UNIVERSITY OF GOTHENBURG

ASSA ABLOY

Support of Embedded Hardware Equipment Facilitated by a Smartphone Application

Master of Science Thesis in the Programme Computer Science

ANNE-KATRIN KROLOVITSCH
LINDA NILSSON

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
Göteborg, Sweden, August 2011

Support of Embedded Hardware Equipment Facilitated by a Smartphone Application

ANNE-KATRIN KROLOVITSCH
LINDA NILSSON

Examiner: GRAHAM KEMP

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden August 2011

Abstract

In today's situation more and more tasks are carried out with the help of smartphones. The chosen research area is the improvement of support for advanced embedded equipment through the use of a smartphone application. This report addresses the issue of supporting a door lock system called Hi-O (Highly Intelligent Opening), developed by ASSA ABLOY. It is significantly harder to support an advanced door lock system than it is to support a traditional, mechanical system. The installation and service of the Hi-O system can be made easier by being able to directly access the information in the Hi-O system and change its configuration. The research question answered in this report is "how can a smartphone application facilitate support of an intelligent door lock system, focusing on the design of a software architecture?". The practical contribution of this research is a prototype for the smartphone application, called Aioli. The theoretical contribution is the architecture along with many empirical findings that are relevant for years to come. This smartphone application will make a considerable improvement compared to how installation and service of the Hi-O door lock system are carried out today. Through defining requirements, designing the architecture and developing a prototype the foundation for the support system Aioli was laid. The Design Research method that was used allowed several architectural evaluations to be carried out using the "architectural Software Quality Assurance" method. Evaluation of the architecture was conducted in collaboration with engineers at ASSA ABLOY. The Aioli prototype was developed for the Android platform and implements a selected part of the requirements specification. Findings show that the specified architecture design is feasible for the support of the Hi-O door lock system. From the evaluations of the prototype it became clear that a product like Aioli is desired.

Contents

1 Introduction

Smartphones [26] have become an asset in today's life that offer many possibilities for completing daily tasks. Individual users can install, configure and run any available application they like. Smartphones offer a lot of different functionalities and the possibility to connect to different kinds of hardware through a variation of communication protocols. They are powerful, easy to use, portable and widely distributed. These attribute have the effect that smartphones are increasingly used in many other settings than just personal usage. An example of this can be seen at Volvo Trucks Center in Gothenburg where a smartphone application has been developed to help the technicians in their daily work [56]. This application contains the whole handbook for the service of trucks, which has the advantage that the technician does not have to leave his working station to go and look something up on a stationary computer. Instead all information is available on his smartphone where the manual can easily be updated and maintained. As mentioned, the communication between different kinds of hardware and smartphones has also become of interest. This can be seen for example in the invention of Mobile Keys which allows clients to open, for example, their hotel room with their mobile phone [11]. The spreading of smartphone usage in different areas makes it attractive for companies to evolve their products by including smartphone compatible communication in their technology.

One area that can be improved with the help of smartphones is the support systems for advanced embedded equipment. Being able to retrieve the right information in a fast manner can improve the support time of products and aid the technicians in their work. This interest has been expressed by ASSA ABLOY who "is the global leader in door opening solutions" [10]. This project was conducted at the consultant firm Diadrom [25] for ASSA ABLOY. In this research, the advanced embedded equipment is the intelligent door lock system called Hi-O [12], which was developed by ASSA ABLOY. Hi-O is an abbreviation for "Highly Intelligent Opening". This technology consists of several door components, each one with an embedded system which communicate with each other over a Controller Area Network (CAN) [22]. Two important benefits gained from this technology is installation through Plug & Play [53] and the fact that it allows for communication with other software systems. The Hi-O system can be managed by a desktop application called Hi-O Manager, which can connect to the Hi-O door lock system and for example retrieve system information. However, this tool is not commonly used among service technicians today. One of the reasons is that the Hi-O Manager is a PC based application and requires the service technician to bring a laptop to every installation and service of the Hi-O system. This however, is not feasible.

Problems with the Hi-O door lock system that were expressed by our informants concern the installation and service process, because there is no clear way to see if an installation or service has been successful and therefore the process includes trial and error. Hence, an efficient and practical tool is desired to aid in this process. Furthermore, problems have been expressed concerning the work process when a Hi-O door lock system has broken down. When a door lock system breakdown is reported by for example a superintendent the service company does not always receive the information needed in order to make an informed decision and therefore needs to send a service technician to inspect the door. Even if the technician can solve the problem, he might not have the right spare parts. In case he cannot solve the problem, for example if it is a mechanical problem, a locksmith needs to be called instead. This shows that the installation and service of door lock systems involve many parties. If the procedure is inefficient it may become expensive for all parties involved. Also, as the distribution of the Hi-O system increases, efficient support becomes even more important.

The purpose of this research was to investigate the possibilities of a smartphone support application for the Hi-O door lock system. The scope of the project was to design a software architecture and develop a prototype as a proof-of-concept. The requirements that were the basis for these artifacts were collected through different qualitative data collection methods. As a research method, Design Research [54] [46] was applied which allowed for an iterative development and evaluation.

The research question addressed in this report is:

> ❝ How can a smartphone application facilitate support of an intelligent door lock system, focusing on the design of a software architecture? ❞

The results of this research project are the architecture and prototype for the support system called Aioli. The outcome consists of a theoretical contribution in the form of Aioli's layered software architecture which supports the communication with external systems and a practical contribution in the form of the Aioli prototype application. The prototype was developed for the Android platform [1] and it implements parts of the architecture. The communication between Aioli and Hi-O is achieved through Bluetooth.

The rest of the report is structured in the following way; Chapter 2 outlines the chosen research approach and explains how the data for this research was collected. Chapter 3 deals with the technical background for Hi-O and the Aioli prototype. In chapter 4 the outcome and result for Aioli are presented. Chapter 5 discusses these results, the future work and related work. Finally, chapter 6 presents the conclusions of this research.

2 Research Methodology

This chapter deals with the process and methods which have been applied in this research project. First the applied research approach is outlined, then the process of data collection is described.

2.1 Design Research

This research project was carried out with a qualitative research approach [23] using the Design Research method [54]. This method was chosen since it allows for a detailed exploration of a new area of interest and it "deals with creating something new that does not exist in nature" [54]. Furthermore, it is an iterative research method that gives the opportunity to gain knowledge in each iteration which is then used to improve the project artifacts. The artifacts created in this research project were a requirements specification, an architectural description, and a functional prototype.

The process of the Design Research method is depicted in Figure 1, it is the general methodology of design research described by Vaishnavi and Kuechler [54]. The process consists of five steps, which are not to be seen as sequential since a later step can give insight to a previous step. Thus, there is no strict order that needs to be followed within an iteration.

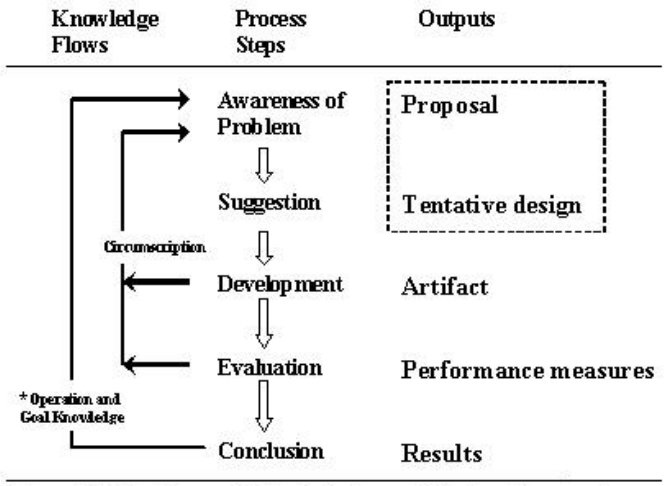


Figure 1: Design Research Process (From [54])

Awareness of Problem
Every iteration starts with the identification of the problems that shall be solved. This raises awareness among all researchers about current problems and the status of the project.

Suggestion
In this step solutions to the problems are discussed from the "existing knowledge base for the problem area" [54]. Thus, a plan is made for how the problems shall be solved during the current iteration.

Development
At this stage, the suggested solutions are implemented in the project artifacts. As Figure 1 shows, it is possible to go back to the "Awareness of Problem" step. This is called the "Circumscription process" which "assumes that every fragment of knowledge is valid only in certain situations" [54]. If it is decided to go back to the "Awareness of problem" step it is due to an incomplete knowledge base and one will start over with the acquired knowledge.

Evaluation
In this step the artifacts created in the “Development” step are evaluated. This is done with methods decided upon during the “Suggestion” step.

Conclusion
The final step of the Design Research method has the purpose to reflect upon the found results, and the processes used. It is also recommended to determine if the findings are generalizable and can be applied in similar situations. As shown in Figure 1 the first four steps are iteratively executed, but once the “Conclusion” step is reached the end of the research project is marked and a new Design Research project can be started.

The type of outcome of every step is dependent on the research project. For this research project the main outcome of the “Development” step were the requirements specification, the architecture description, and the functional prototype which implements parts of the architecture. Three iterations of the Design Research were performed, each of which had a different focus point (see section 4.1).

2.2 Data Collection Approach

Since this research was conducted using a qualitative approach it was feasible to use different forms of data collection. Table 1 shows the data collection activities which were carried out with different participants.

The data collection follows the characteristics for qualitative research described by Creswell [24]. For example, we used several data collection methods including interviews, an observation, and architectural evaluation workshops. This correspond to the characteristic stating “Qualitative research uses multiple methods that are interactive and humanistic”. Creswell also expresses that “Qualitative research takes place in the natural setting” and as can be seen in the table most activities were carried out in person. All the activities listed as In Person were also on site, meaning that we visited the participants at their work place to carry out the data collection activity. Furthermore, we used different mediums for carrying out the the data collection, since the participants were at different locations we were not able to meet all of them in person. This is described by Creswell in the characteristic stating“Qualitative research is emergent rather than tightly prefigured” with the example that the data collection procedure might change.

The first four interviews in Table 1 were conducted in order to get an understanding of the problem domain and to collect requirements for the Aioli application. The selected format for the interviews were semi-structured. Robson [44] describes them as flexible and allowing for predetermined questions where the interviewer can give explanations, change wording, omit or add questions if needed. Hence, semi-structured interviews were chosen, because they offer a lot of freedom during the process of conducting interviews. The interviews were structured so that one of us acted mainly as the interviewer and the other one took notes with pen and paper. However, both could ask questions and take notes if clarification was needed.

We had the opportunity to observe an installation of the Hi-O door lock system. This installation was carried out by a service technician from Sesam Lås & Larm [48] and took a full day. This observation was conducted in order to receive a more detailed picture of what an installation requires and how the Aioli application could help in the process. The service technician had previously done several installations and services on Hi-O door lock systems. To record the installation we took notes of the activities carried out by the service technician, noted the reaction to problems that occurred and how these were solved. During the observation we asked questions about what was happening in order to be able to follow the process. We also asked questions relating to the problems encountered. At the end of the observation we conducted a short semi-structured interview with the service technician in order to collect data on the end user’s perspective and on his opinions regarding the Aioli application.

Table 1: Data Collection Overview

No. of participants	Data Collection Type	Company	Position	Medium	Data Recording
1	Semi-Structured Interview	ASSA ABLOY	Product Owner	Skype Video Call	Sound Recording and Notes
1	Semi-Structured Interview	ASSA ABLOY	Support and Service Responsible	Speaker phone	Notes
1	Semi-Structured Interview	ASSA ABLOY	Second Line Support	In person	Notes
1	Semi-Structured Interview	ASSA ABLOY	R&D Manager	In person	Notes
1	Semi-Structured Interview	Sesam Lås & Larm	Service Technician	In person	Notes
4	Semi-Structured Group Interview	ASSA ABLOY	3 Software Engineers and 1 R&D Manager	In person	Sound recording
1	Observation	Sesam Lås & Larm	Service Technician	In person	Notes
4	Architectural Evaluation Workshop	ASSA ABLOY	3 Software Engineers and 1 R&D Manager	In person	Sound recording
2	Prototype Evaluation	Diadrom	2 Software Developers	In person	Notes
1	Prototype Evaluation	Sesam Lås & Larm	Service Technician	In person	Notes
1	Brainstorming	Diadrom	—	In Person	Notes
4	Initial Meeting	Diadrom and ASSA ABLOY	1 R&D Manager and 1 Software Engineer from ASSA ABLOY 2 participants from Diadrom	In Person	Notes
2	Additional Information Collection	ASSA ABLOY	1 R&D Manager and 1 Software Engineer	Email	Email

In total, three architecture evaluation workshops were conducted (see section 4.6). The first and third workshop were carried out by us and the second workshop (listed in Table 1) was conducted with participants from ASSA ABLOY. These workshops were conducted in order to gather feedback about the architecture and to get a quantitative measure of its quality. The second workshop was ended with a group interview which had the purpose of getting information on the participants' thoughts of the architecture as well as their opinion on the workshop.

For the evaluation of the Aioli prototype first two software developers at Diadrom were asked to test the prototype and give feedback on the everything they could think of. At a later stage we met with another service technician from Sesam Lås & Larm who we asked to evaluate the prototype as well.

During the brainstorming session and the initial meeting the research topic and possible scenarios for Aioli were discussed. During the initial meeting we were also provided with the Hi-O hardware used for the prototype. Moreover, close contact with the contact persons at ASSA ABLOY made it possible to have frequent email exchange to collect additional data. These were usually of technical nature such as the implementation of Hi-O or regarding the problem domain.

3 Background

This chapter describes the technical background for this research project. It explains the details of the Hi-O door lock system and the development environment used for the Aioli prototype application.

3.1 Hi-O

One or two Hi-O door lock systems can be connected to a gateway. Each door lock system consists of the gateway and one or more other components. There are many types of components such as door opening buttons, security locks and door automatics. Each component contains an embedded system and they are all connected in a CAN network. The components and the gateway can be arranged in a bus, mesh or star topology [15]. In a CAN network it is important to terminate at least one of the components in the network, this is known as bus termination. It is used to minimize signal reflection on the bus [36]. If the bus termination is not activated on at least one component, communication on the network is not possible.

The Hi-O components can run in three different operation modes. The first mode is Installation where the communication is not encrypted, password is not set and the components are in Plug & Play [53], which allows them to detect each other automatically. The second mode is Configuration where the encryption is switched off, the Plug & Play is off and no password is set. The third mode is Running, in this mode the system is encrypted, the password is set and Plug & Play is off. The Running mode is used when the door lock system is installed and taken into use. For the message encryption XTEA is used with a 128 bit encryption key, also message authentication with rolling codes is used [13]. In the door lock system the components make up a decentralized system where all component are aware of each other.

The Hi-O door lock system can either be standalone or up to 16 Hi-O systems can be connected to a sub-central. Hi-O can also be connected to an Electronic Access Control (EAC) system [14]. The EAC can log some data and change parameters in the Hi-O doors belonging to it. However, the EAC is an optional feature and it may not be desired to connect the door lock system to it in all installations.

3.2 Aioli Prototype Development Environment

The development environment for the prototype was set up as seen in Figure 2. The environment consisted of three Hi-O components (a push button, a motor lock and an electronic strike), a development board, a Bluetooth adapter and a smartphone running Android. The Hi-O components communicate using CAN, they are connected with each other and to the development board. The board was developed by ASSA ABLOY prior to this research project and is used for development of the Hi-O system. The board has a RS232 [17] interface for serial communication which provides the possibility to send serial data with ASCII encoding to the board which it then translates to CAN messages and sends them to the Hi-O components.

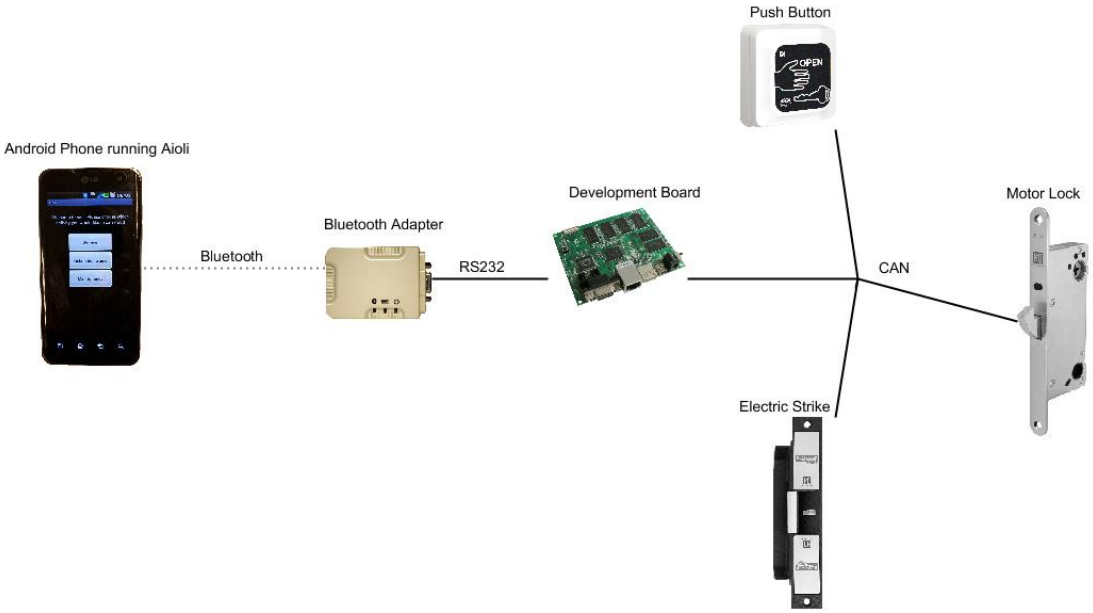


Figure 2: Development Environment for the Aioli Prototype

For the Bluetooth connection between Aioli and Hi-O the Bluetooth adapter bluemax05 [27], which supports RS232, was used. The adapter was set as master which allowed it to be detected by the smartphone.

The development platform used for the Aioli prototype was Android, which is based on the Java programming language. The platform version that was used was Android 2.2 [4]. The phones which were used for the development was a LG Optimus 2X [34] which ran Android 2.2 and a Samsung Galaxy SII [47] which ran Android 2.3.3[5]. Since Android is forward compatible, it is possible to run an application built for Android 2.2 on a smartphone that runs a later version.

