placeholder

Master thesis Computer Science

Dennis Cardinaels

Promotor: Professor Mieke Haesen

Assistant: Jens Brulmans

2018 – 2019

Preface

TODO

Abstract

TODO

Contents

1 Introduction

2 The Electronic Health Record

Health information systems have become an integral part of health care. They support patient care as well as administrative and financial tools. At the heart of these systems lies the electronic health record. An electronic health record (EHR) is a repository of electronically maintained information about an individual's health status and health care, stored such that it can serve multiple legitimate uses and users of the record [1]. An EHR system provides tools to manage and interact with these records. These tools include reminder generation, data analysis, order entry, and decision support. It helps the clinician to organize, interpret and react to medical data. A record where the health information is managed by the patient, is called a personal health record [2]. We will focus on the electronic health record and the systems that interact with it.

The purpose of this section is to provide a general overview of electronic health records. First, we describe the issues surrounding paper which ultimately led to the development of the first digital health systems. Hereafter, we define the five essential components of an EHR system. Section 2.3 describes the impact of EHR systems by listing benefits and drawbacks. The next two topics of discussion are interoperability and usability. Recent technological advancements led to new methods to deliver care, such as telemonitoring which we discuss in section 2.6. To conclude, we briefly discuss privacy and security concerns associated with digitally stored medical data.

2.1 Moving away from paper

For modern medicine, traditional paper-based medical records are not suited for today's world filled with technology. The drawbacks of information on paper are obvious when compared to digitally stored information.

Storage Paper records need to be stored in a safe location and require a lot of physical space. Also, the organization of these records is a daunting task, doubly so when they are fragmented across multiple locations. This is a process that wastes time and therefore increases health care costs. Also, losing or misplacing records is a possibility. For example, should this happen to lab results, tests have to be redone which again wastes time and increases costs. Finally, the amount of paper added to the records on a weekly basis is substantial and consequently very eco-unfriendly [3]. The ink and paper costs should not be underestimated. Storing data in a digital format solves these issues, but raises several other questions: shall we store the data on premise or choose a cloud provider? What is our backup policy?

Data quality Medical information stored on paper is static and requires more effort from clinicians to process it. For example, the clinician has to compare two manually filled-in paper charts in order to assess the recovery of a patient. What if the data is illegible due to poor handwriting? What if data is incorrectly read from the computer display or simply not written down? Such medical errors directly impact the quality of care that is delivered [4, 5]. Digital records allow computation such as data aggregation, data processing and statistics generation. Also, a summarizing paper noted that the use of EHR systems leads to more complete, accurate, comprehensive, and reliable data compared to paper-based records [6]. This is the result of system functions that detect incorrect data input and ensure that all data fields are filled.

Medical data such as x-ray images are captured at a very high resolution. Software tools can view these images without much loss in quality, while printed images are subject to significant quality loss. Images can be printed, but other data types can not. To bundle this kind of data with a paper dossier, a physical medium such as the aged compact disc is required. Again, this raises several storage and transportation issues which the EHR avoids.

Accessibility Paper records are location-bound and are often only accessible by one person at a time. Transporting paper records requires a lot of manual work in which the clinician sends either the original or a copy of the record. Another option is to convert the paper record to a digital format. One study described that updating several copies of medical records was a very cumbersome process for clinicians [7]. If a copy of a record is updated, how can these changes be reflected to the original and the other existing copies? How many copies are in circulation and where are they located? Converting the paper records to a digital format raises several other questions: where are data values entered inside the digital system? What if there are extra notes scribbled on the document? What if values are missing, but required by the EHR? Data exchange and saving changes seems simpler for digital records, however interoperability of different systems is complex. This is discussed in section 2.4.

Security While paper records can be safely stored under lock and key in any particular room, several security issues still exist. For example, the storage location can be broken into. This allows the perpetrator to retrieve, alter, or destroy critical medical information. Other possibilities of data loss include flooding and fires. These are scenarios that can't be predicted and can happen at any time.

A common strategy to detect data alteration and prevent data destruction is to create regular backups. However, for paper medical records this is simply not feasible. These data sets include thousands of individual papers. The resources required to create copies of such a large data set is enormous and increases the ecological footprint of the institution substantially. For digital records this is a much easier process, but in this case other privacy and security issues arise. These are discussed in section 2.7.

Electronic health record systems solve many of the aforementioned problems associated with the use of paper. However, there are other factors responsible for the continuing presence of paper in clinical environments. Three categories are predicted to be the cause of paper generation: policy requirements, sub-optimal system design, and flaws in the user interface of the EHR system [3]. The referred study focused on user interface flaws and highlighted the following issues:

- Cumbersome interface design leads to handwritten notes on paper.
- The EHR system is not well integrated into the clinical workflow. This leads to paper-based workarounds which *do* align with the workflow.
- The visual organization of the data in the EHR system is incompatible with the mental model of the clinician, which leads to manual transformation of patient data.

Notice that these subcategories relate to usability and human-computer interaction issues. The study interviewed twenty individuals concerning the use of paper in their workflow. Afterwards, the researchers categorized recurring paper-based workaround strategies found during analysis. This resulted in the following 11 categories, ordered by descending frequency of use:

- Efficiency: enhanced perceived or actual efficiency when using paper.
- Workarounds related to the clinician's knowledge of the health system, skill level with technology, and ease of use of the health system. For example: paper is used because it is simpler than the software.
- Memory: cases where paper is used as a reminder tool.
- Sensorimotor preferences: paper serves as a means of having something concrete to deliver or to quickly jot down some notes.
- Awareness: paper helps clinicians be aware of new information.
- Task specificity: cases where the health system lacks specificity, is not customizable, or sends too many alerts (alert overload).
- Task complexity: paper processes are used because the health system does not support it. An oncology order is an example of a complex task, tailored specifically for each patient, which a simple EHR system may not support.
- Data organization: paper may present information better compared to a cluttered computer screen.
- Longitudinal data processes: in case tracking data over time is easier to do on paper.
- Trust: paper is used as a form of proof which health software can't provide.
- Security: using paper to avoid the use of an insecure system.

While these workarounds can improve efficiency, they also circumvent the health system and its safety checks, which leads to medical errors. However, they also imply that the EHR system is not in line with the workflow of the clinicians. Several of these workarounds are caused by poor usability, which is the topic of discussion in section 2.5.

2.2 Components

As mentioned before, EHR systems do not simply store medical information. They consist of many functional components which ultimately determine how well they perform in health care. We define the following five components as essential [1]:

Integrated view of patient data An EHR system must allow storage of a wide range of data types. This includes text, numbers, images, video, and others data formats. A lot of paper may be generated in case the system doesn't support a common data type, as described in section 2.1. Data standards were developed to store and exchange both simple and complex data types, ranging from systolic blood pressure values to x-ray images. Medical data standards are the central topic of section 2.4.1.

Clinician order entry The procedure in which the clinician enters treatment instructions is called order entry. An order entry system assists the clinician during the decision-making process to ensure that the instructions are correct. As a result of more complete and correct data, less medical errors are made. Paper is also avoided because care settings can exchange these orders via the EHR system.

Clinical decision support A decision support system aids the clinician by suggesting actions when certain situations occur. If for example, a patient is due for vaccination, the system displays a pre-filled order dialog to remind the clinician. The system can do this for a bulk of patients, which saves precious time otherwise spent on performing manual checkups. Decision-support implementations benefit from artificial intelligence. As such, these systems are often complex, due to the many parameters that are involved in the decision-making process.

Access to knowledge resources Clinical questions often arise during the clinician's workflow. Instead of asking colleagues or searching through multiple manuals, the clinician can turn to the EHR system for information. Digital systems can easily access large sources of information. Also, if the EHR system is context-aware, searching for the required information takes even less time.

Integrated communication and reporting support Communication lies at the heart of health care delivery. It is common for patients to receive care from several clinicians spread across multiple institutions or departments. The availability of communication tools in an EHR system directly affects the quality of care as paper is avoided. These tools should support common standards to facilitate interoperability and efficient data exchange, which is discussed in section 2.4.

All aforementioned components should be present in health software. This example of a simple prescription management tool features all five components:

- Present a list of medication currently taken in a clear manner.
- Create, edit, or remove prescriptions.
- Provide alerts for drug-drug and drug-allergy interactions. Suggest medication dosage with respect to parameters such as weight.
- Provide extra information regarding medicines, such as side effects.
- Send prescription orders to the pharmacy.

This prescription management tool is an example of a tool commonly found in EHR systems. Many more examples are given in section 3.4. Ideally, a single software package provides all the necessary tools. If some functions are still missing, institutions may deploy other applications to fill these gaps. In such cases, there is a need for interoperability, discussed in section 2.4.

2.3 Impact

The transformation of the health care industry towards the use of technology has a profound impact on many aspects, such as finances and efficiency. As mentioned in section 2.1, paper is not reliable nor durable to provide efficient care. The identification and research process of all the potential benefits and drawbacks of EHR systems is not simple. There are countless of practices, small and large, providing many different types of care. As a result, most studies focus on one care setting at a time, such as the intensive care unit of a hospital. Therefore, the effects of an EHR system found in one care setting should not be generalized to others.

One study conducted in 2011 summarized literature which studied the benefits and drawbacks of EHR systems in different care settings [8]. This study served as the primary source of information for this section. First, we describe the advantages of EHR systems on several outcomes, whereafter we describe the disadvantages.

2.3.1 Potential advantages

The positive impact of EHR systems was studied for clinical, organizational, and societal outcomes. For each of the outcomes a definition is given, together with the observed benefits.

Clinical outcomes Measurable changes observed in quality of care are related to this outcome. Quality of care can be defined as "doing the right thing, at the right time, in the right way, for the right person, and having the best possible results" [9]. Quality of care includes six dimensions [10]. However, research focused mainly on the following three: effectiveness, efficiency, and patient safety.

EHR systems lead to increased adherence to evidence-based clinical guidelines, resulting in more effective care. There are three possible reasons that explain why clinicians don't follow these guidelines: either they don't know them, don't know it applies to the patient in question, or have insufficient time. An EHR system tries to overcome these issues. Computerized alerts are also linked to the improvement of care effectiveness. EHR systems are associated with keeping test redundancy to a minimum. Redundant testing is inefficient, costly, and time-consuming.

Studies found that the use of EHR systems resulted in a significant reduction of medical errors. This directly improves patient safety. However, a few studies discovered that the error rate in fact rose, which we later discuss as an unintended consequence of EHR systems.

Organizational outcomes The billing system provided by an EHR system increases revenue. Reasons for this are a decrease in billing errors, and the improved ability to capture and track bills. EHR systems also avoid many costs. To give a two examples of cost savings: the printing and managing records is no longer necessary, and as previously discussed, less redundant tests are performed. Other benefits include better operational performance, improved legal and regulatory compliance, and fewer malpractice claims.

Societal outcomes The increased availability of data improves the ability to conduct research. This also helps the public health field in monitoring diseases and the detection of looming outbreaks. The use of EHR systems is also linked to increased physician satisfaction [11], which is a detrimental factor to improving quality of care.

2.3.2 Potential disadvantages

An EHR system provides several financial benefits. However, it also introduces new financial issues. The adoption and implementation of an EHR system has high upfront costs which involves the purchase and installation of hardware and software, conversion of paper records to digital ones, and user training. Now that EHR systems are common, acquisition costs saw a significant decrease.

Once an EHR system is acquired, it needs to be maintained. The evolving nature of technology requires frequent hardware replacements and software updates. Consequently, users need additional training to adapt to these changes. This also leads to revenue loss due to a decrease in productivity. The high upfront and the ongoing maintenance costs are considered to be largest barrier to the adoption of EHR systems [12].

The use of an EHR system may cause some unintended consequences [13], such as an *increase* in medical errors. Poor system usability, lack of training, and lack of system integration are possible causes of this rise [14]. Usability is the topic of section 2.5. Another unintended consequence is that an EHR system may evoke negative emotions due to workflow disruptions and adaptation difficulties. To conclude, clinicians may become overdependent on technology. Institutions should guarantee their ability to provide care in case there are technical issues.

