototype of web based application.

12.3 Project planning

Based on the breakdown structure defined in the previous section, a project plan was formulated accordingly (Figure 24). The beginning of the project it was not suitable to follow the agile approach because it was required to gain domain knowledge first. Therefore, the study of domain was reserved for the first two months.

12.3.1. Initial

After the domain study, a prototype vision was created and discussed with stakeholders and split into features considering the priorities. The following is initial backlog which was defined in beginning of the project. Figure 25 shows the agile product backlog which is sorted by priority (F1 - higher priority, F7- lower priority) and each feature has acceptance criteria.

![Figure 24 The Project Planning](image-url)

![Figure 25 The initial agile product backlog in Scrum](image-url)
Every week the project manager has been updated with project progress and an achieved result was validated during the PSGM (Project Steering Group Meeting).

12.3.2. Final
Two months after the project had been started, the following changes were made which affected the project planning.

- Integration between the Mobile app and ISCV
- Study for new back-end deployment (SMART on FHIR, FHIR testing servers)
- The project manager has been changed during the project
- Clinical expert availability
- Addition of new features about security and privacy

Due to the time frame, it was probably not bearable to achieve all features that were defined in the backlog. Therefore, each feature is defined by amount of work to do (e.g., S – small, M - medium, L - large) in order to choose priorities. Figure 26 shows the final agile backlog.

12.4 Conclusion
In this project, it was not suitable to follow agile scrum approach completely because of the clinical expert’s availability and the lack of a larger Development Team. Therefore, we used some technics of agile scrum approach such as creating vision of the prototype, features, and acceptance criteria and reprioritizing the features regularly.

Also, the project of progress was presented by candidate every month in PSG meeting and the stakeholders (except clinical expert) gave feedback and made validation for prototype.

In the project, a number of technological and project risks were identified as mentioned in section 5.2. It caused small changes in the direction of development for the product. These changes were clearly presented to the stakeholders and proposed mitigation strategy from the candidate. Some of these risks are mentioned as future work for the product.
13. Project Retrospective

In the previous chapter the project was evaluated from a project management point of view. In this chapter the project is evaluated again, but this time as a personal reflection. What went as expected, what did not, and what are the lessons learned? Moreover, the design opportunities from Section 4.2 are revisited.

13.1 Reflection

When I look back at the CardioNotifier project after working on it for nine months, the overall project looks successful.

This project gave me the full experience of Software Engineering in real practice. I could apply the theoretical knowledge gained during the OOTI education in real life, which made my knowledge more concrete.

During the project, I could make use of my good communication skill when to communicate with both end-users and software team members.

Contact with the end-user in this project was very limited. However, I did not miss any chance to speak with clinical experts at Philips, who were not involved in this project directly, that helped me a lot to analyze the problem and customer needs.

With the people with Software background, I could communicate very efficiently. I did not lose any single opportunity to communicate with them to discuss the problem and clarify technical possibilities. Frequent contact with team members at Philips in both Netherlands and China and also, technical support team of FHIR and SMART on FHIR in US helped me to solve many difficult technical issues.

The most challenging part in the project was dealing with stakeholders who have non-Software background. I was constantly exploring the easy to understand way to achieve efficient communication.

Also, one of another challenge was working individually rather than working as a team which was totally new for me. The project is a one-person project; therefore, I needed to perform as a project owner, researcher, organizer, and implementer all at once. However, I was lacking some of the skills such as critical thinking and self-organizing which I never needed before because I was always a part of a team where the things were already planned and organized by someone else. Therefore, to successfully control the project process, I needed to be critical on complicated subjects and I needed to be self-organized to take all the controls in hand.

As a result, I leaned following lessons that helped me to grow a lot as a Software Architect and Designer

13.1.1. Lessons Learned

Lessons learned from technical point of view:

- Architectural and design patterns: In this project we applied several architectural and design patterns which helped us to solve certain problems. Finding that patterns was not that easy, requiring several weeks of reading. As a result, now I have a better understanding of Architectural Patterns and their relations with the quality attributes. I am motivated to learn more about
Software Architecture and Design and am looking forward using them in the future projects.

- Also, with a help of the project, I could earn good knowledge about FHIR and SMART on FHIR, the difference between them, and when to use them.
- The project helped me to understand the domain Cardiology which I found extremely interesting
- The project gave me an opportunity to conduct a research in different domain which I have to background knowledge. With this opportunity I could understand what the difference between research and development projects are.

Lessons learned from organizational point of view:

- With help of the project, I could make use of Agile and scrum project development method in real situation.
- The project gave me insight about the importance of critical thinking and self-organizational skills. With help of the project, now I have better understanding about where I make mistakes and how I should tackle them. To analyze the problem better I have to come up with many questions and deep dive to answer all of them. To control the situation better, I need to take all the initiative and make an initial proposal of solution by myself first and then it becomes easier to get feedbacks from others. I am planning to concentrate on further the improvement of these skills during future projects by using the lessons I learned during this project.