4 Result

This chapter presents all results of this research. In section 4.1 an outline of the iterations is given followed by the data analysis results in section 4.2. After that the domain research results are explained in section 4.3 moving on to the presentation of the functional requirements in section 4.4.1, which also are acquired from the data analysis. This is followed by the design and evaluation of the Aioli architecture in section 4.5 and section 4.6. Finally, in section 4.7 the Aioli prototype is presented and it is evaluated in section 4.8.

4.1 Iterations

The Design Research steps outlined in section 2.1 were iterated over three times, each iteration had a different focus. This means that the part in focus was given more time and attention in that particular iteration. However, all research artifacts were extended and improved in all iterations.

4.1.1 Iteration 1

In the first iteration we focused on the data collection and formulating the research problem. As the first part of the data collection process a brainstorming session with our contact person at Diadrom was conducted as well as an initial meeting with our contact persons at ASSA ABLOY was held, in order to discuss the research topic. After that the first four interviews in Table 1 (see section 2.2) were carried out as well as a literature study to research if similar problems had been solved before (see section 5.3). The first steps towards getting to know Android development were also taken.

4.1.2 Iteration 2

The main focus of the second iteration was on conducting the data analysis and writing a requirements specification. This specification is based on the results of the data analysis of the interviews and the observation. The observation was carried out in the beginning of this iteration (see section 2.2). Moreover, two architectural evaluation workshops (see section 4.6) were carried out, one was conducted by us, the other one was conducted together with employees from ASSA ABLOY (see Table 1). Also, the first functionalities for the prototype were implemented according to the architecture we started to design during this iteration.

4.1.3 Iteration 3

During the third iteration the focus was mainly on the architecture and the prototype. All requirements that were not yet designed for in the architecture, were considered in this iteration. A final architectural evaluation was conducted based on this architecture and finally, three evaluations of the Aioli prototype were performed.

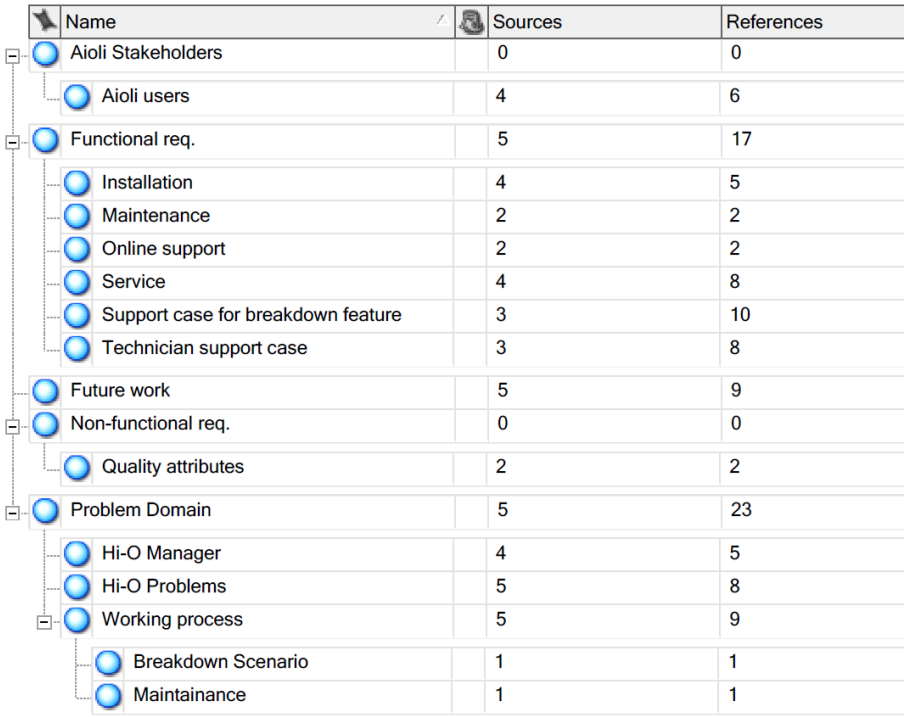
4.2 Data Analysis

In Design Research the Development and the Evaluation steps contribute to the knowledge base which is built up during the research project. As part of building this knowledge base a qualitative data analysis was carried out on the data collected during the interviews conducted in the first iteration (the five first interviews in Table 1, section 2.2) and the observation from the second iteration.

The analysis was carried out following the template approach for qualitative data analysis described by Robson [45]. According to the template approach a list of categories was selected and used for sorting the data into these categories. In this case, the chosen categories were Aioli

stakeholders, functional requirements, non-functional requirements, future work, and problem domain. Using the analysis tool Nvivo 9 [42] the data was labeled and sorted into these categories. The data in each category was then grouped into sub-categories. The result is shown in Figure 3. The sources column in the figure shows in how many different sources (interviews and observation) the categories and sub-categories are mentioned. It can also be seen in the references column the number of times each category and sub-category was brought up.

This technique allowed us to see relations in the data and indications of categories and sub-categories that were expressed as more important or occurred more frequently than others. This was used as the base for creating the problem domain and writing the requirements specification.



Name	Sources	References
Aioli Stakeholders	0	0
Aioli users	4	6
Functional req.	5	17
Installation	4	5
Maintenance	2	2
Online support	2	2
Service	4	8
Support case for breakdown feature	3	10
Technician support case	3	8
Future work	5	9
Non-functional req.	0	0
Quality attributes	2	2
Problem Domain	5	23
Hi-O Manager	4	5
Hi-O Problems	5	8
Working process	5	9
Breakdown Scenario	1	1
Maintainance	1	1

Figure 3: Qualitative Analysis Result

4.3 Domain Research

One of the goals of the interviews and the observation was to get a better understanding of the problem domain. This is reflected in the results of the data analysis where the category called “Problem domain” has the most references, see Figure 4. During the course of the data analysis, we decided that several references in the Problem Domain category should be in their own subcategory. This is shown in Figure 4. These categories include concerns around the Hi-O manager tool, current problems with the Hi-O door lock system and what today’s working process looks like when a Hi-O system needs to be installed or repaired.

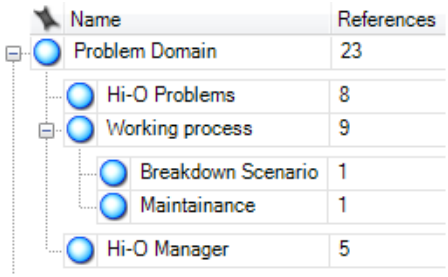


Figure 4: Domain Model statistics

From the analysis of the gathered data a domain model showing today’s situation was created as seen in Figure 5. This model was discussed with a contact person at ASSA ABLOY and changes were made to it accordingly. The figure shows the Hi-O door lock system and identifies the different relationships between the entities. This domain model is a representation of the Hi-O door lock system and how it is used, developed and maintained. The model was created in order to understand the problem domain better and to find out how the Aioli application would fit into this domain.

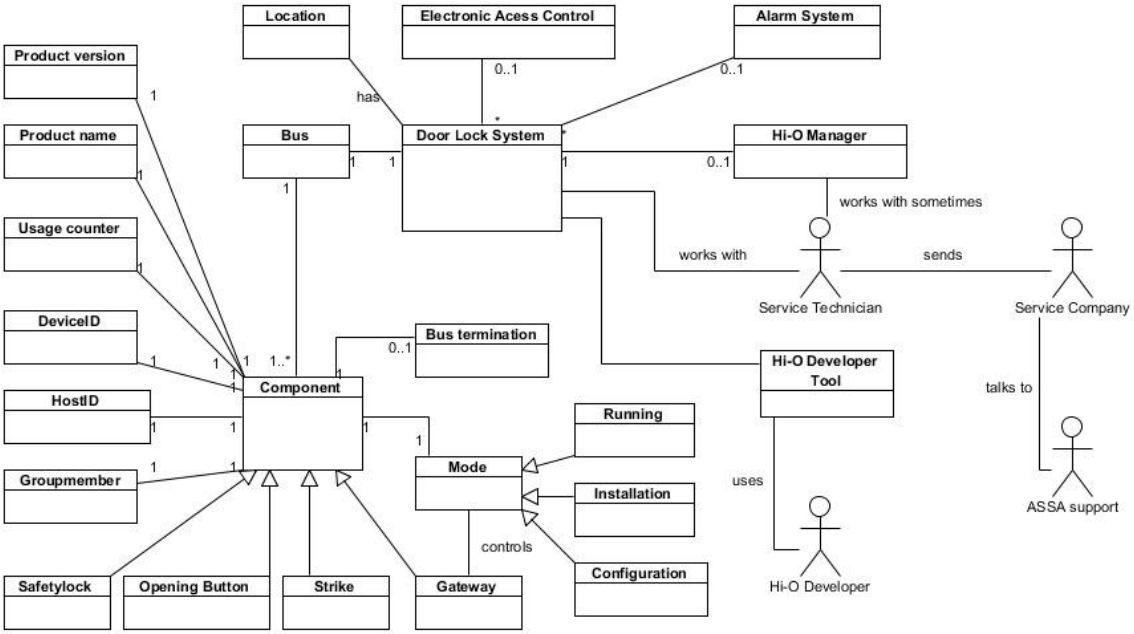


Figure 5: Domain Model of Hi-O in today’s situation

As the figure shows, components that are connected on a bus can be of different type. Each component has a set of attributes such as the product name, version, device ID, and host ID. The components have a usage counter which indicates the number of cycles a component has been run. Components have a group membership which indicates if the component is on the inside or the outside of a door. Furthermore, each component can be in a specific mode either in Installation, Configuration or Running. The gateway can control the mode for all other components and set them to the same mode to achieve a functional system.

Additionally, other systems can be connected to the door lock system, such as an EAC or an alarm system. These have the purpose of adding functionality.

In case of installation or service, a service company sends a service technician who works with the door lock system either directly or via the Hi-O Manager which aids in the process of changing the door configuration. This tool however, is for various reasons not used widely today. Developers of Hi-O maintain and extend it via the Hi-O Developer tool.

4.4 Requirements Specification

In the following sections we present the requirements in three different forms. First, the functional requirements are expressed in the form of functional scenarios that explain what Aioli and its users should be able to accomplish. Second, all stakeholders of Aioli and how they are affected by the application are presented. Finally, the non-functional requirements (also called quality attributes) for the application are stated.

4.4.1 Functional Scenarios

Before the data collection process was started, the brainstorming session from iteration one was carried out to find out in which situations this smartphone application could be of help. It was suggested that the application could prove useful during the process of installation/breakdown of the Hi-O door lock system. The installation/breakdown scenario also included the feature of an online support session, where the support center could establish a connection with the smartphone in order to diagnose the problem.

As this project was conducted in an iterative manner, the scenarios were constantly evolving as a result of the feedback that was first received from the contact persons at ASSA ABLOY and later from all types of data collection that were conducted. Therefore, after gaining more insight into the problem domain it was realized that the installation/breakdown scenario, was too complex to be its own scenario and was therefore split up into a few separate scenarios; installation, breakdown, service and online support session. As stated in the general requirements (see Appendix A) the application should have different types of users who should be allowed to carry out different functions. One user group consists of the service technician and the other group consists of the superintendent. The breakdown scenario was therefore further divided into two scenarios allowing both user groups to access their specified functionalities.

A presentation of all scenarios is given below; it is a summary of the functional requirements which were found during the interviews and observation. The requirements specification can be found in the Appendix A.

The general requirements stated in Appendix A are either applicable to the whole application or are prerequisites for the scenarios. Since it was decided that the application should support different user groups, the users will have to log on to the application in order to determine which functionality should be accessible. Another general requirement is that the connection between the Aioli application and the Hi-O door lock system shall be established via Bluetooth and in case the Bluetooth connection is lost while working with the Hi-O door lock system, Aioli shall automatically reconnect and notify the user in case the connection could not be established. The last general requirement is that the user should be able to save his contact information in a settings screen. This information is attached to each support case that is sent to the support in order

for the support to be able to contact the service technician. The following sections describe the functional scenarios. The Installation, Service, Support Case Service Technician and the Online Support Session scenarios are meant to be used by the service technician user group. The Breakdown Support Case scenario on the other hand is supposed to be used by the superintendent user group.

Service
When a Hi-O door lock system breaks down but still has the possibility for a Bluetooth connection, it is possible for the service technician to connect Aioli to the door lock system and retrieve information from it in order to diagnose the problem. During the diagnosis the technician will be asked to follow a few steps which will send commands to the door lock system. Aioli will give several options towards what the reason behind the error could be. Also, the service technician has the possibility to change settings such as operation mode in order to configure the system. Regardless of if the Bluetooth connection was successful or not the service technician has the possibility to search in a list of common problems to see if a colleague had a similar problem before and how it was solved. If the problem cannot be resolved, the user has the possibility to send a support case to the ASSA support.

Installation
During the installation of a Hi-O door lock system, the application will present a checklist to the service technician with all the installation steps. This will serve as a reminder and simplify the process. The service technician can add comments to this checklist. The list can be saved locally on the smartphone, but will be sent to an external database once the installation has been completed. Furthermore, the service technician will be able to search for previous installation documentations and see the comments of other service technicians. Once the door lock system has been set up and Aioli has successfully connected to it, the user will be able to see all components installed in the door and will furthermore be able to change the system settings, such as the operation mode. In case something goes wrong during the installation, the technician will be able to switch to the service mode where the problem can be diagnosed.

Support Case Service Technician
Since Hi-O is one of many door lock systems, service technicians can run into problems which they have not come across before. In this situation it is of great importance to get help from the support team. If retrieving component information, information from the diagnostics run through and the current operation mode of the Hi-O system in the Service scenario was successful, then it is included in the support case. Even if it was not possible to retrieve all this information from the Hi-O system, the service technician has the possibility to add information to enrich the description of the experienced problem. This added information can be in the form of pictures, videos, sound and text. This support case will then be sent via email to the ASSA support.

Online Support Session
This scenario is to be used after the service technician has sent a support case and the person at the service center needs real time information. In this case the service technician can video call the person at the ASSA support who then can access the door lock system via the smartphone application in order to facilitate remote system diagnostics. This is to be able to instantly help the service technician with the installation or service of the Hi-O door lock system. The service technician will be able to stream audio and video to visualize the problem to the ASSA support.

Breakdown Support Case
Once the superintendent discovers a problem with a Hi-O door lock system, he is able to use Aioli to create a support case which he then can send to the service company that is responsible for the security of the door lock system. The superintendent uses Aioli to connect to the Hi-O system and retrieve door component information. If the retrieving was successful, the information

is included in the support case. This information will be hidden from the user in this application mode, because it is sensitive information and shall only be accessible by employees of the service company. However, the user has the possibility to add additional information to the support case. For example, the user can add pictures, video, sound and text to describe the problem he has experienced. All this additional information has the purpose of giving a rich impression of the door lock system problem, so that an informed decision can be made about the problem and the service technician knows exactly which replacement parts to bring.

Figure 6 is a model of the identified entities and it demonstrates the Aioli application during runtime. The figure shows which actor initiates the different scenarios and which scenarios respond to the actors. Furthermore, the possible flows between the scenarios are visualized. For example it can be seen that after completing the Installation scenario it is possible to move to the Service scenario in order to test the installation. Likewise, it is possible for the service technician to initialize the Support Case scenario from the Service scenario. In this case it is also possible to move back and forth between the two scenarios.

The figure also shows that the scenarios rely on the possibility to make use of the smartphone platform and other applications installed on it in order to add different kinds of media to a support case. Finally, it visualizes also that the connection between the Aioli smartphone application and the physical Hi-O door lock system is formed via Bluetooth.

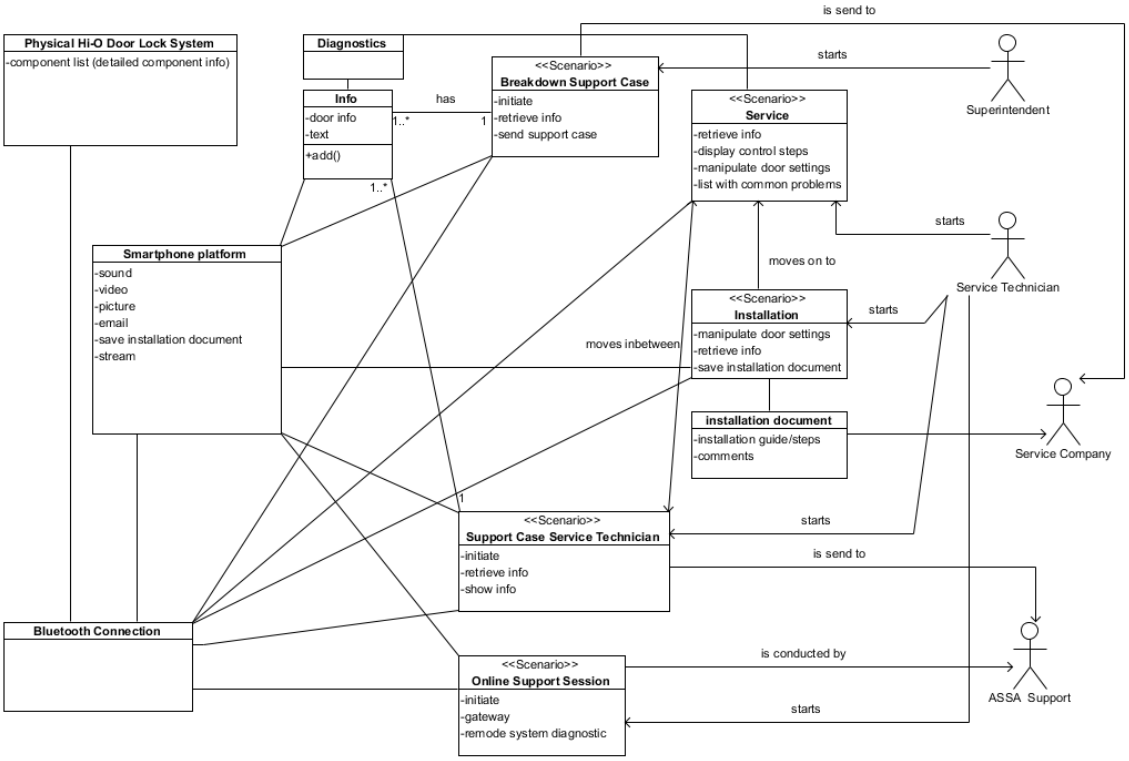


Figure 6: Problem Model for the Aioli Application

4.4.2 Stakeholders

The stakeholders defined for the Aioli application are grouped into four categories: the direct users, the indirect users, the customer and the developers. These are presented in Table 2 and described in more detail in the following paragraphs.