2.4 Interoperability

Health institutions often deploy several systems to satisfy all their requirements [15]. As a result, data is spread over several repositories which opens up the possibility of data duplication and synchronization issues. If two systems are unable to communicate with each other, clinicians may resort to paper workarounds as mentioned in section 2.1. Sensory data also needs to find its way into the health record of the patient. If the measuring device has no means to directly send its data to the EHR system, then clinicians are forced to do this manually. Therefore, system interoperability is very important in a health care setting.

Interoperability is "the ability of two or more systems or components to exchange information and to use the information that has been exchanged" [16]. To go into more detail, interoperability can be divided into four concepts [17]:

- Technical interoperability: moving data from one system to another.
- Semantic interoperability: allow the sender and recipient to understand the same data in the same way without ambiguity.
- Process interoperability: when people share a common understanding across a network, systems interoperate, and work processes are coordinated.
- Clinical interoperability: the ability for two or more clinicians in different care teams to transfer patients and provide seamless care to the patient.

Health care systems and devices should implement standards to achieve interoperability. If every application stores data in its proprietary format, many point-to-point interfaces need to be created to allow communication. The left graph of figure 1 illustrates this. However, if systems adhere to a common standard by which they communicate, this is avoided. On figure 1 the standard is indicated by the star in the middle of the right graph. The next section briefly describes standards.

2.4.1 Standards

When excessive diversity creates inefficiencies and affects effectiveness standards are required [1]. A hospital contains many independent units spread across

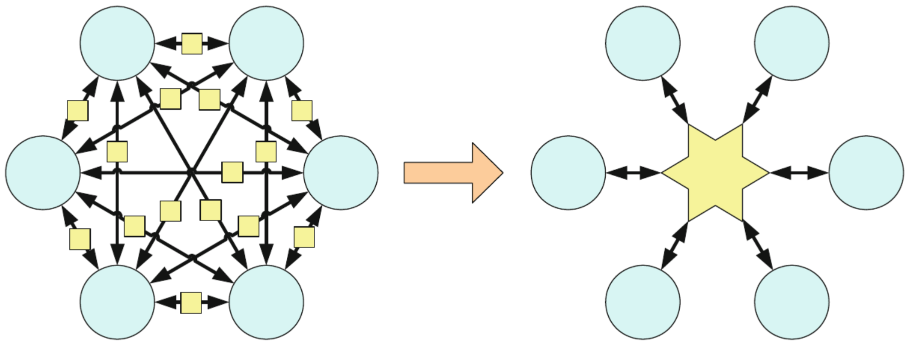

Figure 1: Comparison of communication by point-to-point interfaces and standards.

primary, secondary, and tertiary care. These units use software best fit for their practice and all record different types of data. For example: the admissions system records patient diagnosis, the pharmacy records prescriptions that were handed out, and the laboratory system records test results. Inevitably, transfer of data between these units is required.

To coordinate multiple systems, data must be exchanged. Nowadays too many different systems exist to create point-to-point interfaces for. Standards try to resolve this problem by defining guidelines which software systems should follow in order to communicate data. An efficient standard requires that data can be easily stored and presented towards the users of an EHR system. Also, security measures, such as authentication and access control, need to be interwoven with these standards.

The use of standards in an EHR system leads to better interoperability which in turn leads to lower development costs. Over time, the continuous addition of proprietary data structures leads to difficult to maintain software and a higher risk of critical bugs. New medical devices and software that comply to standards prevent this. However, there is currently no regulator that enforces the use of existing standards. As a result, good interoperability lies in the hands of the software vendors.

We close this section by giving two examples of standards. To start, the most commonly used set of standards is Health Level 7 [18]. These message-based standards provide a framework for the exchange, integration, sharing, and retrieval of electronic health information. Second, DICOM is used for the communication and management of medical imaging information [19]. DICOM allows the exchange of persistent objects, such as images, which is not possible with HL7 messages.

2.5 Usability

We define usability as follows: "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" [20]. As the health care sector deploys more and more devices and software solutions, it is of importance that these are usable. In case they are, fewer medical errors are made which improves patient safety and increases the efficiency of clinicians. Systems with poor usability may have the opposite effect, increasing the amount of errors [14]. This also leads to more paper generation as clinicians want to circumvent the poorly usable system, as mentioned in section 2.1. Mobile technology seen today features smaller screens and receives input primarily via touch. These devices ask for completely different user interface design principles. However, no research concerning mobile device usability in health care settings was found.

Problems detected at the start of the development cycle are less resource intensive to solve. They may expose fundamental issues which would have been a lot more costly to deal with at later stages of development. Therefore, it is important to conduct usability evaluations early on in the development phase to ensure that the health systems are developed in a cost-effective manner that results in an efficient, effective, and useful product [21]. Usability is difficult to evaluate in the health care sector, due to the specific needs that many different work environments require. By providing customization options, each care setting may tailor the system to their wishes. However, this kind of flexibility may create other usability issues.

Several methods exist to evaluate usability. An expert review is an example of such a method in which an expert evaluates the usability. A heuristic evaluation is similar in this regard as it consists of several experts evaluating a system. However, this method tests a system against a set of predefined usability heuristics. To give an example, Jakob Nielsen created 10 usability heuristics that are still widely used today [22]. The heuristics are described in appendix A.1. It should be noted that expert reviews identify predicted usability issues. To have a better chance of identifying issues that may arise during real use, an actual end user should evaluate the system. However, conducting such tests are costly in terms of time and resources, which may not be feasible. To conclude, the following list describes 14 usability principles, which are similar to Nielsen's heuristics, for the design of EHR systems [23]:

1. *Consistency*: design consistency and standard utilization.
2. *Visibility*: system state visibility.
3. *Match*: system and world match.
4. *Minimalism*: minimalist design.
5. *Memory*: memory load minimization.
6. *Feedback*: informative feedback
7. *Flexibility*: flexible and customizable system.
8. *Message*: useful error messages.
9. *Error*: use error prevention.
10. *Closure*: clear closure.
11. *Reversibility*: reversible actions.
12. *Language*: user language utilization.
13. *Control*: user control.
14. *Documentation*: help and documentation.

2.6 Telemonitoring

At the start of 21st century mobile phones were small bricks with functionality limited to calling and text messaging. Today, smartphones are used to navigate the internet, play games, watch media, and so forth. These recent technological advancements bring many new opportunities which may transform the health care industry. An example of such an opportunity is telemonitoring. Monitoring patients at a distance through the use of information technology is called telemonitoring [24]. Here, the patient uses special monitoring devices to gather data which in turn is sent to the health care provider.

While many health conditions can be monitored, telemonitoring is most prevalent in chronic disease management. Chronic diseases are defined as long-lasting health conditions and include the following four major types: cardiovascular disease, cancer, diabetes, and chronic obstructive pulmonary disease (COPD) [25]. Cardiovascular disease is the leading cause of death worldwide [26]. Together with cancer, these two diseases accounted for almost half of all deaths in the United States of the year 2015 [27]. Speaking of finances, 86% of all health care expenditure went towards patients suffering from one or more chronic conditions [28]. The prevention of chronic disease and improving the management thereof after onset, will yield significant benefits.

These conditions often occur as a result of an unhealthy lifestyle [29]. Therefore, it is very important to spread awareness on healthy habits. Suggestions often include the following: avoid smoking, maintain a healthy weight, exercise daily and limit time spent sedentary, and maintain a healthy diet. These lifestyle changes become even more important during chronic disease management.

The management of chronic diseases may benefit greatly from telemonitoring. Patients have access to care, while having the comfort of staying at home. As a result, the patient saves time and costs associated with consultations. Relatives of the patient may also be more at ease as they know the caregiver is monitoring the situation. On the other end, health care institutions save resources as care is delivered without having the patient visit [30].

At its core, telemonitoring relies heavily on interoperability as it has the potential to add many different devices to the health care system. The caregiver should be able to receive, read, and react to this data. Interoperability should already be accounted for in case the device was specifically engineered for the health care sector. However, a smartphone for example, is built for general use and not necessarily for health care. As such, these devices rely mainly on software and other connected devices to allow telemonitoring. On the other hand, extra attention must be given to usability as the telemonitoring devices often have small-sized screens. Good user interface design will lead to higher user satisfaction and may therefore impact the patient's adherence towards the management process.

2.7 Privacy & security

Privacy refers to the desire of a person to control the disclosure of personal health and other information [1]. On the other hand, confidentiality is the ability to control the release of personal health information to a care provider under the agreement that the information will not be spread or used further. Security is the protection of privacy and security, which is achieved through policies, procedures, and safeguards.

We can ask several questions related to health information data access and storage: who owns the data? Is it the health provider or the patient? What types and how much data needs to be stored? Who can read the data? Who can write to the data? Can someone access specific health information without consent? In order to deal with the some of the challenges concerning data access and storage, these questions need to be answered [31].

Medical data access raises several privacy concerns. For example, the research of the medical data of a large population may uncover looming threats or an epidemic. However, this is only possible if researchers have free access to this data, as it is not feasible to ask every individual of a population for consent. However, medical data should not be too accessible. As more and more people gain access to health data of the population, the chance of data misuse or unauthorized access increases. Therefore, striking a balance between free information access and protection of privacy and confidentiality is difficult. An argument that promotes free access is to anonymize the data by removing identifiers such as the person's name. However, the individual pieces of information can still reveal the their identity.

Compared to paper, access to digital repositories of health records is easier. This calls for extra security measures to unauthorized access. To gain access to certain records, clinicians are required to authenticate themselves. Also, access control be present. This security technique limits access to certain data based on the privileges associated with the authenticated user. Whether the authentication succeeds or not, the audit system logs will always record the attempt. Such systems need clear rules and policies defined by the institution on who can access what. Data encryption is mandatory, should data breaches ever occur.

To further prevent malicious use of health data, institutions should educate their staff to make them aware of the privacy rules and policies that are in place. Appropriate punishments should be in place should someone violate these rules.

3 Design

This section describes the design process that was followed to create the application featured in this thesis. First, we propose a solution for several issues highlighted in the literature review. Based on the proposal, additional literature was reviewed to guide the design of the prototype. Section 3.2 describes these guiding principles. The first step of the design process led to the creation of several personas and scenarios, found in section 3.3. Hereafter, we define the functionality of the prototype with the help of a small brainstorm session. To conclude, a concrete specification and mockups of the prototype are given.

3.1 Proposal

The literature review indicated that several issues arose after EHR systems were installed. In this section we gather those issues and propose a solution that remedies them. During the definition of our solution, arguments are given as to why certain choices are made.

Current issues In some cases, the use of an EHR led to an increase in medical errors. Research showed that poor usability was one of the causes for this rise [14]. Another study found that the use of paper workarounds was also responsible for this increase [3], which again hints at a system with poor usability. However, the use of these workarounds also indicates that the system is not in line with the workflow of the clinician.

The installation of a new EHR system disrupts the current workflow significantly [8]. Large updates to an existing system may also cause disruptions. Clinicians need to be trained in order to work efficiently with the new system. If there is insufficient training, more medical errors may occur. It takes some to become accustomed to these changes, which leads to a temporary loss of productivity. Sometimes this adaptation process is difficult for some individuals which may spark negative emotions.

Solution We propose a EHR web application featuring at its core a customizable dashboard. The clinician may freely place and arrange modules on the dashboard, similar to placing widgets on the homescreen found within the Android OS. Behind-the-scenes, creating new modules for the dashboard should be a straightforward process. In terms of usability, the modules share a clear and concise design, and the dashboard can adapt to different screen sizes. We now go into more detail.

The choice of creating a web application was quickly made, since these applications are cross-platform out of the box. Updating the application is done by simply refreshing the web page. The development and extension of native applications for several operating systems is very costly as multiple teams of developers are needed. Also, recent web technology advancements allow much more complex applications to be run straight from the browser. To give an example, Microsoft provides a web app version of its Office Suite.

