**NTNU – Trondheim**
Norwegian University of
Science and Technology

# NoFall: A Platform for Mobile Applications Focused on Fall Prevention

Finn Terje Johansen

Master of Science in Informatics
Submission date: November 2014
Supervisor: Monica Divitini, IDI
Co-supervisor: Babak A. Farshchian, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Norges teknisk-naturvitenskapelig universitet

Master Thesis

# NoFall: A platform for mobile applications focused on fall prevention

*Author:*
Finn Johansen

*Supervisor:*
Monica Divitini
*Co-Supervisor:*
Babak A. Farshchian

*A thesis submitted in fulfilment of the requirements for the degree of Master in Informatics, Software*

November 2014

NORGES TEKNISK-NATURVITENSKAPELIG UNIVERSITET

# *Abstract*

Master in Informatics, Software

**NoFall: A platform for mobile applications focused on fall prevention**

by Finn Johansen

BACKGROUND: Fall related research is a major field of research, and different solutions emerge rapidly. Many of them are technical and focus on mobile technology. To be able to develop high-level applications of good quality rapidly, well-defined and well-structured back-end platforms are crucial elements. There exists a gap in a well-defined platform that provides back-end data-storage functionality, Such a platform will help developers to focus on the front-end; making it easier to develop high-level applications quicker and of higher quality. OBJECTIVE: The major objective of this study is to develop a data-storage platform, called NoFall, to ease the development process of fall preventative high-level applications. This will decrease required development time and enable development teams to focus more on functionality and user interface. NoFall will be developed for Android devices. RESEARCH METHOD: First, this thesis investigates the current research in fall detection and prevention. Second, State-of-the-Art solutions are examined and discussed. Finally, requirement specifications are formed based on the problem elaboration and usage scenarios. RESULTS: The resulting platform has shown, through the development of test applications, that this is a step in the right direction for the increased quality of high-level applications targeting fall prevention. Due to the fact that the result is a proof of concept, some important future improvements are identified, such as improvement of the database, making the platform easier to use - and supporting more usage scenarios.

**Keywords**: database, platform, rapid prototyping, fall prevention, fall detection, android, high-level application, mobile application, nofall

# *Sammendrag*

BAKGRUNN: Forskning relatert til fallulykker blant eldre er et stort forskningsfelt, og ulike løsninger utvikles hele tiden. Mange av dem er teknologiske og fokuserer på mobilteknologi. For å være i stand til å utvikle høynivå-applikasjoner av god kvalitet hurtig, er veldefinerte og velstrukturerte back-end plattformer viktige elementer. Plattformsløsninger som har fokus på back-end datalagringsfunksjonalitet er i dag en mangelvare. Slike plattformer vil gjøre det enklere å ha større fokus på utviklingen av applikasjonenes front-end, noe som fører til at det blir enklere å utvikle høynivå-applikasjoner raskere og med høyere kvalitet. MÅL: Målet med masteroppgaven er å utvikle en datalagringsplattform - NoFall - som skal være med på å forenkle utviklingsprosessen av mobile høynivå-applikasjoner for falldetektering og fallforebygging. Dette vil redusere utviklingstiden og gjøre det mulig å ha større fokus på funksjonalitet og brukergrensesnitt innenfor et utviklingsprosjekts rammer. NoFall vil bli utviklet for Androidbaserte enheter. Kravspesifikasjonen er utformet i henhold til funnene i problemanalysen og brukerscenariene. FORSKNINGSMETODE: Oppgaven starter med en litteraturstudie som danner grunnlaget for en analyse av problemområdet. Deretter vil flere av de mest anerkjente løsningene innen problemområdet bli gjennomgått og diskutert. Til slutt vil en kravspesifikasjon bli utformet i henhold til problemanalysen og brukerscenariene. RESULTAT: Den resulterende plattformen har vist gjennom utvikling av testapplikasjoner å være et steg i riktig retning for økt kvalitet av høynivå-applikasjoner rettet mot fallforebygging. Siden resultat er et "proof of concept" er flere viktige videreutviklingssteg identifisert, som for eksempel forbedringer av database slik at den støtter flere bruksscenarier.

Nøkkelord: database, plattform, rask prototyping, android, falldetektering, fallforebygging, høynivå-applikasjoner, mobile applikasjoner, NoFall

# *Acknowledgements*

I would like to thank Monica Divitini for her supervision of this project, and Babak A. Farshchian for his engagement, guidance and feedback throughout the entire project and the opportunity to gain insight in an exiting field of research. I would also like to thank Helge Johansen for support and feedback through the dissertation.