The direct users are the ones who will use the Aioli application. There are two types of direct users, the first type of stakeholders is the superintendent. The superintendent user group includes

Table 2: Aioli Stakeholders

Category	Description
Direct Users	Superintendent
	Service Technician
Indirect Users	ASSA support
	Service Company
Customer	ASSA ABLOY
Developers	Aioli Developers

any person who would, prior to Aioli, be responsible to call the service company in case a door lock system breaks down. An example of another user belonging to the superintendent user group is a security responsible. The superintendent user group will use Aioli to report a breakdown of a door. The second type of direct user is the service technician who will use Aioli on site during service and installation of a door lock system.

The indirect users are users who receive information from the Aioli application. Also in this category there are two types of user groups, the first group is the ASSA support who will receive support cases from the service technician. They will also be able to set up a remote connection with the Aioli application in order to give online support to the service technician. The second indirect user group is the service company who receives support cases from the superintendent user group.

The customer is ASSA ABLOY who expressed a desire for the Aioli application. They had ideas and suggested requirements for how Aioli could contribute to the Hi-O product.

The developers group includes both us who designed the architecture and implemented the prototype and future developers who will implement the complete Aioli application.

4.4.3 Quality Attributes

The following scenarios characterize and specify the chosen quality attributes for the Aioli application. The specification of the quality requirements follows the standard put forward by Bass et al. [16].

Figure 7 shows that the specification of a quality attribute involves several details. For example one needs to know the source that generates the stimulus, which can be a person, a computer, or any other actor. The stimulus is the event that needs to be handled when it arrives at the application. The environment specifies the condition and circumstances in which the stimulus can occur, whereas the artifact marks the areas of the application that are affected by the stimulus. To handle the stimulus, a response and response measure need to be determined in order to know whether the quality attribute has been fulfilled or not.

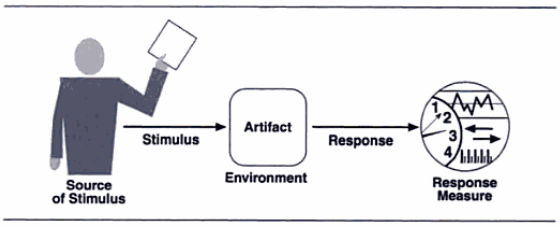


Figure 7: Quality attribute parts (From [16])

The quality attributes which have been chosen for this project are Security, Modifiability, Portability, and Usability. The following sections give a short description of all scenarios, the detailed

quality attributes can be found in Appendix B.

Security, Scenario 1: Authentication/Authorization
This scenario deals with how to make sure that only authorized and authenticated users can access and use the Aioli application. In order to be granted access, the potential user needs to enter his personal login data. This login functionality is the first layer of protection and gives access only to authorized users. Additionally when a potential user enters the password it is not displayed on the screen in order to prevent others from copying the password.

Security, Scenario 2: Data Confidentiality
Man-in-the-middle attacks, where an intruder tries to access the communication link between Aioli and the door lock system. To protect the user's Aioli application session all data which is sent over the communication link is encrypted.

The precondition for the usability scenarios is that the user is familiar with the smartphone in general and does not need training in getting accustomed to the smartphone platform.
Usability, Scenario 1: Recovery
When a user has inserted information and accidentally presses a button that cancels the operation, the inserted information will not be lost. When the user returns, the information is presented to him. Thus, he does not have to enter everything again.

Usability, Scenario 2: Learnability
Since this application will mainly be used by service technicians who have to concentrate on many things when installing or serving a Hi-O door system, it is of importance to provide easy navigation through the application. Each functional scenario that is described in section 4.4.1 is mapped to the application. Thus, depending on what the user would like to do, be it installation or service of a door, there are application modes which guide the user through the application.

Modifiability, Scenario 1: Ease of Modification
This quality attribute is about how to ensure that necessary modifications to the application can be made easily. For example, if the functionality of the Hi-O door lock system changes then modifications need to be made to Aioli. Also, it might be possible that the user interface needs to be adapted. For a developer with experience in Android development and knowledge about Aioli, a major change should not take longer than ten working days. A major change is a modification that affects at least two layers. A minor change should not take longer than three working days and affects one component (see sections 4.5.1 and 4.5.2).

Portability, Scenario 1: Change Platform
It cannot be assumed that all service companies use Android smartphones. Yet, they should be able to use the Aioli application. Hence, it is likely that the application needs to be able to run on another type of platform, for example the iPhone platform iOS [9]. Although, this change can only be planned for on an architectural level since every smartphone platform has a different implementation language, API and design recommendations. Therefore, the use of the Android API is separated from the application logic (see chapter 4.5).

4.5 Architecture Design

The result of the Aioli architecture was documented using the architectural blueprint called the "4+1 View" [32], which was designed by Philippe Kruchten in 1995. The reason why this blueprint was chosen is because it allows visualization of the architecture from multiple stakeholders view points. All architectural design decisions are organized according to four views; Logical View,

Development View, Process View and Physical View. The “plus one” view, or fifth view, is the scenarios which were presented in section 4.4.1. The scenarios bind the other four views together and drive the architectural development forward. The four views are explained and an example of the realization of one scenario is presented in the following sections.

For every view the blueprint suggests different notations and tools that can be applied. Yet, Kruchten states that these suggestions are generic and can easily be replaced with other notations and tools. Thus, it was decided to use the UML 2 notation [39] for the design of the views. However, it is important to point out that some diagrams are not designed with pure UML notation standard, but have been modified in order to give detailed clarifications. To create the diagrams the tool Visual Paradigm for UML [55] was used. Figure 8 shows the different views, how they correspond to each other and which stakeholder perspective it describes. The different views will all be described in terms of the Aioli application architecture in the following sections.

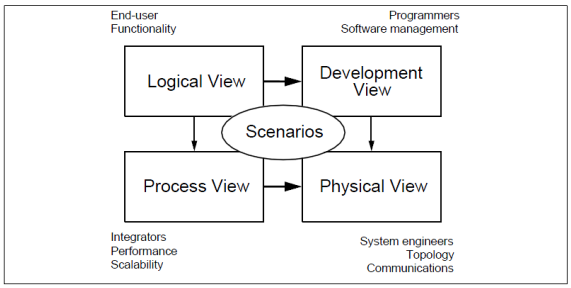


Figure 8: The "4+1" View (From [32])

4.5.1 Logical view

In the logical view the focus is put on the functional requirements of the application. This means that this view defines the granularity to which the functionality is divided into separate classes. Figure 9 can be seen as a simplified class diagram where not all functions and attributes are included. However, it is extended to also show which classes are responsible for the communication with external systems, such as the databases, the ASSA support, and the Hi-O door lock system. Moreover, it shows which classes are responsible for the communication with external mobile applications, such as the camera, gallery, and email client applications. In order to group functionality and facilitate coherence, it was decided to put the different classes into packages. Each package holds classes that are responsible for functionalities that lie in the same category.

The Android standard specifies that when declaring the graphical layout, the user interface elements are declared in XML [57] files which are instantiated and handled in Java [31] files. This allows for a clean separation of the implementation of user interface elements and their logic. All classes that deal with the graphical user interface are grouped into three packages; GUI, GUI_ServiceTechnician, and GUI_Superintendent. These are explained in the first three sections below, followed by all other packages for Aioli.

GUI — Package
This package contains the main screen that both user groups see when the Aioli application is started. In this screen the user is asked to enter his login information in order to be authorized as a legitimate user of the system. When the login information is found valid, it needs to be determined which kind of user access should be given. If the user is a service technician the functionality in the GUI_ServiceTechnician package is given access to, if the user is a superintendent, the functionality in the GUI_Superintendent is chosen.

GUI_ServiceTechnician — Package
The graphical user interface in this package facilitates the access to all functionality for the installation, service, support case and the online support scenarios. The classes in this package communicate with the Controller class in the Coordinator package. Also this package is connected

to the GUI_Logic package. Furthermore, this package is responsible for the communication with the third party programs in order to meet all requirements.

GUI_Superintendent — Package
The graphical user interface in this package facilitates the access to all functionality for the breakdown support case scenario. The classes in this package communicate with the Controller class in the Coordinator package as well as some classes in the GUI_Logic package.

HardwareComponentDetails — Package
In this package the elements of the physical door lock system are defined. The Component_Representation class holds all attributes that a physical component, which is installed in the Hi-O system has. Such attributes can be the amount of times a component has been used, the name, version, device ID of the component, and the information that states if the component is on the inside or the outside of the door. Since Hi-O is a door environment that can contain one or two doors, a gateway is needed which connects these doors to a sub-central. Thus, the Gateway_Representation class is a type of component which resembles this physical gateway. The DoorLockSystem_Representation class holds an array of all components that are installed in the door lock system. This package is only connected to the Coordinator package via the DoorLockSystem_Representation class.

Coordinator — Package
The Controller class in this package processes and delegates commands that are received from the graphical user interface packages. Since it is of great importance to avoid communication errors, the Controller is designed to be a Singleton class. This design decision ensures that there is only one Controller at all times. The Parser class has the responsibility to read and interpret the data which it receives from the physical door lock system via the ThreadManager class. The Commands class stores all commands that are sent to the Hi-O system in order to receive information. The ThreadManager class is responsible for the communication with the BluetoothConnector in order to send and receive information between the Aioli application and the Hi-O door lock system.

GUI_Logic — Package
It was decided that this package is responsible for providing the logic for carrying out the tasks in the different scenarios that can be initiated from the graphical user interface. The InstallationDocument_Logic class provides the ability to show all the steps in the installation document and it provides the possibility to enter comments to every step that shall be carried out. The HealthCheckDiagnostics class is responsible for diagnosing the information retrieved from the components. Furthermore, it gives suggestions towards what the possible errors could be according to the diagnostics. Similar is also the purpose of the Service_Logic class which provides the ability to put together all information for every component. The SupportCase_Logic class is responsible for collecting the data that is sent in a support case email to either the ASSA support or the service company support center. Finally, the OnlineSupport_Logic facilitates the ability of real time online support. The UserVerifier_Logic class handles the logic behind the user verification.

BluetoothConnector — Package
As the name suggests the BluetoothScanner class in this package has the responsibility to search for all available Bluetooth devices that are in range. When the user has chosen a Bluetooth device he wants to connect to, the BluetoothScanner sends the information of this device to the Controller class. Once a Bluetooth connection between a physical Hi-O door lock system and the Aioli application on the smartphone is established, the BluetoothConnectionManager manages this connection. This class enables the possibility to send information between the smartphone and the Bluetooth device.

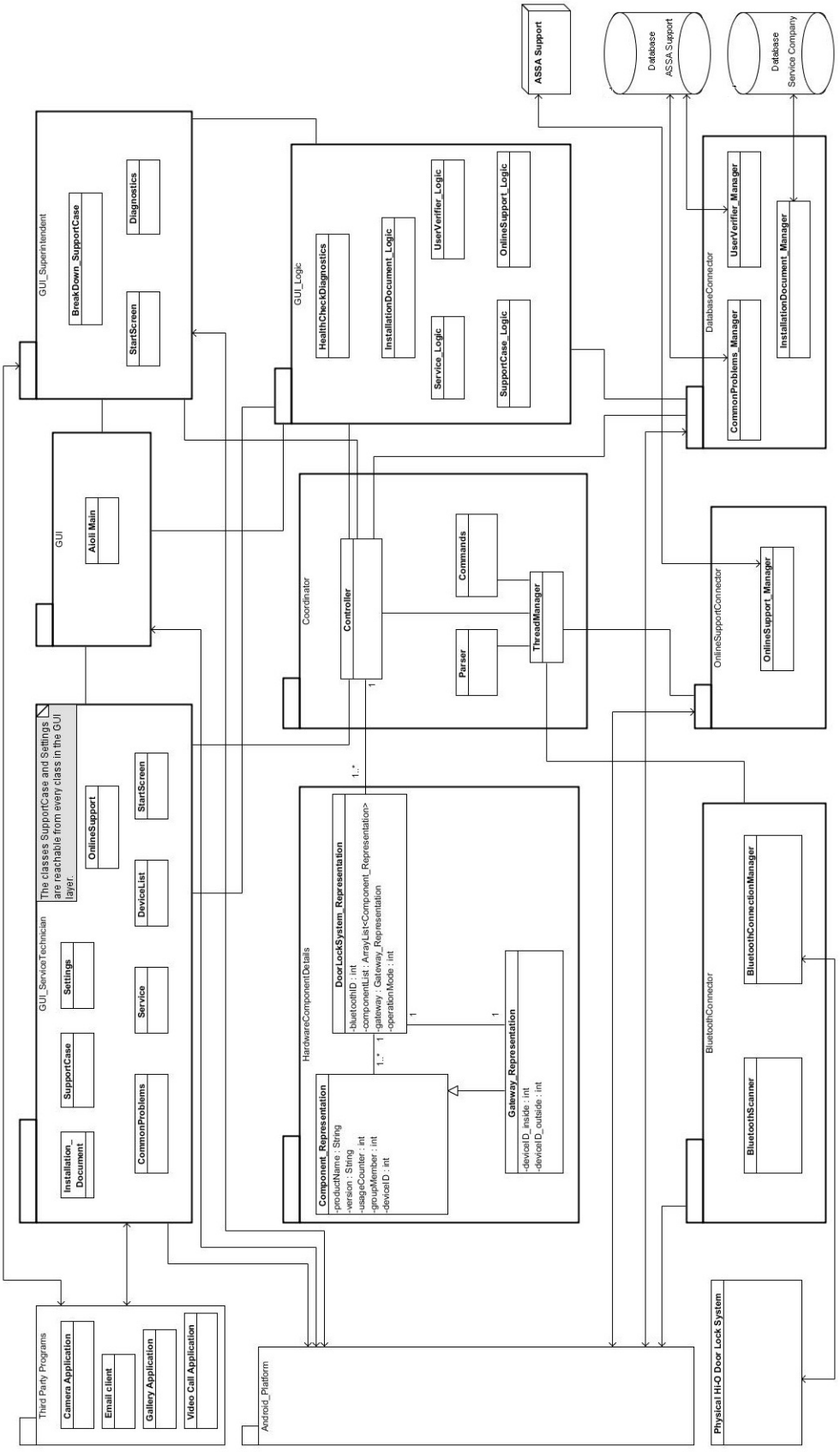


Figure 9: Logical View

DatabaseConnector — Package
The InstallationDocument_Manager class is responsible for the communication with a database maintained by the service company. The purpose of this communication is that a service technician can view installation documents from previous installations in order to see if the problem that he is experiencing with the Hi-O door lock system has been solved previously. Also, he has the possibility to save his installation document to the database. Similarly, there is a database that is maintained by ASSA ABLOY that lets the CommonProblems_Manager as well as the UserVerifier_Manager classes communicate with it. Via the CommonProblems_Manager class the service technician is able to retrieve a list of experienced Hi-O problems. This list is editable by the service technician and ASSA ABLOY. As the class name suggests, the UserVerifier_Manager class sends login information, which shall be verified, to the database.

OnlineSupportConnector — Package
The OnlineSupport_Manager class is responsible for the wireless communication with the ASSA support during the online support scenario.

All graphical user interface packages as well as the connector packages make use of the Android platform in order to meet the desired functional requirements. Therefore, there is a package called “Android platform” which symbolizes the Android API that was used for the prototype. The package called “Third Party Programs” shows different Android applications that are installed on the smartphone and that are made use of by Aioli. The camera application is used to either take a picture or record a video for the support case scenario. The gallery application is used to fetch a picture or video from the gallery that has already been taken at a previous occasion. The email client application is used for sending the support case to either the service company or to the ASSA support. Finally, a video call application such as Skype [49] is used during the online support scenario so that the service technician and the person at ASSA support can have a video conversation.

4.5.2 Development view

The development view focuses on the organization of the software module in the development environment. According to Kruchten it is packaged into small chunks which are then assigned to groups of developers [32]. The design of the development view is shown in Figure 10. It is divided into four areas, shown with the gray boxes, which state who is responsible for developing the different parts relating to the Aioli application. The Third Party Apps are obtained from the Android Market [30] or are delivered with the operation system, thus there are many different developers who create these application. The Android Smartphone Platform is developed by the Open Handset Alliance [40] and Aioli uses the Android API. The databases and the application for the ASSA support are thought to be developed by ASSA ABLOY who have already developed the Hi-O system. In the middle of the figure is the Aioli application which is developed by future Aioli developers.

The package for Aioli in Figure 10 corresponds to the logical architecture (see section 4.5.1). It demonstrates that the Aioli application is built using the layered architecture pattern. The architecture consists of three layers, two of these interact with the Android platform. The packages shown in the logical view are divided into these layers according to their responsibilities. The Graphical User Interface Layer contains the GUI, GUI_ServiceTechnician and the GUI_Superintendent packages. The Application Logic Layer holds the HardwareComponentDetails package, the Coordinator package as well as the GUI_Logic package. Finally, the Communication Layer contains the BluetoothConnector package, the OnlineSupportConnector package as well as the DatabaseConnector package. Hence, the responsibilities of these layers are according to the responsibilities of the packages. The Graphical User Interface layer is responsible for the layout of the application and passing on commands the user has given. The Application Logic layer has the responsibility of data processing (e.g. parsing the information received from the BluetoothConnector) and providing the logic to most functionality for the different scenarios. As the name suggests, the

Communication layer is responsible for establishing and maintaining a connection with the physical door lock system to which it sends commands and from which it receives data. Furthermore, this layer is responsible for the communication with external databases and the ASSA support. Since it is a layered architecture only neighboring layers are allowed to communicate with each other, any other form of interlayer communication is not permitted.

The Android Smartphone Platform is modeled separate from all layers to emphasize the portability quality attribute which the Aioli application shall fulfill. It shall be possible to switch to a different smartphone platform. As can be seen in Figure 10 the Application Logic layer does not interact with the Android Smartphone Platform, which alleviates the transition to a different platform.