The dashboard presents a novel approach regarding interoperability. Instead of providing interoperability for multiple standalone applications, the dashboard is continuously expanded with new modules. In other words, applications are integrated as new modules of the dashboard. This module-based approach has several benefits:

- There is no need to switch between multiple standalone applications, which may have completely different designs.
- Clinicians only need to learn to use the modules that are required for their workflow, instead of a complete software package.
- Many different workflows can be supported by using a combination of modules.
- The dashboard can easily adapt to workflow changes by allowing the addition and removal of modules.
- Module updates do not affect other modules or the dashboard, preventing total workflow disruption.
- Free placement of modules on a grid allows the dashboard to support many different screen sizes.

To guarantee a streamlined user experience throughout the dashboard, new modules inherit a template which determines its appearance. Changing this template, will change the appearance of all the modules. Also, dashboard will be designed with touch technology in mind to prevent usability issues such as the fat-finger problem and screen clutter.

In practice, vendors develop modules according to guidelines defined by the dashboard application. Integrating new modules to the dashboard still requires some work from the institution's IT staff, but no where near as much when integrating a standalone application. Once the module is integrated, it can be seamlessly updated. The dashboard also has the ability to import patient data.

In the end, the dashboard application features cross-platform availability, easy extension of functionality, a usable and customizable interface, and a simple learning process. At this stage, additional literature was consulted to facilitate the creation of an effective design.

3.2 Guiding principles

The translation of the aforementioned proposal to an intuitive prototype is guided by several principles. It was paramount during the design and implementation phases that the final prototype provided a positive user experience. This final prototype is the result of a series of steps defined by the MuiCSer

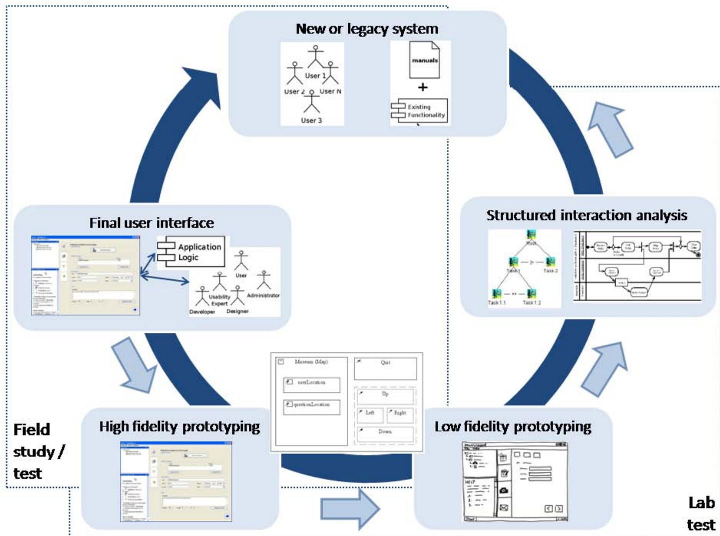

Figure 2: The MuiCSer process.

framework, which is explained first. Hereafter, several design principles are mentioned that serve as helpful guidelines during the design process of a dashboard application.

3.2.1 MuiCSer

This section describes the MuiCSer framework, designed for user-centered software engineering processes in a multidisciplinary context [32]. This framework focuses on optimizing the user experience during the entire software engineering cycle to ensure that the needs of the end user are fulfilled. By combining user-centered design methodologies and software engineering principles, the user experience of the final product can be improved substantially. Given the multidisciplinary context of this thesis, MuiCSer guided the entire development process leading to the final prototype described in section 4.

The MuiCSer process is summarized in figure 2. After each phase, the result is evaluated, verified and validated to ensure that the required functionality is present. In turn, the received feedback can be used to reiterate over the previous phase. On the figure, this is denoted with the light blue arrows, while the single dark arrow represents the overall direction of the process. What follows is a brief description of each phase:

1. New or legacy system The MuiCSer process starts with either an existing system in need of improvement, or with a new system which needs to be built from the ground up. This requires an analysis of the tasks and needs of the user, as well as the objects and resources required to perform these tasks.

Personas and scenarios are the resulting artifacts of this phase. First, personas describe the personalities of the potential end users including hobbies, skills and the environment they surround themselves in [33]. Its goal is to uncover behavior patterns which can be of use when designing a user interface. Second, scenarios tell stories describing the use of a fictitious system from the point of view of a persona [33]. To summarize, personas and scenarios try to sketch the usage of the system for which a design must be made. These artifacts are found in section 3.3.

2. Structured interaction analysis During this phase, the results of the analysis are used to create task models. These models specify concrete tasks and goals which can be dissected into specific actions or steps the user has to take. These artifacts lay the foundation for designing a user interface which supports these tasks and goals. Task models were not created. Instead, via a brainstorm session (section 3.4), common tasks of a health system were grouped to form modules. The user interface will be designed to support these modules.

3. Low fidelity prototyping After the task definition, low fidelity prototypes are created. Paper sketches and mockups are examples of low fidelity prototypes. Without spending too much time and resources, presenting these prototypes to the end-user or customer can yield valuable feedback. However, there is often no functionality present. Typically multiple versions of these prototypes are created until the customer is satisfied, after which the development of a high fidelity prototype starts. Low fidelity prototypes were created for all modules defined during the brainstorm session and are found in section 3.5.

4. High fidelity prototyping The development of high fidelity prototypes requires a lot more effort compared to low fidelity prototypes, as they offer functionality closely resembling the final product. However, the end user can now test both the design *and* the functionality. This yields much more meaningful feedback. Section 4 describes the development of the high fidelity prototype.

5. Final user interface When the latest iteration of the high fidelity prototype satisfies all user requirements, the final user interface can be created. The high fidelity prototype may serve as the starting point of the final product in order to save time and resources. As a final step, the task models are checked against the resulting interface to check if all required functionality is present. No final product was created during the period of this thesis. However, the results of the usability test described in section 6 should determine if it is feasible to commit further research towards the prototype.

3.2.2 Dashboard design

The proposal mentions a dashboard. However, dashboards are used in many contexts. Common examples include a car speedometer and stock analysis. In computer networking, dashboards are used to quickly interpret network traffic. In our case, we want to empower the user to create his or her own dashboard. To facilitate this goal, several design principles should be kept in mind.

While some dashboards provide fancy graphics, many miss the key point of what they are supposed to accomplish. The primary goal of a dashboard is clear communication, which is achieved through effective design. A formal definition is as follows: “A dashboard is a visual display of the most important information need to achieve one or more objectives; consolidated and arranged on a single screen so information can be monitored at a glance.” [34]. This definition contains four key characteristics:

Dashboards are visual displays. It is important that data is represented visually. Graphics such as charts allow more efficient communication compared to textual information. For example, trends and outliers are easier spotted in a line chart when compared to a table containing the same data. Instead of using different visualizations for the sake of variety, one should *always* choose the visualization which best suits effective communication of the data in question [35].

Dashboards display information needed to achieve specific objectives. The data that is shown, must be relevant to the job at hand. Sometimes data needs to aggregated from multiple sources after which it can be tailored according to the context wherein it must be presented. This manner of processing data yields effective data visualizations.

A dashboard fits on a single computer screen. In order to see as much information at a glance, scrolling should be prevented. If multiple screens are present, then it is no longer a single dashboard. This leads to another question: what type of display is the information shown on? In today’s world, computer displays come in many shapes and sizes. Therefore, a responsive layout can be very beneficial, but scrolling becomes inevitable as the screen decreases in size. As an example, it is impossible to present the same dashboard built for a Full HD monitor on a smartphone display. While the latter display may have the same resolution, the content will be too small to read comfortably.

Dashboards are used to monitor information at a glance. Important data should be immediately noticeable, whereas specific details should be hidden. Therefore, by summarizing or aggregating the data a more effective visualization may be achieved. However, if the user wishes to view the detailed data, the dashboard should provide means to do so. Also, careful thought must be given to what information is of importance so it is never hidden.

To create an effective dashboard, the user-base must be well known and understood. What type of users are we dealing with? What are their characteristics?

These are important questions to ask, as one user may not comprehend one visualization while the other can. Therefore, the focus should be put on the user during the design of the dashboard [36]. We can ask the following questions to better understand the needs of the user:

- What metrics does the user need to see?
- What context does each metric require to make it meaningful? Do we need to visualize the variance, target to reach, trend. . . ?
- What visualization best communicates the metric?

On that end, sketches and mockups are helpful during the design process. Multiple iterations each incorporating feedback received from the users, ensure that the end result is satisfactory.

3.3 Personas & scenarios

The first step of the MuiCSer framework involves the creation of personas and scenarios. These artifacts should give us a better understanding of the end-users and how they will use the dashboard application. Each persona and scenario pair tries to highlight problems the users face and how the new system tries to solve them.

3.3.1 Jake

Persona Jake is a 25-year-old male who currently works as a nurse at the hospital in his city. He has been working there for four years and lives alone in his apartment. Still being a young adult, Jake grew up with technology. As a result, he experiences no difficulties when using new software on his computer or smartphone. His hobbies include music, playing the guitar, and video gaming.

Currently, the workflow at the hospital is dated. A new health information system was introduced to summarize and gather all medical data in a single program. Because this is an all-in-one system, a lot of features that Jake does not need still clutter the screen. Navigating the system is difficult and customization is not present. Jake wishes to only view the features he uses most while hiding the features he does not need.

Scenario Jake starts his first day using the new system by reading the manual that is accompanied with it. He starts the application which shows a list of patients for which he can create a personalized dashboard. When Jake opens the dashboard associated with a patient, he notices that many modules can be selected.

After adding a few modules to the dashboard, Jake has all the functionality he needs to do his work. During this process, he added a wrong module which was easily removed. Hereafter, Jake resizes and moves the modules until he is satisfied with the layout. Over time, Jake updated the dashboard of several patients by adding a module due to changes in his workflow. All he had to do was to open the module list and select the module he needed.

By examining the new module Jake quickly notices that it looks similar and is customizable in the same manner as the other modules. Jake feels he is in control of the system and convinced it will boost his productivity.

This scenario highlights the benefits of customization. Hiding unwanted and adding useful components allow for a clear dashboard to be displayed. The user is in control and is able to quickly view data of importance.

3.3.2 Dan

Persona Dan has been a general practitioner ever since he graduated from university. He is on the job for 21 years by now and the preferred doctor in his town. Throughout the years Dan used a multitude of systems and he is always on the lookout for better ones. Because he has been a general practitioner for such a long time, he has 500 patients that visit him at least twice a year. Some patients, especially the elderly, visit Dan monthly.

Most of Dan's patients visit for illnesses such as fever and a cold. To diagnose these illnesses, a description of the patient's symptoms suffices most of the time. Dan adds each diagnosis to the electronic health record of the patient. However, if a patient with a history of blood pressure issues visits, Dan has to perform a more complex diagnosis involving historical data. In the current system that Dan uses, it is very difficult to search for and work with this data. But what bugs Dan the most is that he has to do it every time this patient visits.

Scenario Dan tries a new health information system which allows customization for each specific patient. Because the diagnoses of most patients are different from each other, Dan creates a default module group which is displayed for each patient. This configuration is overridden if that patient already has a specific configuration. The default module group includes past diagnoses, known allergies, patient information (blood type, last weight, height…), and the medication the patient currently takes.

The first patient of the day describes what sounds like a fever. Dan confirms this and hands the patient a prescription. The diagnosis and prescription information are added to their respective modules. Dan did not need other health data to perform the diagnosis. Therefore, Dan does not change the configuration of this patient.

The next patient, an elderly woman, visited Dan for the second time this month. Dan knows from the past that it will probably be a heart related issue. Dan searches for a 'heart' module and adds it to the configuration of the elderly woman. Now both the default module group and a heart rate module are present, which is unnecessary according to Dan. He removes some modules of the default module group. Now, the next time the elderly woman visits, that configuration will be automatically loaded.