### 13.2 Design Opportunities revisited

In Section 3.3, assessment criteria for a technical design that are relevant for this project were identified. Below, for each of these criteria, it is verified whether the design adheres to that criterion.

- The artifact satisfies the requirements described in Section 6 that are validated and verified in Section 10.
- The system is easy to use because the new GUI designed and implemented prototype (Section 9.1) qualifies all usability requirements. Also, the system is easily adaptable and configurable with help of the design and technology choices (Section 8.1, 8.2, and 8.3).
- The business rule engine was used in message generator (Section 8.3.1). It separates logic and data which improve its flexibility and maintainability.
- The interface definitions followed FHIR standard (Section 9.4) which makes it able to connect easily with any other FHIR standardized modules. Therefore, the architecture and design is concluded as generic.
- The result of the project proves that the concept CardioNotifier works with the real demonstrator. Therefore, the architecture and design is convincing (Section 9.2) and also, it is technically realizable.
## Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANSI</td>
<td>American National Standards Institute</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>APNs</td>
<td>Apple Push Notification service</td>
</tr>
<tr>
<td>Client</td>
<td>Operations that are performed by the client in a client–server relationship in a computer network.</td>
</tr>
<tr>
<td>CVIS</td>
<td>Cardiovascular Information Management System provides a cardiologist access to all relevant patient data.</td>
</tr>
<tr>
<td>Database server</td>
<td>Same as server, used to host a database</td>
</tr>
<tr>
<td>EDA</td>
<td>Event-Driven Architecture</td>
</tr>
<tr>
<td>EHR</td>
<td>An Electronic Health Record is an electronic version of a patient’s medical history that is maintained by the provider over time, and may include all of the key administrative clinical data relevant to that person's care under a particular provider.</td>
</tr>
<tr>
<td>FCM</td>
<td>Firebase Cloud Messaging</td>
</tr>
<tr>
<td>FHIR</td>
<td>Fast Healthcare Interoperability Resources is the latest standard to be developed under the HL7 organization.</td>
</tr>
<tr>
<td>GCM</td>
<td>Google Cloud Messaging</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HL7</td>
<td>Health Level Seven</td>
</tr>
<tr>
<td>HTTP</td>
<td>HyperText Transfer Protocol</td>
</tr>
<tr>
<td>ISCV</td>
<td>IntelliSpace Cardiovascular is a web based cardiology software solution which provides access to cardiovascular clinical information</td>
</tr>
<tr>
<td>JDBC</td>
<td>Java database connectivity technology that defines how a client may access a database</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>MPNS</td>
<td>Windows Push Notification Service</td>
</tr>
<tr>
<td>OOTI</td>
<td>Ontwerpersopleiding Technische Informatica</td>
</tr>
<tr>
<td>PEng</td>
<td>Professional Doctorate in Engineering</td>
</tr>
<tr>
<td>REST</td>
<td>Representational State Transfer</td>
</tr>
<tr>
<td>RESTful APIs</td>
<td>Interfaces that adhere to the REST style</td>
</tr>
<tr>
<td>Server</td>
<td>Operations that are performed by the server in a client–server relationship in a computer network.</td>
</tr>
<tr>
<td>SMART on FHIR</td>
<td>SMART on FHIR is a platform to integrate apps with Electronic Health Records, portals, Health Information</td>
</tr>
<tr>
<td>SOA</td>
<td>Service-Oriented Architecture</td>
</tr>
<tr>
<td>ST</td>
<td>Software Technology</td>
</tr>
<tr>
<td>TU/e</td>
<td>Eindhoven University of Technology</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>Web server</td>
<td>Same as server, available over the Internet</td>
</tr>
<tr>
<td>WNS</td>
<td>Windows Notification Services</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
</tbody>
</table>
Bibliography


About the Author

Tamir Tsedenjav received his BSc degree in Computer Science from National University of Mongolia in 2008. During his study period, he participated in a number of software contests successfully and received a job offer from one of the Japanese software companies named Udom Co., Ltd. He worked there as a Software Engineer between 2008 and 2013. During the working period, he worked on a number of joined projects with Hitachi, Ltd., and was involved in the development of Automated Warehouse and ERP systems as a developer and tester. He is interested in Software Architecture and Design.
4TU.School for Technological Design, Stan Akkermans Institute offers two-year postgraduate technological designer programmes. This institute is a joint initiative of the four technological universities of the Netherlands: Delft University of Technology, Eindhoven University of Technology, University of Twente and Wageningen University. For more information please visit: www.4tu.nl/sai