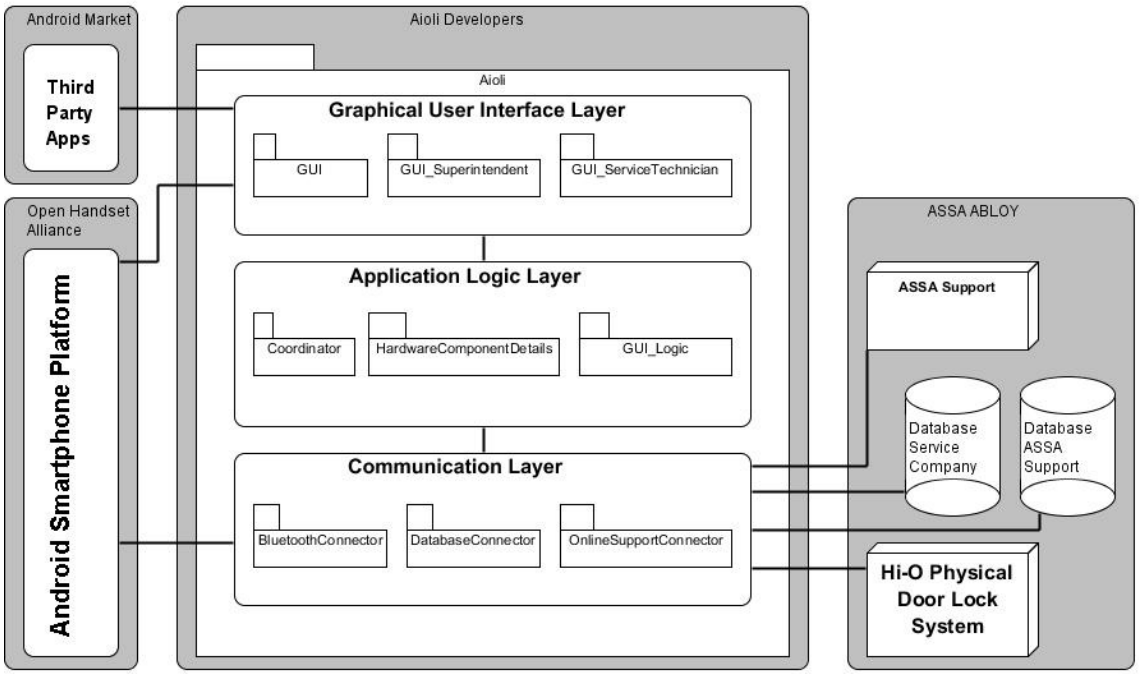


Figure 10: Development View

4.5.3 Process view

The process view shows the different threads in the application and how they interact. On Android each application executes in one process and on one main thread by default [3]. This main thread is responsible for dispatching events to the user interface and therefore cannot be blocked since it would lock the user interface [3]. The Aioli application runs in one process but uses several threads for the Bluetooth connection and for the network connection.

Figure 11 visualizes which classes from the logical view (see section 4.5.1) the different threads are executed in. The colored classes in the figure show which thread they are executed in, as can be seen most classes are executed in the Main thread. The CommunicationManager thread is responsible for starting other threads used for communication, it also sends commands and parses the results. The two Bluetooth related classes are executed in two private classes within the BluetoothConnectionManager class. The NetworkConnection thread is used for setting up the communication with ASSA support.

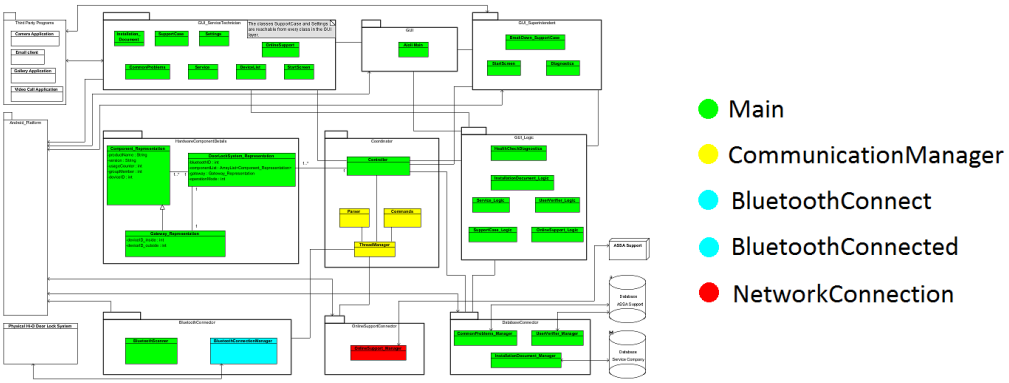


Figure 11: Process View mapped to Logical View

The sequence diagram in Figure 12 displays how the Bluetooth connection with Hi-O is set up and how the communication is handled in the threads. In this sequence diagram it is assumed that the Bluetooth device which Aioli should connect to is already found and that the Main thread knows which device to connect with. For setting up the Bluetooth connection the Main thread creates the CommunicationManager thread from the Controller class. It then sends a message containing which device Aioli should connect to. The CommunicationManager creates the BluetoothConnect thread with a parameter saying which device it should connect to. The BluetoothConnect is responsible for setting up a socket connection with a Bluetooth device. If a connection was set up successfully the socket is returned to the CommunicationManager and the BluetoothConnect thread is stopped. The CommunicationManager then uses the socket to start the BluetoothConnected thread, which is responsible for the communication with Hi-O. The actions performed by the BluetoothConnected thread are displayed within the nested sequence diagram called Bluetooth Communication. This nested diagram shows that the BluetoothConnected thread has a write function which takes a command as input, the command is sent to the Android Platform which sends it to the Hi-O Bluetooth adapter. Hi-O then replies to the Android Platform which passes on the result through the BluetoothConnected thread to the CommunicationManager. The Hi-O Bluetooth adapter may split the result into several messages, in which case the Hi-O Bluetooth Adapter calls the Android Platform once for each message. The Parser class in the CommunicationManager decides when it has received the entire result. When all messages for one result is received, the complete result is sent to the Main thread.

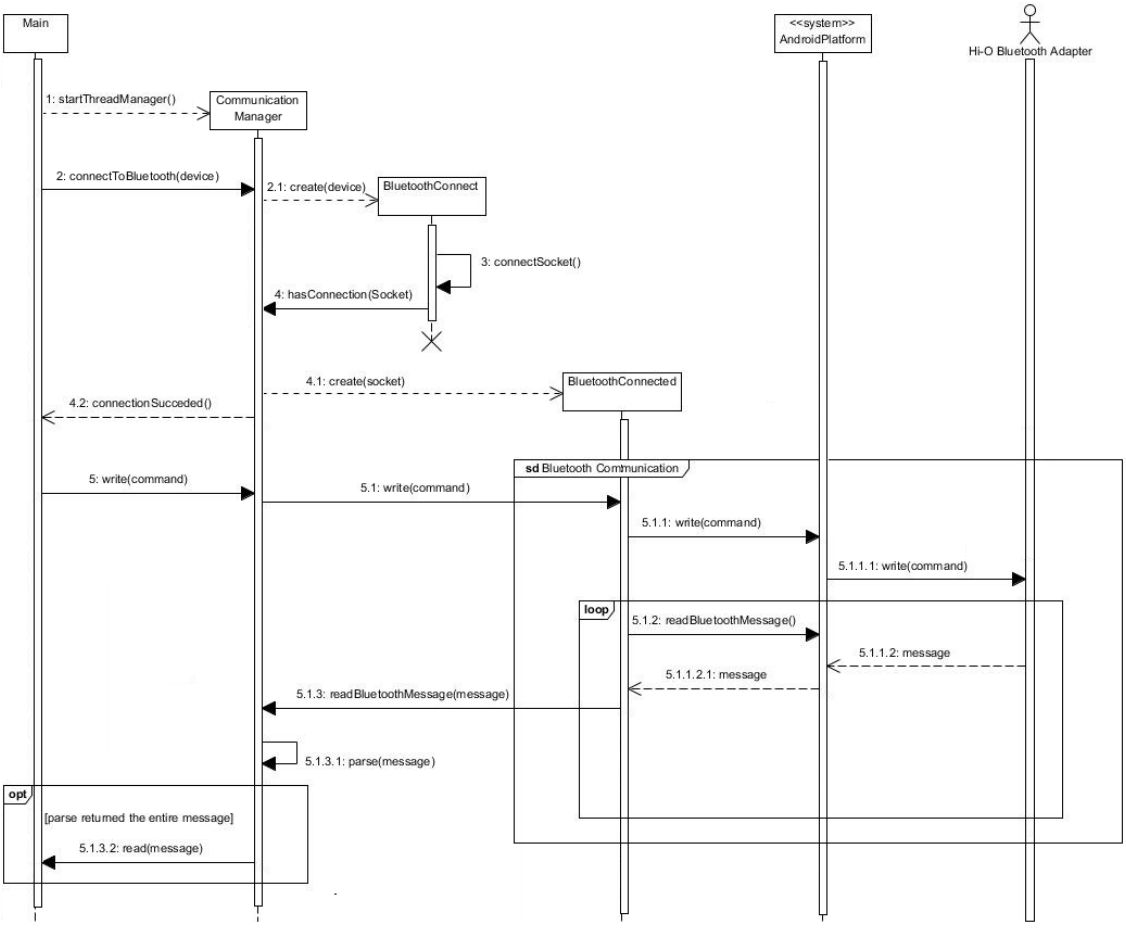


Figure 12: Process View - Bluetooth Connection

To describe the process of setting up the Online Support scenario (see section 4.4.1) the sequence diagram in Figure 13 has been developed. In this sequence diagram it is assumed that the CommunicationManager thread is running, the Bluetooth connection is already set up as shown in Figure 12, and the BluetoothConnected thread is running. The Main thread calls the CommunicationManager to start the remote connection to Hi-O by executing the function startOnlineSupport. The Main thread then continues by using a third party application to start a video call with the ASSA support. The CommunicationManager creates a new NetworkConnection thread which starts a network connection with the ASSA support who then can send commands to Hi-O via this connection. When the NetworkConnection thread receives a write call with a command from the ASSA support, it calls the write function in the BluetoothConnected thread with the command as a parameter. The BluetoothConnected thread then handles the communication with the Hi-O system in the nested Bluetooth Communication sequence diagram which is a part of Figure 12. When the BluetoothConnected thread receives a response from Hi-O, it calls the CommunicationManager with a message from Hi-O. This message is parsed (as described for Figure 12) and if the entire result was received, it is sent to the NetworkConnection thread which passes it along to the ASSA support. The Main thread is responsible for canceling the network connection when the video call is finished.

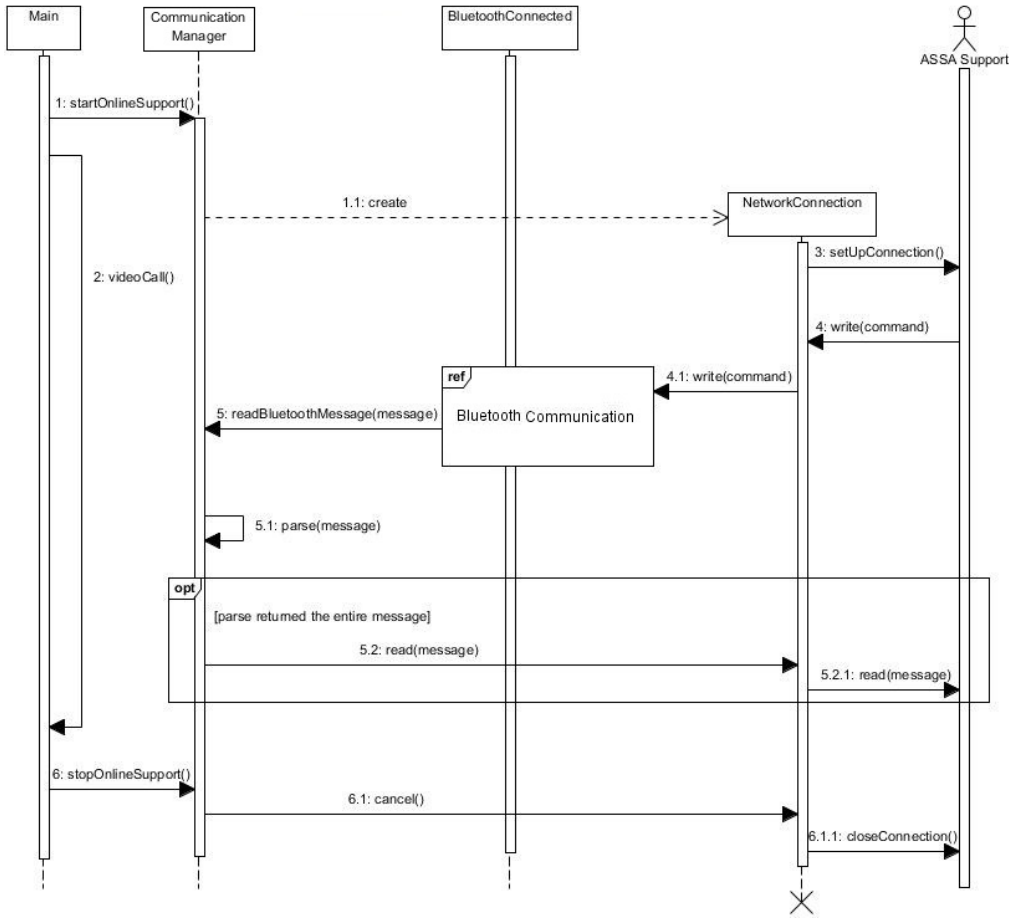


Figure 13: Process View - Online Support

4.5.4 Physical view

According to Kruchtens 4+1 blueprint [32] the physical view maps the software to the hardware. Therefore, a deployment diagram was created to show this relationship. In the diagram the types of communication between the hardware elements are also shown; dashed lines symbolize wireless communication and solid lines represent wired communication. In the middle of the diagram is the smartphone device which runs the Android platform. On the platform the Aioli execution environment is displayed along with the third party applications used by Aioli. The execution environment for the Aioli application contains the logical view (see section 4.5.1) to depict the relation between the two views. To the left is the physical Hi-O door lock system and to the right are the external databases as well as the support system of ASSA's support. Aioli communicates with the Hi-O system via Bluetooth. With ASSA's support and the databases it communicates via WLAN.

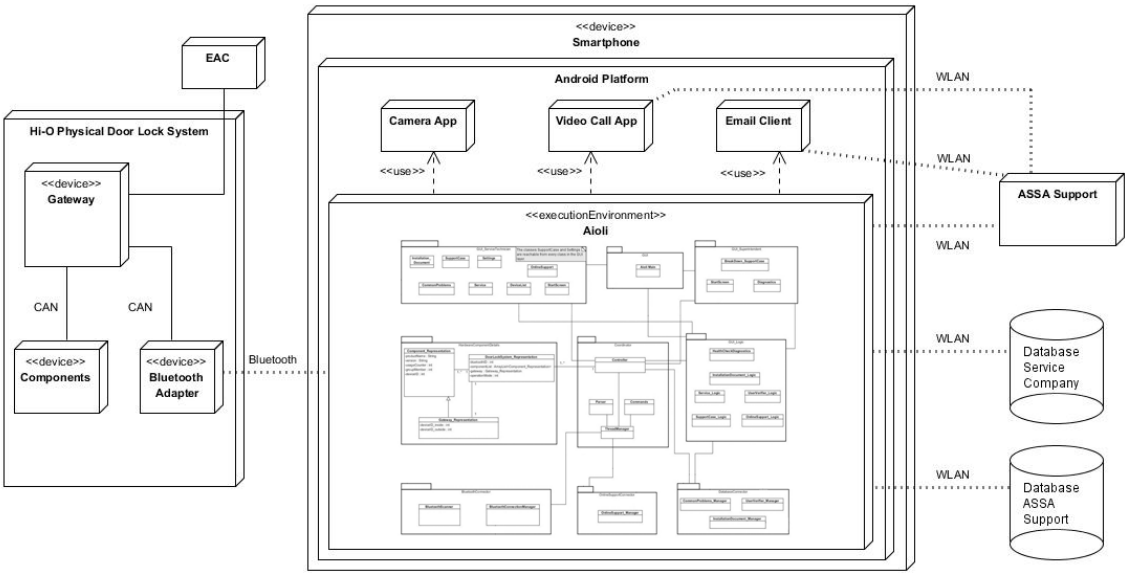


Figure 14: Physical View

4.5.5 Scenario Realization

According to Kruchten the scenarios presented in section 4.4.1 constitute the fifth view, which serves "as a validation and illustration role after this architecture design is complete" [32]. In order to show how the scenarios are met in the architecture it has been decided to demonstrate this with the Service scenario. In this demonstration it is assumed that a Bluetooth connection between Aioli and a Hi-O door lock system already exists.

As can be seen in Figure 15 the Service scenario can be carried out as follows. The service technician indicates via the graphical user interface that he would like to retrieve the detailed component information from the Hi-O door lock system. The Service class needs the Service_Logic class to compute the command, which is then sent to the Controller class. This class passes the command on to the ThreadManager class which talks to the BluetoothConnectionManager class. This class has the responsibility of talking to the physical Hi-O door lock system. The BluetoothConnectionManager will then send the response back to the ThreadManager, which sends the response to the Parser class that has the responsibility of extracting the needed information. This is then sent to the Service class so that it can be displayed to the service technician.