One specific patient had broken his arm three times in less than a year. When the patient came for a routine visit, Dan immediately added a module to easily view x-ray photos and view them in a timeline.

Providing customization options as described in this scenario has immediate benefits. Initially, it will take some time to configure each dashboard for all the patients, but the system will try to make this process quick to complete. Once the configuration is finished, time spent during a consultation can decrease dramatically.

3.3.3 Emily

Persona Just like Dan, Emily has been a successful general practitioner for quite some time. However, she has different needs of an EHR system. Between each patient visit, there is a period of time in which Emily does not know what happens to patients regarding certain parameters. For example, if a person regularly measures his heart rate and blood pressure because of cardiovascular disease, then it is important that the doctor is made aware of these values. If Emily notices a negative trend, she can contact the patient to schedule an early consultation.

In this case, Emily would like to visually see the progression of the heart rate and blood pressure values. Also, easy side-by-side comparison of both parameters would be useful. Currently, she needs to navigate from one screen to the other to do the comparison.

Scenario Emily recently received notice of a new platform that allows 3rd party applications to easily integrate with it. Now, a mobile application can send new values towards the platform. This allows Emily to monitor the patient remotely, which saves precious time for both parties. The platform can be customized to display multiple charts of data values.

A patient who recently had a cardiac arrest is continuing rehabilitation at home. However, the patient has chest pains and pays Emily a visit. She tells the patient to regularly measure his blood pressure and heart rate, and to take note of these values in a mobile application. After they have scheduled the next visit, the patient is sent home. As the next few weeks pass by, Emily notices a climbing trend on this patient's blood pressure chart. She tells herself to regularly check up on this patient as there is currently no reason to panic.

After another two weeks, Emily takes another look at the blood pressure data. The chart clearly shows that the blood pressure values keep rising. Emily decides to call the patient to schedule an early visit. The system helped Emily to intervene as soon as the situation seemed to worsen.

This scenario gives an example of how telemonitoring can be integrated into the dashboard platform. As mentioned in section 2.6, the onset of a disease can be detected when a caregiver can monitor the patient at home.

3.3.4 Anna

Persona Anna is a software engineer working at the local hospital. From the early 2000's, she was responsible for the integration of health software systems. As such, she has experienced the continuous change associated with health software firsthand. Each unit of the hospital uses different software, while they all share a central system to view patient records. The interoperability of these applications is one of Anna's responsibilities.

As time goes on, new software and medical instruments that improve certain workflows are purchased. Sometimes these replace old systems, otherwise they are added to work alongside them. Both are becoming increasingly difficult to do, as the codebase of the EHR system grows. When an older system gets replaced, Anna has to weed through the code and delete little pieces, hoping other parts wont break. It is a frustrating task as often band-aid solutions are applied.

Scenario A new EHR system was installed that should improve integration of other applications. Anna skims through the documentation and starts the transition process. She quickly notices that for each application she needs to create a back end data structure and a module that represents it in the EHR system. The new EHR system integrates new modules with minimal downtime.

After the initial transition period, everything is put into place. Not soon after, new bedside monitors with accompanying software were purchased. These replace the old bedside monitors and consequently, the software. As a first step, Anna reworked the back end data structure to work with the readings from the new device. The data of the old device is converted to the new data structure, so no data is lost. Anna designed the new module to be similar in appearance and functionality. After implementing the module, the staff barely notices any change and workflow resumes with minor disruption.

This persona and scenario highlight the importance of interoperability. Significant time and effort can be saved if this process is simplified. Also, maintenance is easier due to looser coupling of the modules.

3.4 Module selection process

In order to test our prototype, the dashboard needs to feature several modules. However, we want to provide general functionality and and avoid workflow-specific functionality. Literature was consulted to find general components which EHR systems often provide. Based on the several lists existing research gave us, a brainstorm session was conducted which determined the modules we will develop for the dashboard. First, we present the several lists of common components found in literature, whereafter we describe the purpose and the results of the brainstorm session.

3.4.1 Common components of EHR systems

Many different EHR systems exist and they all offer a different feature set. This gives health institutions a choice on which one to implement. However, each EHR system often provides some core features which are an absolute necessity in health care. Research listed many features of EHR systems. In this section we list the EHR system features found by each study, which lay the foundation of the brainstorm session.

Hillestad R., Bigelow J., Bower A. et al. (2005) provided a very general list of components [5]:

- Show information during order entry.
- Telemonitoring for chronic disease management.
- Alerts and reminders system.
- Workflow guidance.
- Reduce adverse drug effects.

Hsiao C., Hing E., and Ashman J. (2014) went into more detail compared to the previous list [37]:

- Computerized provider order entry for medications.
- Drug-drug and drug-allergy interaction checks.
- Generate and transmit permissible prescriptions electronically.
- Record patient demographics, including smoking status.
- Maintain up-to-date problem list of current and active diagnoses.
- Maintain active medication list.
- Maintain active allergy list.
- Record vital signs.
- Ability to generate an electronic copy of health information.
- Generate clinical summaries.
- Exchange key clinical information.
- Supports data privacy and security.

Elnahal S., Joynt K., Bristol S. et al (2011) went into even more detail. They revealed the following four categories of EHR components [4]:

- Clinical documentation: patient demographics, physician notes, nursing notes, problem lists, medication lists, discharge summaries, and advanced directives.
- Results viewing: lab reports, radiology reports, radiology images, diagnostic test results, diagnostic test images, and consultant reports.
- Computerized physician order entry: lab tests, radiology tests, medication, consultation requests, and nursing orders.
- Decision support: clinical guidelines, clinical reminders, drug-drug interaction alerts, drug allergy alerts, drug-lab interaction support, and drug dosing support.

To conclude, the **National Center for Research Resources (2006)** described the following key components and subcomponents [38]:

Figure 3: The affinity diagram created during the brainstorm session.

- Administrative system components: registration, admission, discharge, and transfer of patients, and links from the patient to resources such as observations, tests, procedures, evaluations, and diagnoses.
- Laboratory system components: integrate orders, results, schedules, and billing.
- Radiology system components: order data, interpretations data, patient data, imaging, patient tracking, patient scheduling, results reporting, and image tracking.
- Pharmacy system components: prescriptions filling and payment forms.
- Computerized physician order entry: service ordering, customized order sets, alerting, and results reporting.
- Clinical documentation: capture of clinical notes, patient assessments, clinical reports, and flow sheets of vital signs, in- and output, and problem lists.

3.4.2 Brainstorm session

This section describes the brainstorm session which intends to group the components described in the previous section to form categories. From these categories we pick several modules which we will then implement in our prototype. An ideal tool for this situation are affinity diagrams.

Affinity diagrams help to categorize and sort a large amount of ideas by grouping similar ideas together and identifying relevant topics [39]. These diagrams are created by following these five steps:

1. Generate ideas: all components of the lists mentioned the previous section serve as our ideas. Each component was written down on a post-it note. Duplicates between the lists are not skipped because we will get rid of these in later step.

2. Display the ideas: all post-it notes were placed on a table and shuffled around.

3. Sort the ideas into related groups: related components were grouped together and duplicates were removed.

4. Create header cards for the groups: these are the names of the categories. We found the following categories: medication, patient information, clinician-specific functions, and administration.

5. Draw the finished affinity diagram: figure 3 displays the result.

Result The following four categories were found:

- **Medication**: all components related to management of medications and prescriptions. These are: adherence, reduce adverse drug effects, drug-drug interaction alerts, drug-allergy interaction alerts, record patient medication list, order prescriptions for pharmacy, drug dosing support, and computerized medication order entry.

- **Patient information**: components that are related to the management of general patient data. These are: patient demographics, smoking status, patient history, medical history, record vital signs data (pulse, blood pressure, temperature, respiration), telemonitoring (cardiovascular disease, diabetes, COPD, cancer), allergy list, problem list, and a current diagnoses list.

- **Clinician-specific functions**: general components that a clinician uses, but are not related to a patient. These are: exchange clinical information, in- and external communication, view clinical summaries, generate alerts/reminders, record medical and follow-up notes, view medical guidelines, search for medical documentation, order and view lab results, highlight lab result abnormalities, and view imaging results.

- **Administration**: maintain patient list, bed planning, stock and warehouse management, emails, and calendar.

Some components such as reminder generation can be applied throughout an EHR system or be implemented as a sub-feature of another component. Literature repeatedly highlighted medication management as an important component of an EHR system, which is also the reason why it is a category. We chose to implement the following modules:

- Patient information: displays demographic information concerning an individual.
- Prescription list: create, update, and remove prescriptions, view the current medication scheme of a person, and provide warnings about potential drug interactions.
- Allergy list: create, update, and remove allergy entries.
- Vaccination list: create, update, and remove vaccination entries.
- Workflow: create workflow schedules for a patient. For example, performing checks at certain points in time can be defined here.
- History: view the medical history of a patient. This module records all changes made to the medical record of the patient.
- Telemonitoring: view data over time for the following parameters: blood pressure, blood sugar, heart rate, blood oxygen level, and weight.

3.5 Application specification

During this section we will describe the different components of the dashboard prototype. This specification should clearly indicate how the application is structured and how it will be used. First we describe the dashboard as a whole, after which we go into more detail for each module.

3.5.1 General structure

After startup, the user is greeted with a login screen. Once the user successfully authenticated, the patient list is shown. This list shows basic information of all the patients available to the user. It is possible to sort and filter this list. Selecting a patient will open the dashboard and load its corresponding configuration.

Dashboard The dashboard consists of two panels. A small panel on the left hand side features the patient information module. Smaller versions of some modules can be added to the panel which each generate a concise summary. Generating a summary is not relevant for every module, which is why some of them will not have smaller counterpart. The smaller modules can only be vertically arranged and resized. Due to the limited size of the left panel, these modules will take up the entire available width. In the case the display of the device is small, the left panel is automatically hidden to give more space to the main panel. Pressing a button in the top left corner will show the summarizing panel on top of the main panel. By supporting smaller screen sizes, the application can be used on mobile devices such as tablets.

The main panel fills up the remaining width of the display. Modules can be freely placed on this panel. Here, modules can be resized both horizontally and vertically. By providing resize capabilities for the modules, an even larger variety of display screens can be supported. If the screen width changes, then the modules will be rearranged to fit on the new display.

Layouting It is possible to create a module layout for each patient. Initially, the dashboard will be empty for all patients. Therefore, the user will need to add modules and arrange them on the dashboard. The layout automatically saves after each change. Dragging modules will push the others aside to give the user a preview of what will happen. Once the module is released, the layout is rearranged.

3.5.2 Modules

We now describe the functionality of each component chosen after the brainstorm session. For each module we describe its goal, functions, and whether it has a small version or not.

Patient information This module displays some basic demographic information of the patient. It is always found at the top of the left panel and it can not be removed, because this is information you should always have access to. However, it can be resized to a smaller version to give more space to other modules. We intend to show the following information for each patient: first and last name, gender, date of birth, national identification number, blood type, height, address, phone number, and smoking status. The small version shows only the full name, date of birth, gender, and blood type.

Prescription list The prescription list module shows an overview of all the patient's prescriptions during a certain period. The user can create, update, or remove prescriptions. In order to create a prescription, the user has to select a start and end date, a medicine, and the dosage. The dosage can be set for the morning, noon, evening, and bedtime. This is in line with weekly pill boxes that help patients adhere to their medication scheme. Interactions between prescriptions are highlighted if their time periods overlap. When the user clicks on a prescription, the following information regarding the medicine is shown: a description, side effects, and the medicines it interacts with.

A small version of this module shows the medication scheme of the patient of today. This list only shows the medicine and the dosage. Prescriptions can not be added, updated, or removed from this small version. The sole purpose of this module is to give a summary. As such, the detailed information of the prescriptions are hidden.

Allergy list The allergy module presents a list of all the allergies the patient has. This module allows the user to create, update, or remove allergy entries. The following information has to be entered for every allergy: name, description, diagnosis date, allergy type, and the severity. Based on literature, the following allergy types can be chosen: food, skin, respiratory, and drug allergy [40, 41]. In case an allergy does not fall within these four types, the user can indicate this. Allergies can be detected via IgE testing. The higher the IgE concentration, the more severe the reaction to the allergy. After consulting literature, five levels of severity can be selected ranging from 1 (mild) to 5 (severe) [42]. The cited source describes seven levels, with level 0 meaning allergy absence and level 6 meaning extremely severe. These two were omitted as absence to an allergy isn't recorded and level 5 also indicates an extreme severe reaction. This module also has a smaller version, which only shows the severity, name, and type of the allergies.

Vaccination list This module is similar to the allergy module, but instead stores vaccination entries. The user can add, update, or remove vaccinations. For each vaccination the disease name and description must be given. Optionally, a date of a future vaccination shot may be entered. Historical vaccinations shots can be added as sub-entries of the vaccination description. These entries contain a description and the date the shot was administered. The smaller version of this module will not show these entries, but only shows the diseases for which the patient is vaccinated and future vaccination data.

Workflow Clinicians need to follow certain procedures. The workflow module is a tool to describe these procedures. For example, a patient has received an artificial hip. During the rehabilitation process, the patient has to reach several goals over the next few weeks. A day after the operation, the patient must be able to sit up straight. The patient should be able to stand the next day. These goals can be written down in the workflow module.

Using this module, users can define workflows and share them if desired. For each new workflow the user must indicate whether it is created specifically for this patient and if the user wants to share the module with other users of the dashboard platform. A workflow consists of a series of steps, which in turn may have sub-steps. Each workflow has a name and a description. The steps of the workflow also have a name and description. However, sub-steps only have a description, as they should be small in scope.

History As the user adds, updates, or removes data from the record of the patient, log entries are generated for the history module. It is important to have a historic view of the patient's medical record, which is the primary goal of this module. Each log entry contains information on what changed, what type of operation was executed (create, update, delete), from which module it originated, and the date it happened. The user can filter the history entries to only show certain operations or modules where changes occurred. There is no small module present for this module, because this data is difficult to summarize.

Telemonitoring To conclude, the telemonitoring module will provide a visual display of vital signs data. The goal of this module is to help the user to identify trends and to compare data of different parameters with each other. The module will generate a chart on which the data is plotted given the selected time period. The following five parameters can be monitored: blood pressure, blood sugar, heart rate, blood oxygen, and weight. These parameters were chosen since they can be measured from home and are closely related to the four chronic diseases described in 2.6.

The user can choose to plot up to two parameters on the chart, in order to view their data in relation to each other. Comparisons between data from the same parameter, but for different time periods are realized by adding two instances of the module to the dashboard. For example, by placing them next to each other, differences are easily spotted. A small version of this module shows the number of data entries and statistics (lowest, highest, and average value) for the selected parameters from a chosen start date until now.

4 Implementation

This section describes the implementation process. The design highlighted the usage of the dashboard through its personas and scenarios, while the low fidelity mockups gave an early glance at its appearance. At its core, the strength of the application revolves around its extensibility and usability. Therefore, a lot of thought went into choosing the best libraries to provide these features.

First, an overview of the entire application is given, after which we go into more detail. The dashboard consists of two major parts common to web development. The back end is described first. This section covers explains the reasoning behind the chosen technology stack and the general structure. We also specify the back end structures and REST API calls for each module and component which the front end has access to. For the front end, we explain how the Polymer 3 library supports our modular design and its underlying principles. Hereafter, other libraries used during the implementation, the general front end structure, and the resulting dashboard and its modules are explained.

It is without question that during the implementation small changes were made to the design. These changes are documented for both parts. During the entire development process, only basic authentication was implemented. The implementation of standards, privacy rules, and security techniques do not contribute towards the goal of the dashboard we defined in the design section. Therefore, no effort was put in these topics.

4.1 Overview

The first major choice that was made, was the type of application to develop: a native application or a web application. It was a relative simple choice, but nonetheless an important one. Native applications are developed, in most cases, for one type of operating system. If a vendor wants to support both Android and iOS devices, two separate applications need to be developed. This requires more developers to not only create the application, but also to maintain it. Another drawback of native applications is the update process. This process often requires the reinstallation of the entire application, which is very disruptive and difficult to do in a care setting.

Web applications saw a surge in popularity. Compared to their native counterparts, web applications today are becoming increasingly similar in terms of functionality and design. Office Online from Microsoft is a good example showcasing this, which features many functions found on the desktop versions of the Office Suite. While these web applications also require updates, these happen on the server side. When the user refreshes the web page, the latest version of the application is automatically retrieved. This simplifies the roll-out process significantly. Also, all operating systems that support web browsers, are capable of running this type of applications. This includes mobile devices. As a result, developers only have to develop one application. However, the variation in screen resolutions and aspect ratios calls for a responsive design.

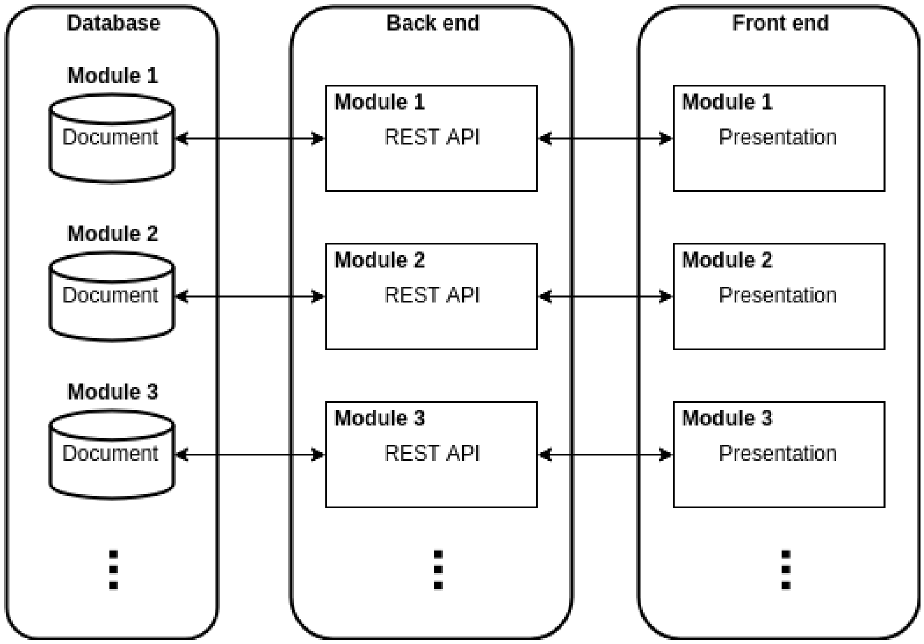

Figure 4: The communication flow by which every module communicates.

With this in mind, the choice to create a web application was evident. Users can view the dashboard regardless of the device they use, without sacrificing functionality. It should be noted that complex operations will not fit easily into a web application. The strength of web apps also leads to its major drawback: poor performance. A native application can draw much more computational power from the device it was made for. To transfer the computation burden to the server is also not a viable solution as this puts additional strain on the network. In case of real-time applications, latency becomes an issue due to continuous data transfer. However, the dashboard application should not experience any performance issues as the modules that were defined in section 3.5 do not require complex operations.

A typical web application consists of two parts: the back end and the front end. In software engineering principles, these refer to the separation of the data access layer and the presentation layer of the web application. For example, the back end fetches data requested by the front end. In turn, the front end presents the retrieved data towards the user. Due to the modular design of the dashboard, each module needs to have its own back and front end structures, visualized by figure 4. As the figure indicates, there is no direct interaction between the modules, which facilitates low coupling. The next sections describe the back end and front end in detail. Changes made to the design from section 3 are explained.

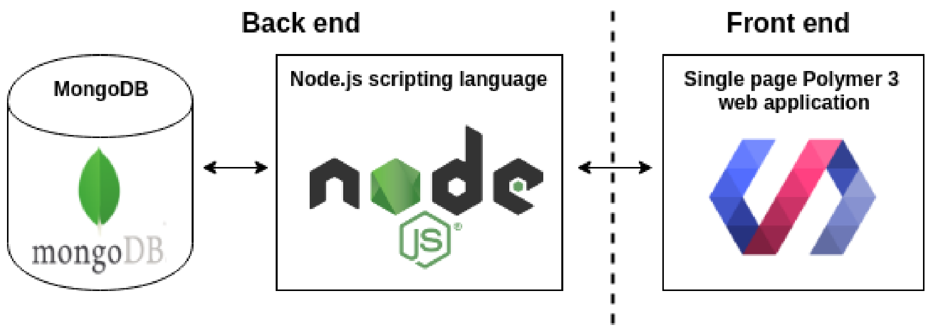

Figure 5: The three core technologies used to build the application.

4.2 Back end

The back end is responsible for everything related to the management of the data present in the dashboard application. It is here that the logic is defined to add, update, or remove records from the database based on the requests it received from the front end. The back end exposes a REST API which determines which requests the front end can send.

This section gives a detailed description of how the back end part of the dashboard was implemented. First, we go over the chosen technology stack after which we explain the back end structure. To close, each component present in the back end is briefly described.

4.2.1 Technology stack

Figure 5 shows the three core technologies used to develop the application. The back end was built using Node.js and MongoDB. We now give a brief explanation of what each back end component tries to achieve and why it was chosen.

4.2.1.1 Node.js

The back end server forms the heart of the dashboard application and Node.js was the preferred choice to implement it. Node.js (or simply Node) is a JavaScript runtime built on Google Chrome's V8 JavaScript engine [43]. Node.js avoids the classic thread-based approach of concurrency, which is relatively inefficient and can be very difficult to use. Instead, it provides concurrency via a single event loop that can support thousands of simultaneous connections, using non-blocking I/O calls. Node also has the added bonus of streamlining web development to the use of a single language.

The main reason Node.js was chosen, was due to the very large collection of packages that is available through the Node Package Manager (npm). For example, the Express package is only available via Node. This package was of significant help during the entire development process. Also, documentation is easily found thanks to the very large community that develops with Node.

4.2.1.2 MongoDB

A SQL database is table-based, whereas MongoDB is document-based. This type of structure enables highly flexible data schemas, as new data fields can be added without affecting existing rows. This is especially useful when a new data structure from a 3rd vendor needs to be added to the database. This supports the dashboard's goal of simple extensibility. Also, the developers of MongoDB officially support Node.js.

4.2.1.3 Node packages

Node allows developers to easily install a broad selection of packages. We now discuss the packages that were used for the development of the back end. These served many purposes, such as routing, encryption, logging, and request body parsing.

Mongoose Using Node without a library to communicate with the MongoDB is very difficult, as data validation and casting must be written manually. By installing the Mongoose, a MongoDB object modeling package, this will be taken care of. Via the package we can define data schemas and its parameters. Hereafter, a data model is created by compiling the schema. It is now possible to create MongoDB documents in a very similar fashion to object-oriented programming, which encapsulates the more complex inner workings of MongoDB. The entire database is constructed via these models.

Express Express is a Node.js web application framework, which helps with the management of routing, connecting middleware, handling requests and others. It helps to organize a web application on the server side into a MVC architecture. Also, Express makes the creation of a REST API server a much faster process compared to vanilla Node. Since we need to transfer data from MongoDB to our front end via a REST API, this framework is absolutely essential.

Bcrypt & JSON Web Tokens Basic encryption is provided by the bcrypt package. Whenever a user registers, the password is hashed and then stored into MongoDB via Mongoose. In case the user successfully authenticates, the JSON Web Tokens package generates a token based on the login name, user id, clearance level, and a secret phrase. This token is sent back to the front end and must be embed into every future request. After one hour, the token expires and the user must authenticate again.

Other The body-parser package helps as its name implies with parsing request bodies. This package is connected to Express as middleware, since it captures the request and parses it before it lands in the developer's hands. Morgan is another helpful package which logs all requests made to the server. As such, it is a helpful debugging tool. Last, but certainly not least, is the nodemon package. Nodemon detects changes in a Node.js project. If changes are found, nodemon restarts the application automatically to apply these changes.