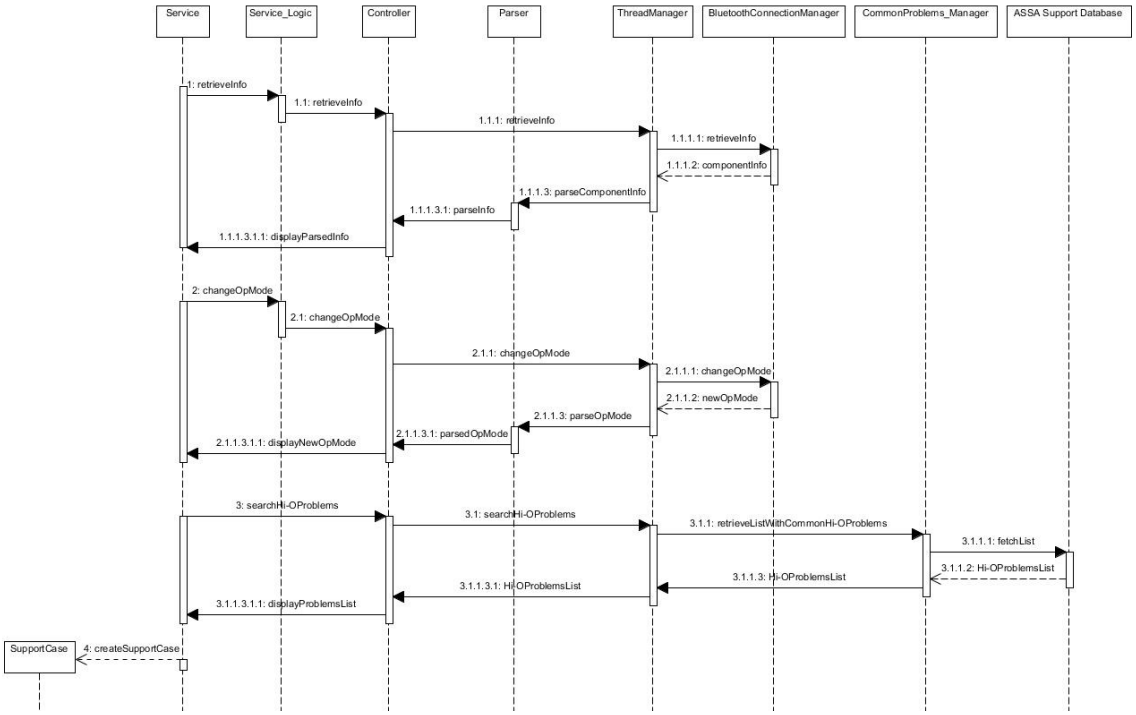


Figure 15: Service Scenario realization

This procedure clearly shows how the three layers communicate with each other. Moreover, the same procedure is used for the diagnostic part (not repeated in the figure), where the service technician is instructed to carry out a set of steps in order to retrieve information from the Hi-O system. The exception here is that when the Parser class has processed the information, it is sent to the HealthCheckDiagnostics class (see section 4.5.1), which suggest possible reasons to the service technician.

A similar communication procedure is used for changing the operation mode, as can be seen in Figure 15. However, when the list for the common Hi-O problems needs to be retrieved the ThreadManager will send the command to the CommonProblems_Manager, which establishes the connection with the external ASSA support database.

When the service technician cannot solve the problem that he is experiencing with the Hi-O system, he has be possibility to create a Support Case from the Service class.

4.6 Architectural Evaluation

The technique which was used to evaluate the quality of the Aioli architecture is called “architectural Software Quality Assurance” or in short “aSQA” [20]. It was published by Danish researchers in 2010 and tested at a few different companies in Denmark [20]. It is a new method which we were not familiar with. Still, it seemed attractive and fitting for the Aioli project, aiming at the aspects which needed to be evaluated. The technique focuses on evaluating if the quality attributes are achieved to the desired level in the architecture design. Another reason why this method was chosen was because “aSQA is a novel and viable technique for continuously assessing, controlling and balancing quality attributes in a development project” [21], hence, it fitted very well with the Design Research process that was used for this project.

Since the evaluation technique focuses on the fulfillment of the quality attributes a metrics for measuring this achievement needs to be used. The creators of aSQA give two suggestions for quality frameworks; either one uses a “metrics-based measuring technique according to the framework of Clements et al.” [41], or one can also use the “quality attribute scenarios as a basis for evaluation” [21] such as the framework suggested by Bass et al. [16]. Due to the fact that the second option was already chosen for the specification of the quality attributes (see section 4.4.3), it was an obvious choice to base the aSQA evaluation on the framework by Bass et al.

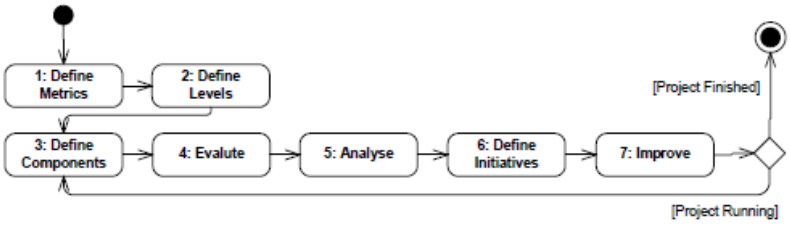


Figure 16: aSQA Process Steps (From [21])

As Figure 16 shows, the evaluation technique has several steps that need to be followed. After deciding upon the metrics, levels need to be defined which specify the scale of measurement. The ordinal scale which was decided upon looks as follows:

1 = Unimportant
2 = Slightly Important
3 = Important
4 = Very Important
5 = Critical

This scale allows for a comparison of values for the quality attributes. In order to conduct a successful evaluation, a set of software components is needed which a level of quality will be assigned to during the evaluation. For the Aioli architecture it was decided that the packages from the logical view were fit for this technique, since they are modular and have specific purposes they are appropriate for an evaluation.

The fourth step in the aSQA flow is the actual evaluation which consists of several actions. For every component one considers every quality attribute scenario and assigns a so called “target-level” to it. This target specifies “what is currently seen as the needed quality level for a specific component” [21]. After that an “importance-level” is set using the same measuring scale. This importance level defines the priority of each quality attribute in a component. Finally, a “current-level” is assigned for each quality attribute in each component, which states the current quality level of a component. When all this data was collected, the health level of each quality attribute in each component is calculated using the formula specified in aSQA [20]. The “formula assigns health from 1 to 5 with 5 being excellent health” [20] the result of these calculations give an indication of which architectural aspects require further attention. The focus level is also computed

with a specified aSQA formula and gives a more specific indication of which aspects need to be improved, as this calculation also takes the importance level into consideration. For the focus level, a level 5 means that a component requires most attention and a level 1 indicates that a component requires least attention.

The next step in the aSQA flow, is to analyze the data and point out the components that need improvement; “the lower the health level is and the higher the importance level is, the higher the focus level will be and thus highlighting that raising the level of the quality attribute for the component is important” [20]. In step six, initiatives are defined that need to be taken into consideration in order to achieve an improvement in quality. The final step is all about making these changes happen.

As can be seen in the aSQA flow in Figure 16, steps three to seven are repeated iteratively until the project is finished. This makes it possible to achieve an architecture design that fulfills the quality aimed for. For the Aioli architecture three evaluations were conducted. The first and third evaluations were carried out by us and the second evaluation was conducted together with the contact persons at ASSA ABLOY.

Evaluation One

The first evaluation mainly had the purpose of understanding aSQA and learning how its technique is put into practice. In this evaluation the architecture was still under development (see Figure 17) and as seen in the result table in Figure 18 there are some components which have a high number for target but a low number for current, this indicates that these were not yet implemented in the architecture at that evaluation. No changes were made to the architecture after this evaluation, it was only carried out as a test run to get to know aSQA and check the state of the architecture.

The following are the components that were used for the first and second evaluation.

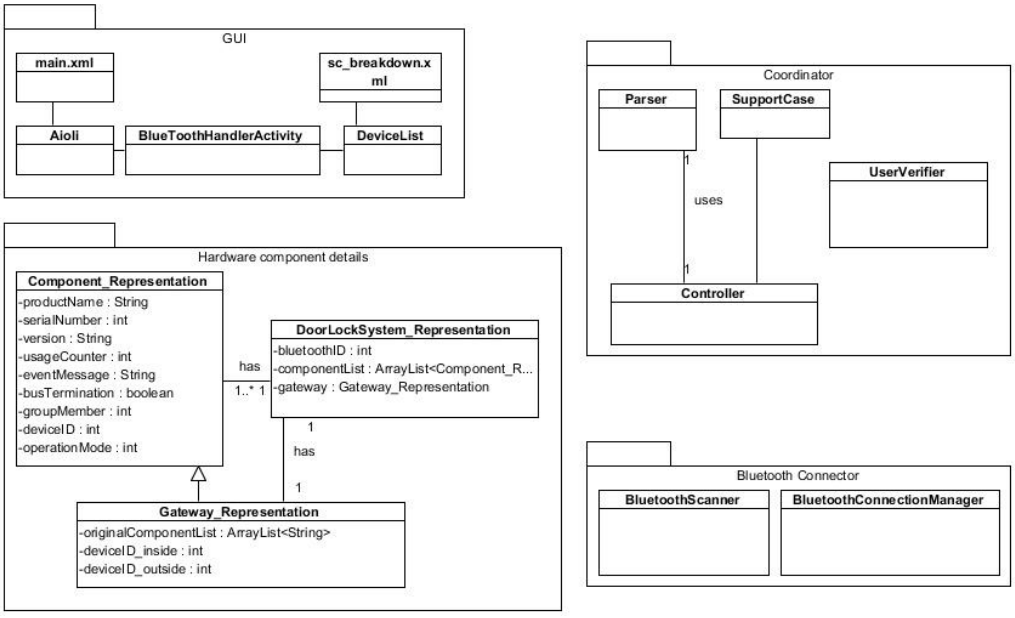


Figure 17: aSQA Components for the First and Second Evaluation

The following were the results of the first evaluation.

	GUI					Coordinator					HWComponentDetails					BluetoothConnector				
	t	c	h	i	f	t	c	h	i	f	t	c	h	i	f	t	c	h	i	f
Security																				
Authentication	5	1	1	4	4	5	1	1	5	5	1	1	5	1	1	2	1	4	1	1
Confidentiality	1	1	5	1	1	5	1	1	5	5	1	1	5	1	1	5	2	2	5	4
Portability																				
Change platform	4	2	3	3	2	3	4	5	4	1	3	3	5	4	1	2	1	4	1	1
Modifiability																				
Ease of modification	5	5	5	5	1	4	4	5	4	1	5	5	5	5	1	4	3	4	4	2
Usability																				
Recovery	5	1	1	4	4	5	1	1	5	5	4	4	5	4	1	1	1	5	1	1
Learnability	5	4	4	5	2	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1

t = target c = current h = health i = importance f = focus

Figure 18: aSQA Result — Evaluation One

Evaluation Two

For the second evaluation the architecture was the same as the one in the first evaluation, but the aSQA session was conducted together with three developers and one manager from ASSA ABLOY. Before the evaluation, they were provided with a presentation of all scenarios, the architecture for Aioli, and were given an explanation of how aSQA is used. Each participant received a paper copy of each view in the architecture as well as a paper copy of the quality attributes. This had the advantage that the participants did not have to memorize all information. Still, it is worth mentioning that it was the first time the participants saw the architecture and it was the first time for them to come in touch with the 4+1 documentation style. Thus, it was a lot of information to take in, in a very short amount of time. During the evaluation each person wrote down their own target and importance level of each quality scenario for each component. These were then discussed and agreed upon in the group. Due to the fact that we knew the state of the architecture in more detail, and there was a time restriction for this evaluation, it was decided that we should decide upon the current level for the architecture. Also, since the same components were used for evaluation one and two, the numbers for the current level were the same.

	GUI					Coordinator					HWComponentDetails					BluetoothConnector				
	t	c	h	i	f	t	c	h	i	f	t	c	h	i	f	t	c	h	i	f
Security																				
Authentication	5	1	1	5	5	5	1	1	5	5	1	1	5	1	1	1	1	5	1	1
Confidentiality	1	1	5	1	1	5	1	1	5	5	1	1	5	1	1	5	2	2	3	3
Portability																				
Change platform	2	2	5	2	1	3	4	5	3	1	3	3	5	3	1	2	1	4	2	1
Modifiability																				
Ease of modification	4	5	5	4	1	4	4	5	4	1	4	5	1	4	4	2	3	5	2	1
Usability																				
Recovery	4	1	2	4	4	3	1	3	3	2	1	4	5	1	1	1	1	5	1	1
Learnability	4	4	5	4	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1

t = target c = current h = health i = importance f = focus

Figure 19: aSQA Result — Evaluation Two

Since the architecture for the first two evaluations was the same, it was interesting to compare the outcome of both evaluations (see Figure 18 and Figure 19). “The lower the health is and the more important the quality attribute is for the component, the more focus there should be on raising the level of quality attribute for the component” [20]. Thus, as can be seen from the

first evaluation (Figure 18) the components that required attention were Gui, Coordinator and BluetoothConnector. According to the result, in the Gui component the authentication scenario as well as the recovery scenario required higher attention. In the Coordinator component the security quality attribute as well as the recovery scenario needed attention and in the BluetoothConnector component it was the confidentiality scenario that needed further focus. This was compared to the second evaluation with the developers at ASSA ABLOY, where the Gui and Coordinator components were the ones that required most attention. In the Gui component the authentication and recovery scenario were the ones requiring attention and in the Coordinator component is was the security quality attribute.

The aSQA technique and the quality attributes were discussed with the participants during the course of the evaluation. Through discussions with the participants it soon became clear that the portability attribute needed to shift focus. It was stated in the scenario description that the quality attribute focus lies on the code level. However, since it is impossible to reuse the Android code when porting the application to, for example, the iOS platform it was decided that it was more accurate if the focus laid on the design and architectural level.

The initiatives (step 6) that were defined after this evaluation had the purpose of improving the architecture to such an extent that the desired target level would be reached in the next evaluation. Therefore, one initiative was to incorporate the feedback from the participants of the second evaluation in the next iteration of the architecture. The second initiative was to go through all requirements and ensure that they all were supported by the architecture.

Evaluation Three

The results and feedback from the second evaluation session increased the knowledge about the evaluation method and pushed the work on the architecture forward. During that evaluation there was one part of the aSQA method that was confusing first for the participants and then also for us. During the evaluation of the architecture it became a challenge to separate the meaning of the target and importance level. As was understood from the research papers on aSQA the target level is used to define the desired quality level of each component at the end of the project, whereas the importance level is used to define how crucial it is to fulfill this quality level. The problem was that it was understood that the measuring scale for these levels should be the same. The reason why this was so confusing is that the target and importance level were almost always the same value. Therefore, we studied the papers on aSQA [20] [21] again, at which point a single sentence in one of the papers was discovered that stated that the scales for the levels should be different. The scale that was used in the first and second evaluation should only be used for the importance level, whereas for the target level and current level a scale could be used like the one outlined in the aSQA research paper [20]. For the third architecture evaluation this was changed. The following scale was used for the target and current level.

“1 = Unacceptable: Important stakeholders find the system unacceptable because of the value of the quality attribute in question.
3 = Acceptable: No relevant stakeholder finds the system unacceptable because of the value of the quality attribute in question.
5 = Excellent: All relevant stakeholders are highly satisfied by the value of the quality attribute in question.” [20]

The components that were used for this evaluation are the packages defined in the logical view (see section 4.5.1). It can be seen that the architecture has grown significantly, due to the fulfillment of the initiatives that were defined after the second evaluation. This has the effect that there were more components that needed to be evaluated in the third aSQA evaluation. As can be seen the graphical user interface was split up into three different components. The GUI component that was used for the first and second evaluation corresponds to the GUI_ServiceTechnician component in the third evaluation and the component which is called GUI in the third evaluation facilitates now the possibility for the user to log on to the Aioli application. The GUI_Superintendent component that can be seen in the third evaluation is used when a superintendent has logged in. It

facilitates the functionality for the breakdown scenario.

Another change that was made to the architecture after the first and second evaluation, is the GUI_Logic component. The reason for this was that it was decided that the logical work should be even more separated from the graphical user interface.

As more functionality was supported by the architecture the more it became obvious that the communication layer needed to be extended in order to facilitate more than just the Bluetooth communication. Therefore, the OnlineSupportConnector component was added in order to make the communication with the ASSA support possible which was desired for the online support scenario. The DatabaseConnector component was added to be able to establish a communication link between Aioli and the databases that are needed for the different scenarios.

When comparing the results shown in Figure 20 with the results from the second evaluation it can be seen that due to the splitting of the graphical user interface components the values of the target level for the different quality attributes have changed in the GUI_ServiceTechnician component. For example the target level of the authentication scenario is not of importance in this component but it is critical in the GUI component. Also, the target level of the Ease of Modification scenario was increased because more functionality was added to this component. Similar can be said about the Coordinator component; after some functionality was moved out of this component the target level automatically changed.

Due to the fact that the semantics of the portability quality attribute changed after the second evaluation the target level for the Change Platform scenario was increased for almost all components. It was regarded that it is important that the concept of the architecture is still applicable when a different smartphone platform is used.

It can also be seen that the target level for learnability in the GUI component is lower in the result of the second evaluation. The reason why it is higher in the first and third evaluation is because the researchers regarded this quality scenario as critical after the observation of the Hi-O door lock system. It became obvious then that the service technician should not have to figure out how the application works when he is standing on a ladder trying to install Hi-O.

	GUI					GUI_ServiceTechnician					GUI_Superintendent					Coordinator					HWComponentDetails				
	t	c	h	i	f	t	c	h	i	f	t	c	h	i	f	t	c	h	i	f	t	c	h	i	f
Security																									
Authentication	5	5	5	5	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1
Confidentiality	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1
Portability																									
Change platform	3	3	5	3	1	3	3	5	3	1	3	3	5	3	1	4	4	5	4	1	4	4	5	4	1
Modifiability																									
Ease of modification	3	4	5	4	1	5	5	5	4	1	5	5	5	4	1	4	4	5	5	1	4	4	5	5	1
Usability																									
Recovery	1	1	5	1	1	4	4	5	5	1	4	4	5	5	1	1	1	5	1	1	1	1	5	1	1
Learnability	5	5	5	5	1	5	5	5	5	1	5	5	5	5	1	1	1	5	1	1	1	1	5	1	1

	GUI_Logic					BluetoothConnector					OnlineSupportConnector					DatabaseConnector				
	t	c	h	i	f	t	c	h	i	f	t	c	h	i	f	t	c	h	i	f
Security																				
Authentication	5	3	3	5	3	1	1	5	1	1	1	1	5	1	1	4	2	3	4	3
Confidentiality	4	2	3	4	3	5	3	3	5	3	5	3	3	5	3	5	3	3	5	3
Portability																				
Change platform	4	4	5	4	1	2	2	5	2	1	2	2	5	2	1	2	2	5	2	1
Modifiability																				
Ease of modification	4	4	5	5	1	3	3	5	2	1	3	3	5	3	1	3	3	5	2	1
Usability																				
Recovery	4	4	5	4	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1
Learnability	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1	1	1	5	1	1

t = target c = current h = health i = importance f = focus

Figure 20: aSQA Result — Evaluation Three

Comparing the target levels with the current levels in Figure 20, it can be seen that all components reach the desired target level except for the security scenarios in the GUI_Logic, Bluetooth-

Connector, OnlineSupportConnector, and DatabaseConnector components. If at this point there would have been another iteration of the project the initiatives for that iteration would have been to improve the security quality attribute aspects of these components.