4.2.2 Structure

As mentioned before, the Express framework can help with organizing a web application into a MVC architecture. The entire dashboard prototype is built around this architecture. The MVC architecture consists of three parts:

- Model: the part that manages the data and receives input from the controller. This is handled by Mongoose.
- View: the view presents the model or in other words, the data. The front end is our view, which is a Polymer 3 server.
- Controller: the controller handles incoming requests from the view. Hereafter it calls upon the model to get the desired data, which it then returns to the view. This is done on the back end Node.js server with the help of Express.

Every module present in the back end has three files of the same name, but each saved into the following folders: controller, models, and routes. They are connected as follows:

1. The front end sends a request to a certain route. Express checks this route and sees that the file `routes/patient.js` uses it.
2. `routes/patient.js` has every possible API endpoint mapped to a controller function which executes certain logic to retrieve data. All these controller functions are found in the `controller/patient.js` file.
3. Last, certain logic gets executed depending on which controller function is called. The controller calls the Mongoose model to perform certain actions on the database, which is defined in `models/patient.js`.
4. Depending on the operation, the controller can send data retrieved from the model back to the view. This transaction does not pass by the router.

In order to extend the dashboard application, these three files need to be created for each component. More concrete steps are given in appendix TODO.

Routing The routes defined in the back end follow a simple structure. All modules that do *not* retrieve patient data, start at the root endpoint: `/`. For example, to log in, the front end needs to send a POST request with the credentials stored in its body to the following endpoint: `/login`. In case a module does retrieve patient data, the endpoint is always prefixed with the patient id. Suppose the front end wants to retrieve all allergies of a patient, then it sends a GET request to the following endpoint: `/uniquepatientid/allergy/`. The file `routes/patient.js` will have an entry to forward all sub-routes that start with “allergy” to the routes defined in `routes/modules/allergy.js`.

4.2.3 Components and dashboard modules

Clinician

Layout

Import

Patient

4.2.3.1 Dashboard modules

Prescription

Allergy

Vaccination

Workflow

Checklist

History

Telemonitoring

4.3 Front end

The front end serves as the face of the dashboard application and is more complex than its back end counterpart. Throughout the lengthy implementation phase several were made, which required some work to be redone, but in the end promotes better extensibility and usability.

A modular dashboard was built using the Polymer 3 framework in combination with several libraries. The topics web components and shadow DOM are very important in realizing loosely coupled and highly cohesive modules, which is why it is important to understand them. After the used libraries are explained, the structure of the front end is described. To close, the end result is shown via various screenshots of the dashboard and its modules.

4.3.1 Polymer 3

Polymer is an open-source library that provides a set of feature for creating web components. These web components work similar to standards DOM elements such as `<h1>Title</h1>`. The library is developed by Google and contributors. Several Google services such as YouTube and Google Earth were developed with Polymer. We now explain two principles that lay at the heart of the library.

4.3.1.1 Web Components

HTML provides us with standard elements such as `h1` and `img`. Web Components allows the developer to create custom elements which are used in the same manner as the ones HTML provides. Suppose a calendar Web Component was created, a developer could add the element to any web page as follows: `<calendar></calendar>`. The actual definition of the calendar module resides in another file which is imported on the page where it will be used. This approach promotes the creation of reusable components, and above all, easy integration which is a primary goal of our dashboard application. At the heart of Web Components is the shadow DOM.

4.3.1.2 Shadow DOM

We assume the reader is familiar with the DOM concept. The shadow DOM can be seen as a scoped subtree inside a custom element. The root of this subtree is called the shadow root. The structure, styling, and behavior of the children within the shadow DOM do not affect any elements outside of it. Therefore, the appearance and functionality of the custom element is encapsulated and hidden at the document level. Only events fired by elements in the shadow DOM can be pickup outside the shadow DOM boundary. This idea of encapsulation is important for our approach towards the dashboard, as each module is responsible for its own appearance and functionality. Developers of custom modules should not need to worry about what happens outside of the shadow DOM.

4.3.2 Libraries

The dashboard must support resizing, and dragging & dropping of modules. To realize this, several libraries were compared to each other. After careful consideration, choices were made. During this section the libraries used to create the front end functionality are described.

Packery & Draggabilly Packery was chosen to create the grid in which the modules are placed. It features algorithms to minimize whitespace and gapless layouts. As items are added and removed from the grid, Packery reorders the grid. To make the items draggable, an additional library supported by Packery must be used, called Draggabilly.

Interact.js This library implements JavaScript drag and drop, resizing and multi-touch gestures for modern browsers. However, Draggabilly already provides drag and drop functionality. Since Interact.js is not supported by Packery, Draggabilly can not be removed. Therefore, Interact.js serves as the library that implements the resizing functionality for the modules.

Other moment c3

4.3.3 Structure

4.3.4 Dashboard

4.3.5 Modules

4.3.5.1 Prescription

4.3.5.2 Allergy

4.3.5.3 Vaccination

4.3.5.4 Workflow

4.3.5.5 Checklist

4.3.5.6 History

4.3.5.7 Telemonitoring

5 Usability test

As the prototype neared completion, a usability test was designed. As the name implies, the purpose of such a test is to assess the usability of a prototype. While the prototype is being tested by a user, the researcher takes note of any usability issues that may arise. This evaluation method results in valuable feedback, as it shows directly how real users use a system [22].

In this section we describe the design of a usability test to assess the user experience of the dashboard prototype. This test will focus on the usability aspect of the prototype, and not on the extensibility. However, in the future work section (7.1) we give suggestions to test this aspect. All the documents created for the test are found in appendix A.3.

5.1 Purpose

The goal of this test is to evaluate the usability of the dashboard prototype. The modules were designed with the usability principles mentioned in section 2.5 in mind. Customization was added by giving the user more freedom in defining the functionality present in the dashboard. However, the experience of the end-users will determine this is of added value when used in a clinical setting. Therefore, we are interested in gathering the following information:

- Is the dashboard easy to use? Is there a low or high risk of errors?
- Is the dashboard easy to navigate? Are all functions easily found?
- Does the dashboard as a whole succeed in displaying information in a clear and concise manner? How about the individual modules?
- Is the layout customization practical? Is it fast, slow, error-prone...?
- How does the usability of the prototype compare to the system the tester uses? Are there multiple systems installed? Are they available on mobile devices?
- Are there already customization capabilities present in the system that the tester uses?

Depending on the answers we receive to these questions, we can determine if it is feasible to continue the development or research of the dashboard application. While our primary concern is the evaluation of our prototype, we want to gain insight on the EHR system(s) the tester uses. This way, we can identify potential improvements or additional features for our prototype.

5.2 Participants

The prototype is designed for use in different medical settings, which are staffed by many different roles. To get the best possible results, we want our group of participants to cover as many different roles as possible. For example, a surgeon has completely different expectations from an EHR system compared to a nurse. This will yield feedback we would otherwise not have received if all participants had the same role.

What we need is variety. Our participants should represent a wide spectrum of the health care industry in terms of the roles they fulfill. On top of that, having participants from different age groups is encouraged. Ideally, we have the following sectors or roles represented by our participants:

- Laboratory unit: conducting tests and management of lab orders and results.
- Intensive care unit: requires close/real-time monitoring of several patients.
- General inpatient care: for example non-critical pediatric care within a hospital.
- Emergency response: for example a paramedic. Needs to respond fast, handle stressful situations, and needs to be mobile.
- A general practitioner: sees many patients daily with a large variety of conditions.

Now we know who we want to test, the next question is how many? Nielsen determined that five participants find almost as much usability problems as for example a group of ten [44]. In other words, five participants results in the optimal benefit-cost ratio associated with user testing. However, there are exceptions to this rule. A quantitative study for example should test at least 20 users in order to get statistically significant results.

Our test is qualitative in nature, since we want to get a deeper understanding of what the needs of the end-user are. We are interested in *why* the user experiences certain things. Therefore, according to Nielsen, we don't need to gather many participants, but preferably still more than five. This is due to the fact that the end-users of the dashboard have different professions and work in different care settings. The needs from each participant can vary greatly between these settings, which is why five participants may fall short in uncovering some important usability issues.

5.3 Test structure

During the recruitment, willing participants may choose when and where the test will take place, given the location will not cause any disturbances. The participants do not need to provide anything for the usability test. After meeting up, the following procedure will take place, with the estimated time for each step:

1. Pre-test, 5 to 10 minutes:
    (a) Greet and brief the participant.

(b) Ask for informed consent.
(c) The participant fills in a questionnaire which asks for demographic information and information concerning the EHR system he/she currently uses.

2. Test of the dashboard prototype, 20 to 25 minutes:
   (a) Give the participant a brief tour of the prototype.
   (b) The participant starts following the step-by-plan as he/she is being observed. No help or suggestions will be given unless necessary.
3. Post-test, 10 minutes:
   (a) The participant fills in a second brief questionnaire about the user experience.
   (b) The participant is interviewed. We go over suggestions, what was good/bad, future modules... Additional questions may be asked as the interview goes on.
   (c) Debrief the participant and hand out reward.

The informed consent form found in appendix A.3.1, is printed twice and filled in on the spot. The two questionnaires were created in Google Forms and are filled in using the laptop which has the prototype installed. Both questionnaires are found in appendix sections A.3.2 and A.3.3. Finally, the prototype is tested using the same laptop via the Google Chrome browser. A backup of the database is restored between each test to ensure that every participant will work with the exact same data.

5.4 Data gathering

The test receives quantitative data from the two questionnaires mentioned previously. However, the qualitative information is more important, which is gathered from the interview at the end of the test and from observing the participant during the test of the prototype.

Quantitative data The first questionnaire asks at the start for demographic information of which the following can be qualified a quantitative data:

- Age.
- Technology experience (Likert scale, 1: not experienced, 5: experienced).

The second part asks for information regarding the current EHR system that the participant uses:

- Satisfaction (Likert scale, 1: very dissatisfied, 5: very satisfied).
- User friendliness (Likert scale, 1: strongly disagree, 5: strongly agree).
- Supports the participant's needs (Likert scale, 1: strongly disagree, 5: strongly agree).

The second questionnaire asks for ratings regarding the usability of the prototype:

- General experience (Likert scale, 1: very bad, 7: very good). Because this is the general opinion of the participant, more steps are defined for this scale.
- Easy to use (Likert scale, 1: strongly disagree, 5: strongly agree).
- Dashboard looks clean (Likert scale, 1: strongly disagree, 5: strongly agree).
- Modules look clean (Likert scale, 1: strongly disagree, 5: strongly agree).
- Modules served as a good example (Likert scale, 1: strongly disagree, 5: strongly agree).
- Enough customization options present (Likert scale, 1: strongly disagree, 5: strongly agree).

No other quantitative data is gathered. During the observation, errors are noted, but not counted.

Qualitative data The bulk of the qualitative data is gathered during the test of the prototype and afterwards during the interview. When the participant is following the step-by-step plan, the following notes may be taken:

- The participant struggled finding UI element X.
- An error was made regarding X.
- The participant hesitated at step X.
- The participant asked question X regarding Y...

Questions related to the usability of the prototype and the current EHR system, current paper use in their care setting, module suggestions... are asked during the interview. Section 6 describes what qualitative data was retrieved. Also, the

5.5 Test scenario

During the step-by-step process, the participant works with the dashboard of three fictional patients. A short background is given for each patient and participant performs actions which relate to the situation of the patient. The participant has seen all aspects of the dashboard after the test of the prototype is complete. Appendix A.3.4 contains the step-by-step process. Also, the screenshots in appendix A.3.5 show the dashboard at the start of each scenario. We now describe each scenario.

Patient 1 Background: Kenny is an account manager of a large accounting firm. The combination of his stressful job, sedentary lifestyle, and smoking habit have lead to health issues. He has frequent palpitations and high blood pressure. At home, Kenny has to measure his weight, heart rate, and blood pressure on a regular basis with devices that send the values to his electronic patient record.

In this scenario the participant does the following:

1. Apply a filter on the history module.

2. Add and configure a telemonitoring module to fill the empty space of the dashboard.
3. Reorder the modules on the summarizing panel, while adding and configuring a small telemonitoring module.

Patient 2 Background: Since Jozefien retired, she has been gaining weight at an alarming rate. As a countermeasure, she regularly visits her general practitioner. Each consultation, the practitioner updates her medication scheme and tells her what she needs to pay attention to.

In this scenario the participant does the following:

1. Remove and add a prescription, check for interactions.
2. Remove the allergy module to replace it with a workflow module.
3. Edit some steps of a workflow.
4. Reset the checklist and add some new tasks to it.
5. Hide a parameter in the small telemonitoring module found in the left panel.

Patient 3 Background: After a serious car accident, Bert is hospitalized in the intensive care unit. Bert has many allergies, an elaborate medication scheme, and misses some vaccinations. Currently he is being treated by several clinicians and being observed by nurses. It is important that everyone is aware of these allergies, the medication scheme, and missing vaccinations. However, the data in the EHR system is not up to date. The latest most up to date medical data is still stored in a paper dossier.

In this scenario the participant does the following:

1. Change and add an allergy.
2. Remove and add a vaccination.
3. Apply a filter on the history module.

6 Discussion

The search for participants began immediately after the usability test design was completed. This section describes the recruitment process. Hereafter the findings of the usability tests are noted.

6.1 Recruiting participants

Recruitment started on the 11th of December, 2018. A total of 13 people were contacted which have the following background:

- 2 nursing students. All had internship experience in a hospital setting.
- 1 elderly care nurse.
- 2 intensive care unit nurses.
- 1 clinical laboratory department head.
- 1 paramedic.
- 2 home nurses, working for the same practice.
- 4 general practitioners, all work together under one practice.

Five individuals immediately were scheduled to test the prototype. The two home nurses and one nursing student declined due to time constraints. The paramedic expressed interest but was unreachable for two weeks until the start of the new year. The practice with the four general practitioners were contacted three times via their shared secretary, twice via phone and one time in person. The secretary promised each time to provide an answer the next day via email, but was never heard from.

As of writing, five participants have tested the prototype and one will test it the 5th of January. The other five tests were conducted between the 16th and 26th of December.

6.2 Findings

All conducted tests faced no issues regarding the application itself. Every participant tested the prototype in a secluded and silent environment. Three tests were conducted at the homes of the participants and the other two in empty classrooms of Hasselt University. The test was estimated to take approximately 40 minutes to complete. In anticipation of lengthy interviews, a hard stop was put in place in case the test passed the 60 minute mark. This was the case for two tests. When the test concluded, participants were rewarded with a bottle of white Sicilian wine with a price tag of €4,29. During the recruitment phase, each contacted individual was aware of a reward, but they did not know what it was beforehand.

Age	Gender	Profession	Experience	Tech experience (1-5)
21	M	Nursing student	4 years internship	4
25	F	Elderly care nurse	1,5 years	4
23	F	Intensive care unit nurse	1 year 5 months	4
23	F	Intensive care unit nurse	5 months	4
49	F	Laboratory department head	28 years	3

Table 1: Pre-test results: general participant information

EHR system	Mobile	Satisfaction (1-5)	User friendliness (1-5)	Complete toolset (1-5)	Custom-izable	# other systems
Orbis	No	3	3	3	No	0
GEMS	No	3	3	3	No	6
Metavision & GEMS	No	5	4	4	No	2+
ICCA	No	5	5	4	Yes	4
HIX	No	4	4	3	Yes	1

Table 2: Pre-test results: EHR systems in use by participants. Note: same order is respected as table 1.

6.2.1 Results

6.2.1.1 Pre-test questionnaire

Table 1 shows some general information from the pre-test questionnaire regarding the participants. It should be noted that 4 out of 5 participants are between the age 21 and 25. They also indicate a higher familiarity with technology, compared to the older participant. The younger participants also have limited work experience. The pre-test questionnaire also indicated that all participants use their smartphone and lap-/desktop daily. Three participants use their tablet a few times every month. One participant uses a smartwatch daily, and is the only one to use a smartwatch at all.

Table 2 shows that all participants use a different EHR systems for their care setting and none of them are used on a mobile device. Three out of five participants gave their system positive scores regarding user satisfaction, its user friendliness, and the set of features it provides. One was neutral and the other slightly negative. Only two EHR systems provide customization. Lastly, one participant uses no less than 6 applications next to the EHR system, while another participant uses none.

7 Conclusion

7.1 Future work

A Appendix

A.1 Nielsen's 10 usability heuristics

1. Visibility of system status: the system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
2. Match between system and the real world: the system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
3. User control and freedom: users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
4. Consistency and standards: users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
5. Error prevention: even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
6. Recognition rather than recall: minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
7. Flexibility and efficiency of use: accelerators unseen – by the novice user – may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.
8. Aesthetic and minimalist design: dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
9. Help users recognize, diagnose, and recover from errors: error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
10. Help and documentation: even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

A.2 Patient JSON import specification

A.3 Usability test documents

A.3.1 Informed consent: original (Dutch)

Note: the form was created in Microsoft Word and fit on exactly one page. The Hasselt University logo was located on the top right of the form.

Beste vrijwilliger,

Dit formulier dient om u informatie te geven over de studie waar u aan gaat deelnemen. Op het einde vragen we u dit formulier te ondertekenen om uw vrijwillige deelname te bevestigen. U kan te allen tijde uw deelname aan de studie intrekken. Een ondertekend kopie van dit document wordt aan u meegegeven.

De data verkregen van u en de observatie wordt geanonimiseerd en niet verder verspreid buiten de grenzen van deze studie.

Onderwerp: medisch dashboard met als focus per-patiënt personalisatie

Deze studie stelt een dashboard voor die een zorgverlener toelaat om per individuele patiënt de nodige functionaliteit samen te stellen. Door ons prototype van het dashboard te testen, kunnen we de sterktes en zwaktes ervan identificeren. We testen het prototype en niet u, de gebruiker. U kunt dus niets verkeerd doen tijdens deze test.

Het verloop van deze test bestaat uit drie delen:

1. Vragenlijst betreffende persoonlijke gegevens en gebruikte medische software (5 à 10 min).
2. Test van het prototype met behulp van een stappenplan (20 à 25 min).
3. Interview en een vragenlijst om de gebruikerservaring te toetsen (10 min).

Voordat de test van het prototype van start gaat, krijgt u een korte uitleg omtrent het gebruik van het dashboard. Zodra de test begonnen is, wordt u geobserveerd. Indien er problemen of onduidelijkheden ontstaan mag u altijd vragen stellen. Tenslotte wordt u aangeraden om luidop te denken.

Verklaring van toestemming

Handtekening testpersoon:	Handtekening getuige:
Naam: ____________________	Naam: ____________________
Datum: __________	

Contactgegevens: Naam: Dennis Cardinaels
Email: dennis.cardinaels@student.uhasselt.be
GSM: 0479 51 47 19

A.3.2 Pre-test questionnaire: original (Dutch)

Description: In de eerste sectie worden enkele demografische gegevens van u verzameld. Daarna vragen we informatie omtrent het EPD (elektronische patiëntendossier) dat momenteel in gebruik is.

Persoonlijke informatie

- Geslacht*[1]: ○ Male ○ Female
- Leeftijd*: ______
- Wat is uw huidige jobtitel?* ______________________
- Hoe lang werkt u al binnen de gezondheidszorgsector?* __________
- Hoe ervaren bent u met computers?*

  Onervaren ○ ○ ○ ○ ○ Ervaren
- Welke apparaten gebruikt u dagelijks?*

  ☐ Computer/laptop ☐ Tablet
  ☐ Smartphone ☐ Andere: __________
- Welke apparaten gebruikt u enkele malen per week?*

  ☐ Computer/laptop ☐ Tablet
  ☐ Smartphone ☐ Andere: __________
- Welke apparaten gebruikt u enkele malen per maand?*

  ☐ Computer/laptop ☐ Tablet
  ☐ Smartphone ☐ Andere: __________

Elektronisch patiëntendossier (EPD)

- Wat is de naam van het huidige EPD?* ______________________
- Wordt het EPD op mobiele apparaten (smartphone/tablet) gebruikt?*

  ○ Ja ○ Nee
- Hoe lang is dit EPD in gebruik? (indien u dit niet weet, laat open) ______
- Hoe tevreden bent u van het EPD?*

  Zeer ontevreden ○ ○ ○ ○ ○ Zeer tevreden
- Het EPD is gebruiksvriendelijk.*

  Helemaal niet akkoord ○ ○ ○ ○ ○ Helemaal akkoord
- Het EPD voldoet aan al mijn eisen.*

  Helemaal niet akkoord ○ ○ ○ ○ ○ Helemaal akkoord

[1] A star indicates a mandatory question.

- Welke functies ontbreken in het EPD? ________
- Het EPD kan aangepast worden naar eigen wens.* ○ Ja ○ Nee
- Indien 'ja' op de vorige vraag, in welke mate? Is dit bijvoorbeeld per patiënt mogelijk? ________
- Worden andere programma's in combinatie gebruikt met het EPD?*
  ○ Ja ○ Nee
- Indien 'ja' op de vorige vraag, welke programma's? ________

A.3.3 Post-test questionnaire: original (Dutch)

Description: De volgende vragen toetsen uw ervaring met het prototype. Indien u nog extra opmerkingen hebt die niet in de enquête genoteerd kunnen worden, mag u dit altijd melden.

Persoonlijke ervaring

- Algemene ervaring met het dashboard.*
  Zeer slecht ○ ○ ○ ○ ○ ○ ○ Zeer goed
- Het dashboard is makkelijk om te gebruiken.*
  Helemaal niet akkoord ○ ○ ○ ○ ○ Helemaal akkoord
- Het dashboard ziet er duidelijk uit.*
  Helemaal niet akkoord ○ ○ ○ ○ ○ Helemaal akkoord
- De modules zien er duidelijk uit.*
  Helemaal niet akkoord ○ ○ ○ ○ ○ Helemaal akkoord
- Alles is makkelijk terug te vinden in het dashboard.* ○ Ja ○ Nee
- Indien 'nee' op de vorige vraag, wat was moeilijk om terug te vinden? ________
- De modules dienden als een goed voorbeeld wat mogelijk is met het dashboard.*
  Helemaal niet akkoord ○ ○ ○ ○ ○ Helemaal akkoord
- Heeft u suggesties voor potentiële modules? ________
- Er zijn voldoende personalisatie mogelijkheden aanwezig in het dashboard.*
  Helemaal niet akkoord ○ ○ ○ ○ ○ Helemaal akkoord
- Wat vond u goed/niet goed aan de personalisatie? ________

A.3.4 Step-by-step plan: original (Dutch)

Note: the step-by-step plan was created in Google Docs and fit on exactly two pages.

Start

1. Log in als:
   (a) Username: “verpleger1”.
   (b) Wachtwoord: “dashboard”.
2. Selecteer patiënt “Kenny Martens”.

Patiënt Kenny Martens
Achtergrond: Kenny is een account manager van een grote boekhouding firma. Zijn stressvolle job, sedentaire levensstijl en gewoonte om te roken hebben gezondheidsproblemen met zich meegebracht. Zo heeft hij regelmatig last van hartkloppingen en een hoge bloeddruk. Thuis moet Kenny regelmatig zijn gewicht, hartslag en bloeddruk meten met apparaten die de waardes doorsturen naar het elektronisch patiëntendossier.