4.7 Prototype Design

This chapter outlines which requirements the Aioli prototype implements and explains the graphical user interface for how they are realized. Also the navigation through the user interface is described, followed by a discussion about which parts of the architecture are implemented in the prototype.

4.7.1 Implemented Requirements

For the prototype it was chosen to implement a selected set of requirements. Some of these requirements were however not fully implemented due to different reasons. The parts that were selected for implementation focused on the communication between the smartphone and the Hi-O door lock system. Therefore, the Service and Support Case Service Technician scenarios (see section 4.4.1) were selected. In order to implement these a few requirements from the General Requirements in Appendix A were also implemented. The implemented requirements are summarized in Table 3 where they are listed with a description of either fully implemented or partially implemented. The result of the implemented requirements and descriptions of what was included in the partially implemented requirements follows below.

Table 3: Implemented Requirements

Service (see Appendix A)	
REQ-1.1	Fully implemented
REQ-1.2	Fully implemented
REQ-1.6	Partially implemented
REQ-1.7	Partially implemented
REQ-1.9	Fully implemented
Support Case Service Technician (see Appendix A)	
REQ-3.1	Fully implemented
REQ-3.4	Fully implemented
REQ-3.7	Fully implemented
REQ-3.8	Fully implemented
REQ-3.9	Partially implemented
REQ-3.10	Partially implemented
REQ-3.11	Fully implemented
REQ-3.12	Fully implemented
REQ-3.13	Fully implemented
General Requirements (see Appendix A)	
REQ-6.3	Fully implemented
REQ-6.5	Fully implemented
REQ-6.6	Fully implemented

One implemented requirement from the General Requirements was 6.3, which states that the connection with Hi-O should be done via Bluetooth. The user interface screen for connecting to the Bluetooth adapter that gives access to Hi-O is shown in Figure 21. Pressing the Search for Devices button starts a search for nearby and discoverable Bluetooth devices, the progress bar in the top right corner lets the user know when this search is completed. Yet, it is possible to select a device before the search is completed since the devices are added to the list as soon as they are discovered. Selecting an entry in the list starts an attempt to connect to the device. Connecting to a device requires the correct password for that device and that the device supports serial communication.

The other requirements from the General Requirements category were 6.5 and 6.6 which state that it shall be possible for the user to store contact information in the settings of Aioli and that these shall be accessible from all other screens in the application. The settings screen is shown in Figure 22 and allows the user to store his email address, name, company and phone number. As seen in Figure 21 an option menu at the bottom of the screen is displayed, this is opened when the user presses the menu button on the Android phone. It allows the user to open the settings or the support case.

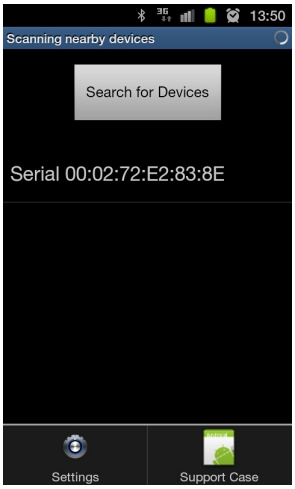


Figure 21: Search Device

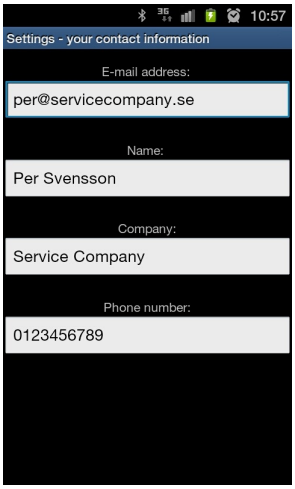


Figure 22: Settings

For the Service scenario five requirements from Appendix A were selected for implementation. However, for some of these not all functionality was implemented. The first requirement was 1.1 which says that it should be possible to read and present a list of all connected components showing their detailed information. This can be seen in Figure 23 where the components are displayed in an expandable list which allows the user to press the name of a listed component to expand it to see the detailed information. The requirements state that there are two different IDs, one for host and one for device for each component. In the prototype only the host ID is displayed since it is the same as the device ID. For the gateway however, both IDs are displayed since they are different. The second and third requirements implemented in the prototype are 1.2 and 1.6 which state that it should be possible to present and set the operation mode of Hi-O. Yet, only the Installation and Configuration modes, which are not encrypted, are supported in the prototype. The possibility of changing the operation mode is included in the Additional information screen in Figure 24, where it displays that the current mode is Installation. The fourth chosen requirement is 1.7, which states that it should be possible to run a diagnostic test. This requirement was partially implemented and included one example of what could be included in the diagnostic test. This is shown in the Additional information screen as the door health check which allows the user to see the output of what happens when the physical door opening button is pressed. The fifth requirement is 1.9 which states that it shall be possible to create a support case. This can be done by pressing the Create Support Case button in Figure 24.

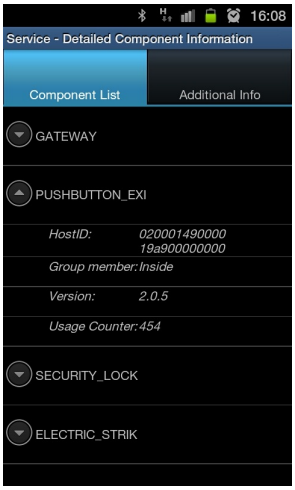


Figure 23: Service — Component list

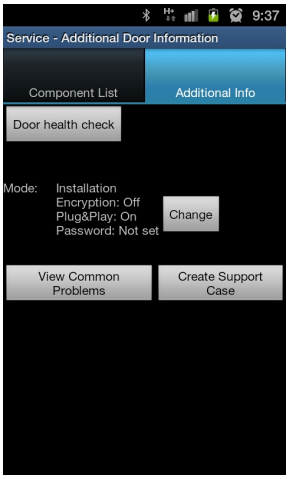


Figure 24: Service — Additional Info

For the Support Case Service Technician scenario nine requirements were included in the prototype and also here not all functionality was implemented for all requirements. The first requirement is 3.1 which states that it should be possible to show the component list from the Service scenario. This is shown in Figure 25. By scrolling down in the top part of the support case screen the user would see the bottom of the support case shown in Figure 26. The second and third requirement are fulfilled by the text box named Additional Textual Information in Figure 26 since it fulfills both requirement 3.4 which says that it should be possible to add the address of the door lock system and requirement 3.11 which states that it should be possible to add comments. The text in this text box is persistent if the user accidentally presses the back button the text is still there when the screen is opened again but it is removed if the user closes the application. The next four requirements express the need to add pictures and videos (requirement 3.7 – 3.10). Although in the prototype it is only supported to add one video and one picture. The user can according to requirement 3.12 add contact information which is the same as the information in the settings seen in Figure 26. The user can then, as stated in requirement 3.13 send all the information described above as an email to the ASSA support by pressing the Send Support Case button.

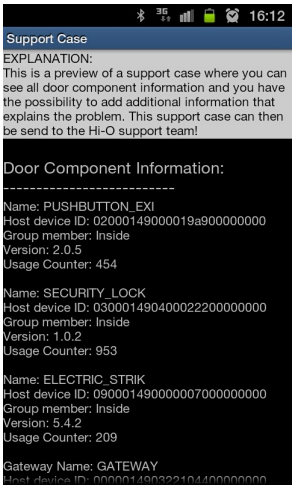


Figure 25: Support Case — Top

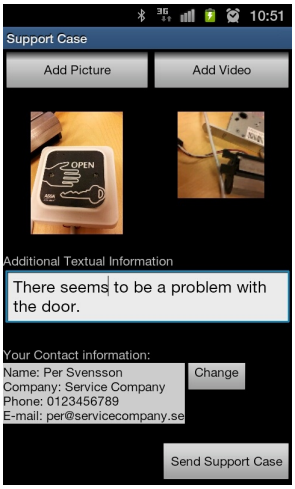


Figure 26: Support Case — Bottom

4.7.2 GUI Navigation

The flow of the user interface is shown in Figure 27. The figure only shows how to step forward in the Aioli prototype since the back button on the Android phones allows the users to go backwards through the screens. This is possible even if for example a progress bar is locking the current screen while waiting for something to finish. From the Main screen it is possible to open the option bar shown at the bottom, this takes the user to the Settings screen where the contact information of the user can be changed. From the Main screen it is also possible to press the Service button which opens the Search Device screen. The option bar in the bottom of the Search Device screen is, as already stated, available for all other screens as well. From that option bar it is possible to go either to the Settings screen or to the Support Case screen. When a device is selected from the list the Service Component List screen is opened. While Aioli communicates with Hi-O to retrieve all information about the components, a progress bar in the middle of the screen locks all functionality. When the communication is completed the expandable list with all component information is shown. Clicking Additional Info will open a tab where it is possible to do a health check, change operation mode or create a support case. The View Common Problems button does not have any functionality in the prototype and is only there to show where this functionality could be added. When the health check button is pressed a pop-up dialog is shown asking the user to press the physical door opening button once. Aioli then reads the information sent from Hi-O and shows the cycles the components ran. For example the locks that were opened and closed and if any delays occurred it will also show the attempts of opening and closing the locks. Pressing the Change button opens a pop-up dialog with radio buttons, allowing the user to switch operation mode in Hi-O. The Support Case screen adds information from the service and displays it, when the user scrolls down there are buttons to add a video and a picture which both open either the camera application or the gallery. The user can review the contact information and change it if needed by pressing the Change button. Finally, the user has the option to open a third party email application. As seen in Figure 27 an email is opened in the Gmail [29] application with the text included and the video and the picture attached.

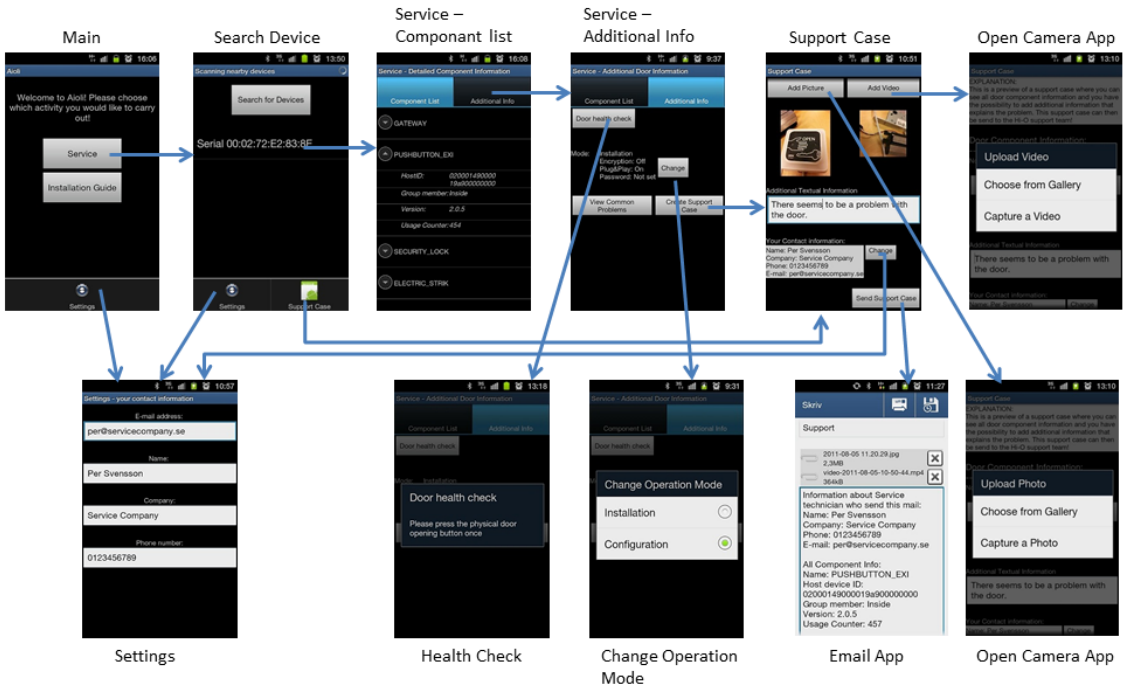


Figure 27: GUI Navigation

4.7.3 Implemented Architecture

To implement the selected requirements in the prototype not all parts of the architecture were needed. An overview of which packages from the logical view the prototype implements or makes use of is shown in Figure 28. The parts included in the prototype are marked in blue. These are the GUI_ServiceTechnician package, the Coordinator, the HardwareComponentDetails, the BluetoothConnector, the Third Party Programs, the Android platform and the Hi-O door lock system. The packages and classes in white were not implemented or used in the prototype. At the time the prototype was developed the ThreadManager class in the Coordinator package did not exist. Instead the Controller was connected to the Parser and the Commands classes which had the responsibilities of the ThreadManager class. Thus, the Controller also handled the communication with the BluetoothConnector package. In the prototype the GUI_Logic package was not used, instead each implemented GUI class handled its own logic. This design decision was made at a late stage of the project and therefore it was not possible to make this change to the prototype.

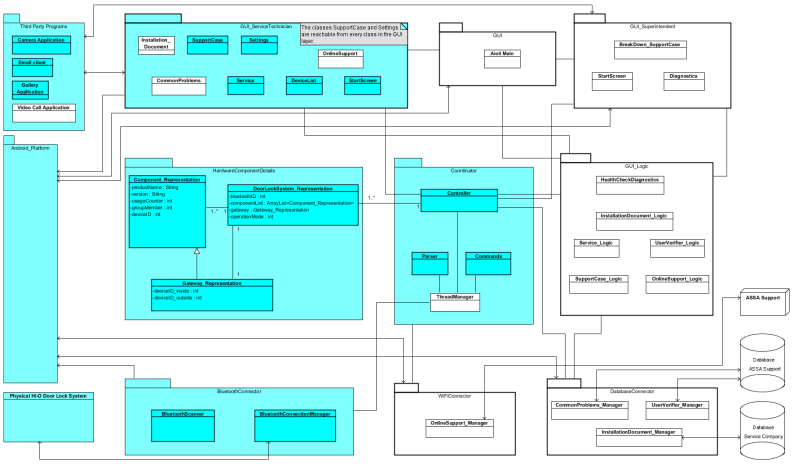


Figure 28: Implemented parts of the logical view

In the process view (see section 4.5.3) the BluetoothConnection in Figure 12 shows how the connection is set up and managed using four threads (see Figure 12). Three of the threads were used in the prototype: the Main thread, the BluetoothConnect thread and the BluetoothConnected thread. Instead of the CommunicationManager that is shown in the architectural description the Main thread handles that functionality in the prototype. This is due to the fact, that this part of the prototype was developed before the architecture was fully designed and there was no time to make those changes to the prototype.

The prototype was divided into the same layers as in the development view in section 4.5.2. From the execution environment for Aioli in the physical view (see section 4.5.4) the prototype uses the camera application, the gallery, and the email client. Furthermore, it also communicated with Hi-O via a Bluetooth adapter with the addition that a development board was added between the Bluetooth adapter and the Hi-O system as described in section 3. The databases and the back-end at the ASSA support were not included in the prototype.

4.8 Prototype Evaluation

In order to evaluate the design of the Aioli application and see if it was understandable we conducted three prototype evaluations. The first two evaluations were conducted with software developers at Diadrom. Separately, they were asked to try to follow the flow of the application and see if it is understandable. Both participants were experienced with the Android platform and had therefore no problems testing the whole application. The feedback we received from these evaluations concerned the fact that there was functionality in the prototype that was not implemented.

For example when the button Add Video was pressed a notification saying that this feature is not yet implemented was triggered. Not being able to test functionality that appears to be available was regarded as a frustrating experience. Since we wanted to avoid this feeling when conducting an end user evaluation, we took the time to implement requirements such as being able to add a video to the support case and send the created support case via email.

The third evaluation was carried out with a service technician from Sesam Lås & Larm. He was excited about the idea to be able to use a hand-held tool when working with the Hi-O system. Stating that todays systems are much harder to check for correctness compared to older systems, since it was a lot easer in older systems to measure the voltage in order to see if everything was working according to specification. He liked the fact that the application read the information from the door after the physical button was pushed. This, he said, is needed to see that the components work correctly.

The service technician also liked the fact that the Aioli application shows all components that are installed in the Hi-O door lock system by name. This information gives a clear indication about the components that are up and running and those that are not. About the details that are shown for every component the participant stated that he liked the component version and the usage counter since it is useful information. However, he stated that in order to work with Hi-O he did not need to know the host or device ID. Yet, this information needs to be retrieved in case a support case is created as these IDs should be available to the ASSA support. One requirement that was expressed during this evaluation was the ability to reset a component. Locked components need in todays situation be sent to the ASSA support in order to reset them. The participant expressed that this process would be improved if a component could be reset via the Aioli application. However, this requirement needs further research before it can be decided if this is possible.

Further feedback we received concerned the documentation of an installation or service of the Hi-O system. For example, the participant stated that it would be great if all information that has been retrieved would be documented so that a history can be kept by the service company and that this documentation can serve as proof for the client that the door look system works as it should. This gave direct support for the installation scenario in section 4.4.1. Also, this feedback suggests an extension to the Service and Support Case scenarios which both should offer the possibility of documentation. In the Service scenario the work that has been carried out on the Hi-O system shall be documented, and in the Support Case scenario, the problems that have occurred shall be documented.

5 Discussion

This chapter deals with the discussion and evaluation of the results gathered during the project. It also includes a discussion on the related work to this project and a discussion about the future work that needs to be carried out in order to have a fully functional Aioli application.

5.1 Evaluation of Result

In this section the experienced effects of using the chosen research method, the results from the requirement elicitation, architecture design, the evaluation and prototype are discussed.

5.1.1 Research Methodology

It was experienced that Design Research worked very well with the iterative architecture design and prototype implementation. Especially, the aspect of developing a knowledge base during the course of the project was useful since new information was gained in every step of the research process. Also, the fact that the iterative design of the chosen research method supported the use of a, to us, unknown architectural evaluation method was found beneficial since we had the chance to conduct several evaluations during the course of the project. Moreover, the fact that the steps in the Design Research method are not sequential but one can go back and forth between them was experienced as a great advantage as more knowledge was gained. Hence, the Design Research method proved to be a suitable choice for this research project.

An alternative to Design Research would have been to conduct a Case Study, which is an in depth study of a particular issue. The reason why we chose the Design Research method instead, is that the process steps it consists of did fit the research project very well, especially because one of the goals of this project was to develop a prototype as a proof-of-concept for the architecture of Aioli. According to Winston Tellis [52] the Case Study research method consists of four steps; Design the case study, Conduct the case study, Analyze the case study, and finally Develop the conclusions, recommendations and implications. These steps were considered too limited for the research project since they only take into consideration the investigation of a problem but do not plan for a solution. The Design Research method was perceived as similar to an iterative development process [51] and was thus preferred.

The decision that different types of data collection methods should be used was mainly because not all participants could be met with in person. Also, the fact that the data was collected through many different approaches (see section 2.2) widened the knowledge base. It was beneficial to observe a Hi-O door lock system installation because we got a better understanding of the service technician's work and the problems that can arise with Hi-O and we were therefore able to make better decisions about the Aioli system. This observation in combination with a variety of interviews was a vital asset to the project's progress.

The collected data was analyzed with the help of Nvivo 9 (see section 4.2), which enabled us to structure the data and get an overview of the most important areas. There are similar tools available, but Nvivo was chosen because it is widely used, easy to learn, and free to download for students at the University of Gothenburg.

5.1.2 Requirements Specification

It is relevant to mention requirements and scenarios that had to be removed from the specification for different reasons. Since some of the requirements are regarded valuable for a sufficient and complete Hi-O support application, we give suggestions on how these removed requirements still could be achieved.

During the data collection phase many desired functional requirements for the Aioli application were suggested. These were included in the early versions of the requirements specification until it was discovered that the Hi-O system does not support them.

One such requirement was that it should be possible to show a list of all missing components. Today, it is not possible to retrieve this information from Hi-O since this information is not automatically stored anywhere. There are several possible solutions to this problem. First of all, this information could be retrieved if it was possible to add some logic to the gateway component so that it can store a list of all components that were present when the system was installed. If that is achieved, then every time Aioli retrieves component information one can compare those retrieved components with the components listed in the gateway and see if all are present. A second solution would be to save this list in a central database instead of the gateway component. A third possible solution, but one that requires more effort to achieve, is to have specific door types. This means that there should be several types of Hi-O doors each specifying which components need to be installed. Then the service technician can just check what type of Hi-O door lock system he is working with and instantly know which components need to be up and running.

Another requirement that could not be met was that for bus termination, where one shall be able to see in the Aioli application which components terminates the CAN bus. As was found out, in today's situation, this needs to be physically carried out on the components. Thus, the only way it can be solved with Aioli is to add a check in the installation guide (installation scenario) in order to remind the service technician to carry out the termination.

Two requirements that seemed not possible to be achieved in the current state of the Hi-O door lock system, were the ability to retrieve any error codes that Hi-O generates and the ability to set a usage limit for each component. This limit determines how many cycles a component can be used before it needs to be serviced or even replaced. Testing this with the hardware that was available to us, we were not able to retrieve this information since Hi-O does not store this. As it turned out at a late stage of the project, this functionality is available in a different combination of Hi-O components which we did not have access to. Thus these requirements need to be taken into consideration when the Aioli application is further developed.

Furthermore, it is worth mentioning that during the course of the second iteration it became clear through the feedback from the contact persons at SSSA ABLOY that one of the scenarios should be removed. This scenario was the Maintenance scenario that was originally part of the functional requirements. In this scenario it was thought that service technicians as well as superintendents would be able to perform a check on all Hi-O door lock systems that are in reach and to which the Aioli application has access to. It was thought that during this check Aioli would connect sequentially to all door lock systems and retrieve the information. Aioli would also check the usage counter for each component in order to determine if they would require service or even replacement in the near future. All information for all reachable door lock systems would then be stored in a log that is sent to the service company responsible for the security of these door lock systems in the building. The purpose of this scenario was to find components that need replacement, hence it should detect a potential door malfunction before the door breaks down. Yet, the feedback that we received was that this scenario was considered as unfitting since it would be unlikely that the user would stand still and wait until Aioli has connected to all Hi-O door lock systems. If the user moves while the maintenance check is in progress the distance to the doors changes and Aioli might loose the connection. Also, for Bluetooth it is not always the case that all door lock systems will be in reach for Aioli, thus it is very easy to just run a check on some of the doors. Furthermore, there are two possibilities for how to use the Bluetooth adapter. Either, it will be installed in every Hi-O door lock system or the user will have his own Bluetooth adapter which can be plugged into every Hi-O door lock system he needs to work with. Both alternatives have their advantages and disadvantages. If there is a permanent Bluetooth adapter installed, the security risk is higher. If the adapter gets plugged in when needed the security risk is lower, but the maintenance scenario is not possible.

5.1.3 Architecture Design

For the documentation of the architecture the 4+1 view model was chosen. We experienced it as a structured way to document the architecture which contributed to that more aspects of the architecture were explored. One example is the process view for the Online Support scenario

where the need for the CommunicationManager thread was discovered. Moreover, the fact that the different views complement each other makes the architecture applicable to more stakeholders, which is useful if Aioli will be developed further.

The application is designed to have two different user types, as described in section 4.4.1, although more can easily be added. Having one application where the users have to be authorized instead of one application for the superintendent and one for the service technician has the benefit that only one application would have to be maintained. The drawback with this solution is that if the user does not have access to WiFi he cannot log in to and use Aioli. One solution to this could be that the user logs in at a place where Wifi is available and he is then able to continue to use Aioli when the Wifi connection is lost.

The architecture was designed for the Android platform but with the aim to make it applicable for other platforms as well. One step towards achieving this was to use a layered architecture as seen in the development view (see section 4.5.2). The layered architecture was chosen since it allows for a separation of the graphical user interface, the logic and the communication with other systems and hardware. Each layer was divided into several packages with different responsibilities, which make the architecture easier to understand and modify.

One of the strengths with a smartphone is that it is possible to use other applications installed on it. This, not only makes the development of the application faster, but also gives the user more freedom. For example when opening the camera application Aioli does not need to know which camera application on the smartphone is used, so the user can chose the one he prefers. The only requirement for this is that the camera application allows other applications to use it.

For evaluating the architecture aSQA was not the only considered evaluation technique. For example the Architecture Tradeoff Analysis Method [50], also known as ATAM, was taken into consideration. This method also focuses on evaluating a software architecture relative to the chosen quality attributes [50]. The reason why ATAM was discarded was due to the fact that one evaluation "typically takes three to four days and gathers together a trained evaluation team, architects, and representatives of the architecture's various stakeholder" [50]. This was considered to be too complex and costly for both us as well as the stakeholders. The aSQA method had the advantage that we could carry it out in only a few hours with fewer people and with the same goal as ATAM.

Although, we made a mistake with the target and importance level in the first and second architecture evaluation, this mistake did not have a negative impact on the evaluation of the architecture. Even though only one scale was used in the second evaluation (see section 4.6), we still received answers on what the participants regarded as the target level for each component. It was also discussed how important it was that these targets are fulfilled. In the second aSQA evaluation, also another way of how to interpret the target and importance level was discussed. In this discussion the target level was still seen as the desired final quality level for all components, but the importance level was seen as a measurement for how important it is to fulfill this quality level in the next iteration. This way a priority for all components in the next iteration phase would have been established and we think that this would be a good way of using the aSQA method.

While using the aSQA method we experienced that such a systematic evaluation of the architecture not only resulted in the quantitative measures but also generated a lot of qualitative information through all the discussions it started. These discussions led to the discovery of errors that most likely would not have been discovered without the evaluations. During the second evaluation we received feedback from the participants saying that if the quality scenarios contain errors or are not precise, it becomes difficult to set a value for the components. One such example was the portability attribute, which should have been designed on architectural level instead of the code level. Also some response measurements in the quality scenarios were too harsh and most likely impossible to achieve. These were changed as a result of the evaluations.

Furthermore, it became obvious that not all quality attribute scenarios are applicable to all components of the architecture. The participants in the second evaluation suggested a solution, to set "not applicable" on those components where a specific quality attribute scenario cannot be applied. This was suggested in order to speed up the process in case one has a lot of components

and quality scenarios.

For the evaluations the selected components were the packages from the logical view. This lead to that not all views received the same amount of attention as the logical view. Even though only one view from the 4+1 was used for the evaluation it still was valuable for the project. Through the discussions started during the aSQA evaluation we discovered parts to improve not only in the logical view. For example, the problems with the quality attributes were discovered (as described above), in the development view it was noticed that the databases were accidentally connected to the wrong layer and this could be corrected.

When the participants were asked if they could imagine using the aSQA evaluation method on their own projects at ASSA ABLOY, the response was positive and curiosity towards trying it out was expressed. Yet, it was perceived as this would be a greater effort as such a detailed architectural documentation needs to be created first. Furthermore, it was expressed by the participants that it was a great experience to try this method in practice because there is always a considerable difference between reading the papers and trying the method in practice.

During the group interview the participants thought that applications like Aioli would be applicable to other intelligent door lock systems. As our proof-of-concept prototype shows, using a layered architecture like the one described in this report (see section 4.5) could be applied where a smartphone needs to communicate with advanced embedded equipment. The architecture also allows to switch the communication medium from Bluetooth to any other preferred medium.

5.1.4 Prototype Design

The prototype was developed as a proof-of-concept in order test the architecture. We did this through implementing the prototype according to the architectural specification. The purpose of the prototype was also to see if it was possible to make a smartphone application communicate with the Hi-O door lock system. The reason we chose to develop the prototype in an environment where a development board was included was to speed up the development process.

The choice of developing the prototype on an Android platform was made because it is one of the leading platforms for smartphones [19] [28] and because of our curiosity about Android development.

Apart from Bluetooth also Near Field Communication [37] was considered as an option for communication with Hi-O but was discarded since the transmission is within such short range (up to four centimeters [38]) . It was considered too unpractical to hold the smartphone within the range while working with the door lock system. A Bluetooth device of class 2 provides a distance of 10 meters [18] and is a more suitable range when using Aioli to communicate with the door lock system. The scenarios we selected to implement in the prototype were, as stated in section 4.7, the Service scenario and the Support Case Service Technician scenario (the scenarios are described in section 4.4.1). These scenarios were chosen mainly because they deal more with the communication with Hi-O than the other scenarios do. The Breakdown Support Case has similar functionality compared to the Support Case Service Technician scenario, except that it displays less information to the user. For the prototype we deemed it redundant to implement the breakdown scenario as well. The Installation scenario was not chosen since it does not include more communication with the Hi-O system than the Service scenario does and would not test that part of the architecture any further, although it would make use of other parts of the architecture. Also, to implement the Installation scenario more research about what to include in the checklists is needed. The Online Support scenario would have been useful to implement, however with the time constraints on the project it was not possible.

The main reason why some of the selected requirements were only partially implemented in the prototype was because we did not see it as necessary for testing the architecture. However, the requirement including to set the operation mode to Running was discarded since it was not feasible to implement the functionality for unlocking the Hi-O systems encryption mode in the prototype. The complete diagnostic functionality would have been interesting to include in the prototype but

it would require a more extensive research of what should be included for the different door lock setups. This is discussed further in section 5.2.

When comparing the final architecture to the implemented prototype two differences become apparent. The first difference is that the ThreadManager class in the logical view and the CommunicationManager thread in the process view (see section 4.5.3) are not included. The second difference is that the logic for the graphical user interface classes is placed in the GUI_Logic package in the architecture but in the prototype each user interface class handles its own logic directly. This is because we stopped developing the prototype earlier than the architecture and additions were made to the architecture after the prototype was considered finished. These additions were added as a result of the first and second aSQA evaluations (see section 4.6). This should not have a big impact on the proof-of-concept evaluation since it is only a matter of splitting up the functionality into different classes and threads as described in section 4.7.

Our experience of Android development is that it is quick to set up and start building an application. The learning curve was quite low since both of us had previous experience with the Java programming language. The parts that caused problems regarding the development of the prototype was that the emulator [6] did not support Bluetooth connections. Therefore, almost all development had to be carried out on the smartphones (see section 3).

Another issue was that although the API is mature, we encountered some problems with it. The most concerning problem for the prototype was that the API for serial Bluetooth communication for Android does not seem to be fully supported. Although we have not been able to confirm this problem with a reliable source we have experienced varying results when testing our application with the two phones mentioned in section 3. Bluetooth often has to be restarted in order for Aioli to connect to the Bluetooth adapter connected to the Hi-O system. Also, this is a commonly discussed problem at various forums.

The second problem with the API was with the expandable list which displays the component details in Service scenario (see Figure 23, section 4.7). Before the requirement of setting a usage limit (see section 5.1.2) was removed, it was planned to let one of the entries in the expandable list have a different layout. One class and one abstract class for ListAdapters [8][7] where tried, which should allow changing the layout of one entry of the list. However, the entry receiving the different layout changed each time the list was redrawn (for example when opening and closing the list). Although, the feature of drawing one list entry with a different style was not included in the prototype it was discovered that it can be solved by including all elements of both the regular and the additional layout and then set elements as invisible depending on the entry.

5.2 Future Work

If it is decided that the Aioli application shall be put into daily use, there are a few further investigations that need to be conducted in order to get a fully functional application. One area which will require further research is the diagnostic part of the application. One needs to investigate which steps the service technician shall perform in order to retrieve all the needed information from the Hi-O door lock system. Further investigations are required for the smartphone communication with the external databases, how this can be practically solved, also it needs to be researched how the back-end for the Online Support scenario can be implemented. Concerning the databases it is assumed that ASSA ABLOY and each service company using Aioli need to maintain their own database, if this should not be possible alternative solutions need to be researched.

Regarding the communication between Aioli and Hi-O, if Bluetooth will be used it needs to be investigated how to best integrate the Bluetooth device with the Hi-O system. One solution could be to have a device which translates CAN to serial as the one used for the prototype. Another solution could be to use a Bluetooth adapter which supports CAN. Then the Aioli application would directly communicate with the components in the Hi-O system. However, this also implies that more research needs to be conducted for how to handle the communication on the Aioli

application side since CAN communication does not seem to be supported in the Android API at the current time.

If serial Bluetooth would be used, as it was for the prototype in this project, further research about the support for serial Bluetooth on different Android phones will be required.

Moreover, further research is required for how to best secure the Bluetooth communication between the Aioli application and the Hi-O door lock system. Regardless of what solution will be chosen for the communication, further detailed research needs to be conducted about how to solve the message encryption on the Hi-O side.

In case every Hi-O door lock system shall have its own Bluetooth adapter, other details such as how the user will know the name and password of the Bluetooth adapter needs to be investigated. This information needs to be distributed to the end users so that they will be able to use the Aioli application.

Also, in order to have a fully functional application which fulfills the requirements that needed to be removed, it is required to investigate the solutions that were put forward in section 5.1.2 which suggest how these requirements could be achieved.

Finally, since it was chosen to send the support cases via email it needs to be decided how to hide the sensitive information from the superintendent user group. One suggestion would be to attach an encrypted file to the email that contains the sensitive information.

5.3 Related Work

Smartphone platforms seem to open the possibility to explore technology in a new light. Not only can working processes be enhanced via a smartphone, as just shown with the Hi-O door lock system, but even the possibility of improving the safety of riding a motorcycle with help of a smartphone has been researched [35]. As shown in the study conducted by Manzoni et. al the idea of a bidirectional audio interaction between the vehicle and the driver was investigated. According to the research “the driver-to-vehicle system is based on an audio interaction located at helmet level” [35], the smartphone facilitates the communication between the helmet (which is equipped with Bluetooth) and the Vehicle Control Unit (VCU) on the motorcycle. The idea is to communicate with the motorcycle in order to give commands and receive information from the motorcycle.

Also in the health care system research projects that involve smartphones has appeared. For example a “breathing bio-feedback system based on the Smartphone and Bluetooth technology” [58] has been developed in a research project. The researchers developed a smartphone application that receives data from the respiration and it generates signals in the audiovisual format to the user. They came to the conclusion that “smartphone based biofeedback is a promising field of combining the biofeedback technique with the convenience of the mobile phone, people can perform desired biofeedback at any time and any where” [58].

Another research example in the health care system is the ability to facilitate “low-cost continuous health monitoring based on commercial-off-the-shelf wireless wearable biosensors” [33]. The monitored data, which can be in the form of heart rate, breathing rate, oxygen saturation, and estimated obstructive sleep apnea, is sent from the biosensors to the smartphone application via Bluetooth or WiFi. The application has been developed in such a way that different types of sensors can be used. The result of this research was a complete system that worked to full satisfaction.

One point that we tried to stress in this report was the importance of a sophisticated and improved support for the embedded hardware. One such solution we presented was the remote online support. This phenomena has also been addressed by Kraut et al. [43]. In their research they “investigate the design and value of collaborative, mobile computer systems to support field repair and maintenance of mechanical devices“ [43]. Their findings show that “collaboration with a

remote expert substantially improves a less skilled person's ability to perform maintenance and repair tasks" [43].

Despite extensive research on smartphone communication with advanced door lock systems in terms of diagnostics no related research could be found. This confirms that the investigated topic of the this research report is still new and exciting.

6 Conclusion

The results of our research project show that it is possible to facilitate support of an intelligent door lock system via a smartphone application.

Implementing the Aioli prototype according to the architectural description presented in this paper shows that it is possible to use the architecture to facilitate support of the Hi-O system. This architecture was designed beyond the functionality in the prototype and was evaluated using the aSQA method. This method contributed to a higher quality level of the architecture within the chosen quality areas. The fact that this method leads to a quantitative measure of the architectural quality gave us a clear indication of which areas needed improvement.

The use of the iterative Design Research method with a qualitative approach allowed for an extensive data collection using several methods which led us to a rigor research result. Through the data analysis a set of functional and non-functional requirements was created which gave rise to several functional scenarios and quality attributes.

According to the prototype evaluation with an end user it can be said that Aioli will be a great improvement to the service technicians daily work with Hi-O since no such tool does yet exists which is easy to bring to the working site and supports the installation and service of Hi-O in a way that the Aioli application does.

Moreover, our informants expressed that support applications like Aioli are applicable to other intelligent door lock systems both today and in the future.