Stappen:

1. Bekijk het dashboard en open en sluit het zijpaneel.
2. Pas een filter toe op de historiek van de patiëntendata:
   (a) Zet de startdatum op een week geleden en de einddatum op vandaag.
   (b) Selecteer enkel de voorschrijvingen en de telemonitoring modules.
   (c) Laat alle operaties geselecteerd staan.
3. Voeg een telemonitoring module toe dat de resterende breedte en hoogte van de pagina inneemt. Stel in:
   (a) Toon bloeddrukwaarden en plot ze op de y-as.
   (b) Selecteer data vanaf vandaag tot ongeveer een maand geleden.
   (c) Bekijk de waarden van enkele datapunten op de lijngrafieken.
   (d) Beschrijf de trend van de data op beide grafieken.
4. Open het zijpaneel en maak het patiënteninformatie moduletje kleiner.
5. Voeg een kleine telemonitoring module toe aan het zijpaneel:
   (a) Selecteer de bloeddruk, hartslag en gewicht als parameters.
   (b) Toon een samenvatting van de laatste 14 dagen.
6. Sleep de telemonitoring module zodat deze boven de huidige medicatie module staat.
7. Maak de webbrowser breder totdat het zijpaneel openklapt en ga naar de patiëntenlijst.
8. Selecteer patiënt “Jozefien Hendrix”.

Patiënt Jozefien Hendrix
Achtergrond: Jozefien is een pas gepensioneerde dame die al heel haar leven

zwaarlijvig is. Nu ze pas op pensioen is gegaan, is ze alarmerend aan het bijkomen. Om dit tegen te gaan, gaat ze regelmatig op controle bij de huisarts. Bij elke raadpleging past de huisarts haar medicatieschema aan en daarbovenop geeft hij suggesties voor waar ze op moet letten.

Stappen:

1. Bekijk de details van enkele voorschriften uit de lijst.
   (a) Verwijder het voorschrift voor "Furosemide".
   (b) Voeg een nieuw voorschrift toe:
      i. Selecteer medicijn "Simvastatine" en een dosering naar keuze.
      ii. Zet de periode van het voorschrift van begin december tot eind december.
      iii. Zijn er interacties? Zo ja, welke?
2. Verwijder de allergie module en voeg in de plaats een workflow module toe.
   (a) Laadt workflow "Consultatieprocedure" in.
   (b) Verwijder de eerste stap "Nieuwe patiënt".
   (c) Voeg een nieuwe substap "Medicatieschema controleren" toe aan de "Pre-consultatie" stap.
3. Reset de checklijst.
   (a) Voeg de taak "Medicatie voorschrijven" toe en vink deze aan.
   (b) Voeg aan de taak "Metingen uitvoeren" een subtaak "Gewicht" toe.
4. In het zijpaneel, verberg de zuurstof parameter in de telemonitoring module.
5. Ga naar de patiëntenlijst en selecteer patiënt "Bert Nieuwenhuize".

Patiënt Bert Nieuwenhuize
Achtergrond: Na een zwaar auto ongeluk, ligt Bert op de intensieve zorgafdeling. Bert heeft een hele waslijst aan allergieën, medicatie dat hij inneemt en is niet helemaal up-to-date met zijn vaccinaties. Momenteel wordt hij behandeld door meerdere dokters en geobserveerd door verpleegkundigen. Het is dus belangrijk dat iedereen op de hoogte is van Bert zijn allergieën, huidige medicatie, enzovoort. Echter zijn de gegevens in het EPD onvolledig, vergeleken met het papieren dossier dat het ziekenhuis in handen heeft.

Stappen:

1. Bij de allergie module, doe het volgende:
   (a) Verander de reactiegraad van de "koeienmelk" allergie naar niveau 1.
   (b) Voeg een "Latex" allergie (type "Other") toe met beschrijving "Jeuk, last van luchtwegen, niezen", reactiegraad 5 en de datum op vandaag.
2. Bekijk enkele inentingsdata van vaccinaties in de vaccinatiemodule.
   (a) Verwijder de "test" vaccinatie.

(b) Voeg een nieuwe vaccinatie "Hepatitis B" met beschrijving "Leverontsteking" toe.
    i. Kies 12 januari als volgende inentingsdatum.
    ii. Voeg een nieuwe historische vaccinatie toe met als beschrijving "1ste vaccinatie, patiënt was 12 jaar oud" en de datum 11 april 1990.

3. In de historiek van de patiëntendata, pas de volgende filter toe:
    (a) Zet de startdatum op 3 dagen geleden en de einddatum op vandaag.
    (b) Selecteer de allergie en vaccinatie module.
    (c) Selecteer alle operaties buiten "import".
    (d) Bekijk enkele details van de historiek.

A.3.5 Screenshots taken at the start of each scenario

References

[1] Edward H. Shortliffe and James J. Cimino. *Biomedical informatics: Computer applications in health care and biomedicine: Fourth edition.* 2014.

[2] Paul C. Tang, Joan S. Ash, David W. Bates, J. Marc Overhage, and Daniel Z. Sands. Personal health records: Definitions, benefits, and strategies for overcoming barriers to adoption, 2006.

[3] Jason J. Saleem, Alissa L. Russ, Connie F. Justice, Heather Hagg, Patricia R. Ebright, Peter A. Woodbridge, and Bradley N. Doebbeling. Exploring the persistence of paper with the electronic health record. *International Journal of Medical Informatics*, 78(9):618–628, 2009.

[4] Shereef M. Elnahal, Karen E. Joynt, Steffanie J. Bristol, and Ashish K. Jha. Electronic health record functions differ between best and worst hospitals. *American Journal of Managed Care*, 2011.

[5] Richard Hillestad, James Bigelow, Anthony Bower, Federico Girosi, Robin Meili, Richard Scoville, and Roger Taylor. Can electronic medical record systems transform health care? Potential health benefits, savings, and costs. *Health Affairs*, 24(5):1103–1117, 2005.

[6] Kristiina Häyrinen, Kaija Saranto, and Pirkko Nykänen. Definition, structure, content, use and impacts of electronic health records: A review of the research literature. *International Journal of Medical Informatics*, 77(5):291–304, 2008.

[7] H. J. Tange. Consultation of medical narratives in the electronic medical record. *Methods of Information in Medicine*, 38(4-5):289–293, 1999.

[8] Nir Menachemi and Taleah H. Collum. Benefits and drawbacks of electronic health record systems. *Risk Management and Healthcare Policy*, 2011.

[9] Agency for Healthcare Research and Quality. Your guide to choosing quality health care. *Rockville, MD: U.S. Department of Health and Human Services, Agency for Healthcare Research and Quality*, 2001.

[10] Institute of Medicine Committee on Quality Health Care in America. *Crossing The Quality Chasm : a New Health System for the 21st Century.* Washington, D.C.: National Academy Press, 2001.

[11] Nir Menachemi, Thomas L. Powers, and Robert G. Brooks. The role of information technology usage in physician practice satisfaction, 2009.

[12] Nir Menachemi. Barriers to ambulatory EHR: Who are 'imminent adopters' and how do they differ from other physicians? *Informatics in Primary Care*, 2006.

[13] Emily M. Campbell, Dean F. Sittig, Joan S. Ash, Kenneth P. Guappone, and Richard H. Dykstra. Types of Unintended Consequences Related to Computerized Provider Order Entry. *Journal of the American Medical Informatics Association*, 2006.

[14] Ross Koppel, Joshua P. Metlay, Abigail Cohen, Brian Abaluck, A. Russell Localio, Stephen E. Kimmel, and Brian L. Strom. Role of computerized physician order entry systems in facilitating medication errors. *Journal of the American Medical Association*, 2005.

[15] T. Payne, J. Fellner, C. Dugowson, D. Liebovitz, and G. Fletcher. Use of more than one electronic medical record system within a single health care organization. *Applied Clinical Informatics*, 2012.

[16] IEEE. *IEEE Standard Computer Dictionary. A Compilation of IEEE Standard Computer Glossaries.* 1990.

[17] Tim Benson and Grahame Grieve. *Principles of Health Interoperability: SNOMED CT, HL7 and FHIR.* Springer International Publishing, 2016.

[18] Health Level Seven International. Health Level 7. `http://www.hl7.org/`. Accessed: 2018-12-07.

[19] Peter Mildenberger, Marco Eichelberg, and Eric Martin. Introduction to the DICOM standard, 2002.

[20] Nigel Bevan. International standards for HCl and usability. In *International Journal of Human Computer Studies*, 2001.

[21] Paula J. Edwards, Kevin P. Moloney, Julie A. Jacko, and François Sainfort. Evaluating usability of a commercial electronic health record: A case study. *International Journal of Human Computer Studies*, 66(10):718–728, 2008.

[22] Jakob Nielsen. *Usability Engineering.* 1993.

[23] Blackford Middleton, Meryl Bloomrosen, Mark A. Dente, Bill Hashmat, Ross Koppel, J. Marc Overhage, Thomas H. Payne, S. Trent Rosenbloom, Charlotte Weaver, and Jiajie Zhang. Enhancing patient safety and quality of care by improving the usability of electronic health record systems: Recommendations from AMIA. *Journal of the American Medical Informatics Association*, 2013.

[24] Marilyn J Field et al. *Telemedicine: A guide to assessing telecommunications for health care.* National Academies Press, 1996.

[25] World Health Organisation. Global status report on noncommunicable diseases 2014. Technical report, 2014.

[26] S. Mendis, P. Puska, and B. Norrving. Global atlas on cardiovascular disease prevention and control. *World Health Organization*, 2011.

[27] National Center for Health Statistics. Health, United States, 2015: With Special Feature on Racial and Ethnic Health Disparities., 2015.

[28] Jessie Gerteis, David Izrael, Deborah Deitz, Lisa LeRoy, Richard Ricciardi, Therese Miller, and Jayasree Basu. Multiple Chronic Conditions Chartbook. *AHRQ Publications No, Q13-0038*, 2014.

[29] Walter C. Willett, Jeffrey P. Koplan, Rachel Nugent, Courtenay Dusenbury, Pekka Puska, and Thomas A. Gaziano. *Prevention of Chronic Disease by Means of Diet and Lifestyle Changes.* 2006.

[30] Stephane Meystre. The Current State of Telemonitoring: A Comment on the Literature. *Telemedicine and e-Health*, 2005.

[31] Marci Meingast, Tanya Roosta, and Shankar Sastry. Security and privacy issues with health care information technology. In *Annual International Conference of the IEEE Engineering in Medicine and Biology - Proceedings*, 2006.

[32] Mieke Haesen, Karin Coninx, Jan Van Den Bergh, and Kris Luyten. MuiCSer: A process framework for multi-disciplinary user-centred software engineering processes. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008.

[33] Sabine Madsen and Lene Nielsen. Exploring persona-scenarios - Using storytelling to create design ideas. In *IFIP Advances in Information and Communication Technology*, 2010.

[34] Stephen Few. *Information dashboard design: The effective visual communication of data.* 2006.

[35] Stephen Few. Intelligent Dashboard Design. *DM Review*, 2005.

[36] Richard Brath and Michael Peters. Dasboard Design: Why Design is Important, 2004.

[37] Chun-Ju Hsiao, Esther Hing, and Jill Ashman. Trends in electronic health record system use among office-based physicians: United States, 2007-2012. *National health statistics reports*, 2014.

[38] National Center for Research Resources. Electronic Health Records Overview. 2006.

[39] Balanced Scorecard Institute. Affinity Diagram. `https://www.balancedscorecard.org/portals/0/pdf/affinity.pdf`. Accessed: 2018-12-22.

[40] American College of Allergy, Asthma, and Immunology. Types of Allergies. `https://acaai.org/allergies/types`. Accessed: 2018-09-26.

[41] Health24. What are the different types of allergies? https://www.health24.com/Medical/Allergy/Overview/what-are-the-different-types-of-allergies-20180313. Accessed: 2018-09-26.

[42] East Kent Hospitals University. Allergy Diagnosis Reference Guide. https://www.mtw.nhs.uk/wp-content/uploads/2015/08/Allergy_diagnosis_reference_guide.pdf. Accessed: 2018-09-26.

[43] Node.js Foundation. About Node.js. https://nodejs.org/en/about/. Accessed: 2018-12-28.

[44] Jakob Nielsen. How Many Test Users in a Usability Study?, 2012.