Acknowledgments

We would like to thank everyone at Diadrom and ASSA ABLOY who helped to make this research project possible. At ASSA ABLOY we would especially like to thank Jakob Stenfeldt and Niklas Knutsson for their expertise with Hi-O and the hardware that they provided us with. Furthermore, we would like to say that we appreciated the opportunity given by Sesam Lås och Larm to let us observe an installation of Hi-O and for the feedback given during the prototype evaluation. We are grateful to everyone who participated in the interviews, observation and workshop of this project. Last but not least we would like to thank our industrial supervisor Henrik Fagrell at Diadrom and our academic supervisor Ulrik Eklund for their guidance and valuable feedback during this research project.

References

[1] Android Developers [2011-07-04], 'Android', http://developer.android.com/guide/basics/what-is-android.html. 2

[2] Android Developers [2011-07-05], 'Android User Interface Guidelines', http://developer.android.com/guide/practices/ui_guidelines/index.html. 53

[3] Android Developers [2011-07-22], 'Processes and Threads', http://developer.android.com/guide/topics/fundamentals/processes-and-threads.html. 21

[4] Android Developers [2011-08-02], 'Android 2.2 Platform', http://developer.android.com/sdk/android-2.2.html. 7

[5] Android Developers [2011-08-16], 'Android 2.3.3 Platform', http://developer.android.com/sdk/android-2.3.3.html. 7

[6] Android Developers [2011-08-17], 'Android Emulator', http://developer.android.com/guide/developing/tools/emulator.html. 41

[7] Android Developers [2011-08-18*a*], 'BaseExpandableListAdapter', http://developer.android.com/reference/android/widget/BaseExpandableListAdapter.html. 41

[8] Android Developers [2011-08-18*b*], 'SimpleExpandableListAdapter', http://developer.android.com/reference/android/widget/SimpleExpandableListAdapter.html. 41

[9] Apple Developer [2011-08-10], 'iOS Dev Center', http://developer.apple.com/devcenter/ios/index.action. 15

[10] ASSA ABLOY [2011-03-22*a*], 'About ASSA ABLOY', http://www.assaabloy.com/en/com/About-ASSA-ABLOY/. 1

[11] ASSA ABLOY [2011-03-22*b*], 'ASSA ABLOY Mobile Keys', http://www.assaabloy.com/en/com/Products/ASSA-ABLOY-Mobile-Keys/. 1

[12] ASSA ABLOY [2011-07-04], 'Hi-O (Highly Intelligent Opening)', http://www.hi-o.se. 1

[13] ASSA ABLOY [2011-08-01*a*], 'Hi-O Technical Specifications', http://mpc.lockwood.production.assaabloy.com/FileExplorer/fetchfile.aspx?id=2300&dl=true. 6

[14] ASSA ABLOY [2011-08-01*b*], 'Hi-O Technology Catalogue', http://mpc.lockwood.production.assaabloy.com/FileExplorer/fetchfile.aspx?id=2301&dl=true. 6

[15] ASSA ABLOY [2011-08-22], 'Hi-O Technology', http://www.hi-o.se/Hio/Templates/PageFaq___256.aspx. 6

[16] Bass, L., Clements, P. and Kazman, R. [2007], *Software architecture in practice*, 2 edn, Addison-Wesley Longman Publishing Co., Inc. 14, 26

[17] Bies, L. [2011-08-02], 'RS232 Specifications and standard', http://www.lammertbies.nl/comm/info/RS-232_specs.html. 6

[18] Bluetooth SIG, Inc [2011-08-09], 'Bluetooth Basics', http://www.bluetooth.com/Pages/Basics.aspx. 40

[19] Canalys [2011-08-04], 'Google's Android becomes the world's leading smart phone platform', http://www.canalys.com/newsroom/google%E2%80%99s-android-becomes-world%E2%80%99s-leading-smart-phone-platform. 40

[20] Christensen, H., Hansen, K. and Lindstrøm, B. [2010*a*], Lightweight and Continuous Architectural Software Quality Assurance Using the aSQA Technique, *in* M. Babar and I. Gorton, eds, 'Software Architecture', Vol. 6285 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 118–132. 10.1007/978-3-642-15114-9_11.
URL: *http://dx.doi.org/10.1007/978-3-642-15114-9_11* 26, 27, 28, 29

[21] Christensen, H., Hansen, K. and Lindstrøm, B. [June 2010*b*], 'Software Architecture at Work Technical Report aSQA: Architectural Software Quality Assurance'. 26, 29

[22] CiA [2011-08-02], 'CAN protocol specification', http://www.can-cia.de/index.php?id=164. 1

[23] Creswell, J. W. [2003*a*], *Research design: Qualitative, quantitative, and mixed method approaches*, 2 edn, Sage Publications, Inc., chapter 10: Qualitative Procedures. 3

[24] Creswell, J. W. [2003*b*], *Research design: Qualitative, quantitative, and mixed method approaches*, 2 edn, Sage Publications, Inc., pp. 181–183. 4

[25] Diadrom [2011-08-15], 'Welcome to Diadrom', http://www.diadrom.se/start.htm#. 1

[26] Dictionary.com [2011-03-22], 'Collins English Dictionary - Complete & Unabridged 10th Edition', http://dictionary.reference.com/browse/smartphone. 1

[27] Eikon srl [2011-08-02], 'BLUEMAX05 Bluetooth serial RS232', http://www.eikonsite.it/products/rs232+bluetooth+adapter/BLUEMAX05.aspx. 7

[28] Gartner Newsroom [2011-08-04], 'Gartner Says Worldwide Mobile Device Sales to End Users Reached 1.6 Billion Units in 2010', http://www.gartner.com/it/page.jsp?id=1543014. 40

[29] Google Inc. [2011-08-08], 'Gmail', https://market.android.com/details?id=com.google.android.gm&hl=en. 34

[30] Google Inc. [2011-08-19], 'Android Market', https://market.android.com/?hl=en. 19

[31] Java [2011-08-24], 'The Java Programming Language', http://groups.engin.umd.umich.edu/CIS/course.des/cis400/java/java.html. 16

[32] Kruchten, P. [1995], 'Architectural Blueprint - The "4+1" View Model of Software Architecture', *Paper published in IEEE Software 12 (6)* . 15, 16, 19, 24, 25

[33] Lam, S., Wong, K. L., Wong, K. O., Wong, W. and Mow, W. H. [2009], A smartphone-centric platform for personal health monitoring using wireless wearable biosensors, *in* 'Information, Communications and Signal Processing, 2009. ICICS 2009. 7th International Conference on', pp. 1–7. 42

[34] LG Electronics [2011-08-02], 'LG Optimus 2X P990', http://www.lg.com/uk/mobile-phones/all-lg-phones/LG-android-mobile-phone-P990.jsp. 7

[35] Manzoni, V., Corti, A., Spelta, C. and Savaresi, S. [2010], A driver-to-infrastructure interaction system for motorcycles based on smartphone, *in* 'Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on', pp. 1442–1447. 42

[36] Microchip [2011-08-09], 'A CAN Physical Layer Discussion', http://ww1.microchip.com/downloads/en/AppNotes/00228a.pdf. 6

[37] NFC Forum [2011-08-04*a*], 'About NFC', http://www.nfc-forum.org/aboutnfc/. 40

[38] NFC Forum [2011-08-04*b*], 'NFC as Technology Enabler', http://www.nfc-forum.org/aboutnfc/tech_enabler/. 40

[39] Object Management Group [2011-08-10], 'Unified Modeling Language: Infrastructure', http://www.omg.org/spec/UML/2.0/. 16

[40] Open Handset Alliance [2011-08-19], 'Open Handset Alliance', http://www.openhandsetalliance.com/index.html. 19

[41] P.Clements, R. and M.Klein [2002], 'Evaluation software architectures: methods and case studies'. 26

[42] QSR International [2011-06-17], 'Nvivo 9', http://www.qsrinternational.com/products_nvivo.aspx. 9

[43] R.E.Kraut, M. and J.Siegel [1996], 'Collaboration in Performance of Physical Tasks: Effects of Outcomes and Communication', *Computer Supported Cooperative Work 96* . 42, 43

[44] Robson, C. [2002*a*], *Real World Research*, 2 edn, Blackwell Publishing, p. 270. 4

[45] Robson, C. [2002*b*], *Real World Research*, 2 edn, Blackwell Publishing, p. 458. 8

[46] Rossi, M. and Sein, M. K. [2011-04-14], 'Design Research workshop: A proactive research approach', http://www.cis.gsu.edu/~emonod/epistemology/Sein%20and%20Rossi%20-%20design%20research%20-%20IRIS.pdf. 1

[47] Samsung Electronics co. [2011-08-02], 'Samsung Galaxy SII', http://www.samsung.com/global/microsite/galaxys2/html/. 7

[48] SESAM LÅS & LARM [2011-08-16], 'Proffs på skydd och säkerhet!', http://www.sesamlas.se/. 4

[49] Skype [2011-06-17], 'Skype', http://www.skype.com. 19

[50] Software Engineering Institute [2011-08-09], 'Architecture Tradeoff Analysis Method', http://www.sei.cmu.edu/architecture/tools/evaluate/atam.cfm. 39

[51] Sommerville, I. [2007], *Software engineering*, 8 edn, Persons Education Limited. 37

[52] Tellis, W. [2011-08-05], 'Application of a Case Study Methodology', http://www.nova.edu/ssss/QR/QR3-3/tellis2.html. 37

[53] The Tech Terms Computer Dictionary [2011-08-02], 'Plug and Play definition', http://www.techterms.com/definition/plugandplay. 1, 6

[54] Vaishnavi, V. and Kuechler, W. [2011-04-14], 'Design Research in Information Systems', http://desrist.org/design-research-in-information-systems/. 1, 3

[55] Visual Paradigm International [2011-08-08], 'Visual Paradigm for UML 8.2', http://www.visual-paradigm.com/product/vpuml/. 16

[56] VOLVO TRUCK CENTER SVERIGE [2011-05-19], 'Volvo Lastvagnar testar konsumentteknik för ökad kvalitet inom eftermarknadsservicen', http://www.volvotrucks.com/dealers-vtc/sv-se/VolvoTruckCenter/newsmedia/pressreleases/Pages/pressreleases.aspx?pubid=10773. 1

[57] W3C [2011-08-24], 'Extensible Markup Language', http://www.w3.org/TR/xml/. 16

[58] Zhang, Z., Wu, H., Wang, W. and Wang, B. [2010], A smartphone based respiratory biofeedback system, *in* 'Biomedical Engineering and Informatics (BMEI), 2010 3rd International Conference on', Vol. 2, pp. 717–720. 42

A Appendix: Requirements Specification

Aioli will have two different user groups which shall have different access rights (see REQ-6.2). Hence, the requirements stated in A.1, A.2, A.3, and A.4 are for the service technician user group. The requirements in A.5 are for the superintendent user group.

Service

REQ-1.1 The system shall be able to retrieve and present a list of all connected components. For each component name, version, group membership, usage counter, device ID and host ID should be retrieved.

REQ-1.2 The system shall be able to retrieve and present operation mode.

REQ-1.3 The user shall be able to select to display a list of common problems and how to solve them.

REQ-1.4 The user shall be able to add a common problem and a solution to it.

REQ-1.5 The support at ASSA shall be able to add a common problem and a solution to it.

REQ-1.6 The user shall be able to set operation mode.

REQ-1.7 The user shall be able to run a test by following a set of instructions to allow the system to collect diagnostic data. The system shall collect and present the information.

REQ-1.8 If possible, the Aioli application shall analyze the data and give suggestions towards what the reason behind the error could be.

REQ-1.9 The user shall be able to create a support case.

Installation

REQ-2.1 The system shall provide an installation documentation possibility containing the following:

- **REQ-2.1.1** The system shall present installation steps which all must be completed.
- **REQ-2.1.2** The user shall be able to add comments.
- **REQ-2.1.3** The user shall be able to search previous installation documents per facility.
- **REQ-2.1.4** The user shall be able to email the documentation.
- **REQ-2.1.5** The user shall be able to save the documentation in a database when the installation is finished.
- **REQ-2.1.6** The user shall be able to save the documentation for an ongoing installation in the smartphone.

When Aioli is connected to the door lock system:

REQ-2.2 The system shall be able to retrieve and present operation mode.

REQ-2.3 The user shall be able to set operation mode.

REQ-2.4 The system shall be able to retrieve and present a list of all connected components. For each component name, version, group membership, usage counter, device ID and host ID should be retrieved.

If something went wrong:

REQ-2.5 The user shall get the option to save the document and switch to Aioli's service mode.

Support Case Service Technician

REQ-3.1 The system shall be able to add a list of all connected components and all their details from REQ 1.1.

REQ-3.2 If the user has performed a diagnosis during service (REQ 1.7) it should be added.

REQ-3.3 The system shall be able to add the operation mode.

REQ-3.4 The user shall be able to add address of the door system.

REQ-3.5 The user shall be able to record sound.

REQ-3.6 The user shall be able to add a sound file.

REQ-3.7 The user shall be able to record video.

REQ-3.8 The user shall be able to add a video file.

REQ-3.9 The user shall be able to take one or more pictures.

REQ-3.10 The user shall be able to add one or more image files.

REQ-3.11 The user shall be able to add text.

REQ-3.12 The user shall be able to add contact information.

REQ-3.13 The user shall be able to sent the support case to the ASSA support via email.

Online Support Service Technician

REQ-4.1 The system shall be able to set up a connection between Hi-O and the ASSA support.

REQ-4.2 The system shall be able to set up a video call with the ASSA support.

REQ-4.3 The system shall be able to facilitate remote system diagnostics.

Breakdown Support Case

REQ-5.1 The user shall be able to create a support case.

REQ-5.2 The system shall be able to retrieve a list of all components. For each component name, version and device ID, host ID, group membership and usage counter should be retrieved.

REQ-5.3 The system shall retrieve operation mode.

REQ-5.4 The user shall be able to add address of the door system.

REQ-5.5 The user shall be able to record sound.

REQ-5.6 The user shall be able to add a sound file.

REQ-5.7 The user shall be able to record video.

REQ-5.8 The user shall be able to add a video file.

REQ-5.9 The user shall be able to take one or more pictures.

REQ-5.10 The user shall be able to add one or more image files.

REQ-5.11 The user shall be able to add text.

REQ-5.12 The user shall be able to run a test by following a set of instructions to allow the system to collect diagnostic data. The system could collect but not present the information from the diagnostic run through.

REQ-5.13 The system shall give feedback when the diagnostic test is completed.

General Requirements

The following requirements are valid for every scenario stated above.

REQ-6.1 The user shall be able to enter his login information. The user shall not be able to use Aioli without this step.

REQ-6.2 According to the login information Aioli shall give the right user group access rights. One user group is for the service technician who has all rights, and the other user group is for the superintendent.

REQ-6.3 The connection between the Hi-O door lock system and the Aioli application shall be done via Bluetooth.

REQ-6.4 In case the Bluetooth Connection between the Hi-O door lock system and the Aioli application goes down there shall be an automatic reconnection, and the user shall be notified if the reconnection cannot be established.

REQ-6.5 The user shall be able to save his contact information in the Aioli application in a settings screen.

REQ-6.6 The user shall be able to access the settings screen from every other screen in the Aioli application.

B Appendix: Quality Attributes

Quality Attribute: Security

Scenario 1: Authentication / Authorization

Source of stimulus:	End user or intruder.
Stimulus:	Wants access to the Hi-O Door Lock System via Aioli.
Environment:	Aioli is not yet connected to Hi-O.
Artifact:	GUI and DatabaseConnector packages.
Response:	Aioli must check if the individual is authorized, and also which user group he belongs to. If access is granted, the user is given access to the functionality which that user group is allowed to access. Password is not shown on screen.
Response measure:	The authorization process shall not take longer than 30 seconds.

Scenario 2: Data Confidentiality

Source of stimulus:	Intruder.
Stimulus:	Man-in-the-middle problem. An intruder tries to access the communication link between the door lock system and a user's Aioli application.
Environment:	At runtime.
Artifact:	Communication Layer.
Response:	Aioli encrypts all data which is sent over the communication link
Response measure:	Only encrypted messages are sent between Aioli and the door lock system.

Quality Attribute: Portability

Scenario 1: Change Platform

Source of stimulus:	Developer(s).
Stimulus:	Wish to change the smartphone platform. For example the iPhone platform iOS shall be used.
Environment:	Not all service companies use Android smartphones, but they should be able to use the application.
Artifact:	On a architectural level. Every smartphone platform has a different implementation language, API and design recommendations. Therefore, the use of the Android API is separated from the application logic.
Response:	Application works on a different platform and no changes occur in the behavior.
Response measure:	For two developers it will take around five months to migrate and test the whole application.

Quality Attribute: Modifiability

Scenario 1: Ease of Modification

Source of stimulus:	Developer(s).
Stimulus:	Wish to make changes to the Aioli application. For example if the functionality of the Hi-O door lock system has changed then modifications need to be made to Aioli. Also, it can be possible that the graphical user interface needs to be adapted.
Environment:	Design time, maintenance time.
Artifact:	The Aioli application.

Response:	Modifications will be done without affecting other functionality.
Response measure:	For a developer with Android development experience and knowledge about Aioli, a major change should not take longer than 10 working days. A major change is a modification that affects at least two layers. A minor change should not take longer than three working days and affects one package.

Quality Attribute: Usability

Precondition is that the user is familiar with the smartphone platform in general and does not need training in getting accustomed to for example an Android phone. Also, the design of the application follows the Android User Interface Guidelines [2], which facilitates a fast acclimatisation for the end user.

Scenario 1: Recovery

Source of stimulus:	The end user.
Stimulus:	User wants to be able to easily recover from mistakes.
Environment:	Happens at runtime.
Artifact:	The Aioli application.
Response:	When a user has inserted information and accidentally presses a button that cancels the operation, the inserted information will not be lost. When the user returns, the information will be presented to him. Thus, he will not have to enter everything again.
Response measure:	System response time shall not be more than one second.

Scenario 2: Learnability

Source of stimulus:	The end user.
Stimulus:	The user wants to quickly learn how to use the software and how to find needed functionality.
Environment:	Happens at runtime.
Artifact:	The Aioli application.
Response:	The application allows the user to easily navigate through the application by being divided into a sequence of steps which need to be followed.
Response measure:	At user testing, the end user understands how to complete all the steps of the application.