# Contents

# List of Figures

# List of Tables

# Abbreviations

| | |
|---|---|
| **HCI** | **H**uman-**C**omputer **I**nteraction |
| **OS** | **O**perating **S**ystem |
| **SPINE** | **S**ignal **P**rocessing **I**n **N**ode **E**nvironment |
| **WBSM** | **W**ireless **B**ody **S**ensor **N**etwork |
| **ADAPT** | **A**ssess: **D**isorientation, **A**ctivity, **P**ost medication, and **T**oileting |
| **API** | **A**pplication **P**rogramming **I**nterface |
| **IRR** | **I**ncidence **R**ate **R**atio |
| **DB** | **D**ata**b**ase |
| **PPP** | **P**urchasing **P**ower **P**arity |
| **SDK** | **S**oftware **D**evelopment **k**it |
| **GUI** | **G**raphical **U**ser **I**nterface |
| **IDE** | **I**ntegrated **D**evelopment **E**nvironment |
| **URI** | **U**niform **R**esource **I**dentifier |

# Chapter 1

# Introduction

## 1.1 Problem Definition

**Back-end Data-Storage Platform for developers to develop high-level mobile applications in the field of fall prevention**

In the field of fall prevention and detection, the usage of mobile solutions is growing. The benefits of mobile applications are that they are relatively cheap to develop, and the development process is short. Smart-phones with touch-screens allows for simple and intuitive interaction, which has a small learning curve, making them ideal for use with the elderly.

To be able to develop applications of high quality in functionality and user interface (referred to as high-level applications) rapidly, both the back-end and the front-end are equally important. Several frameworks for the development of mobile applications, such as SPINE[32] are available to the developers. There exists however a gap in a well-defined platform that provides back-end data-storage functionality, that enables the developers to focus on the front-end; making it easier to develop applications quicker and of higher quality. The structure of the back-end data-storage platform will be based on state of the art knowledge of which patient information is crucial for predicting the level of risk of falling. This master thesis will explore different models for such a platform based on the hypothesis:

**Hypothesis**

Having a well structured back-end data-storage platform for fall detection applications will make it easier for future developers to build high-level applications that are well defined and achieve a higher level of quality.

The expected deliverables are:

- An analysis of fall related research for the elderly
- A State-of-the-Art analysis of existing solutions and technologies
- A data-storage platform for the development of high-level applications based on requirement specification and architecture design; the platform will consist of:
  - A modular database for fall preventative applications
  - API for accessing the database
- A set of example applications demonstrating the platform

The platform will be developed for Android and released under the open source Apache 2.0 license.

## 1.2 Problem Description

The elderly are at a high risk of falling and injuring themselves. Therefore having systems to prevent falls, and reduce the likelihood of them occurring is important. Considering several factors such as movement patterns, medication, physical activity levels, one can try to determine if the person is at a high risk of falling.

The purpose of this dissertation is to develop an Android data-storage platform that supports different sources of input, providing standardization to make it easier to collaborate results, and an extensive and well-defined database. Examples of applications the platform will support are data gathering (e.g. pedometers), surveys (Q & A), tests (Time up and Go test) with many more. The results of this is the hope of reducing development time and increase the quality of the product, making it possible for the developers to focus on high-level implementation. Such a solution can be beneficial for the elderly and for the health service, as well as researchers and developers.

To summarize, the platform is an attempt to improve these issues:

- Time of development - Reduce the time by reuse of a code base
- Quality of Solutions - Both technical and functional
- Interoperability - Enable applications to work together and share the same database

## 1.3 Motivation

Due to the fact that the rate of fall accidents among the elderly population is high and have a high economic impact on society as well as causing significantly reduced quality of life for these persons it is of very high importance to develop knowledge and systems to prevent falls. The use of mobile applications as tools to perform this task is growing steadily, moving away from paper surveys and testing performed with a healthcare professional.

In all research and development activities there are limited availability of resources, in terms of money, time, and personnel. This thesis will focus on investigating and developing a solution that will work as a platform for high-level application development and thereby providing a well-defined back-end solution that will adapt and work with several usage scenarios, fitting general cases as well as niche cases.

## 1.4 Research Questions

The dissertation posed several research questions to help focus and facilitate the direction.

**Main Research Question**
What is the effect of a data-storage platform on the ease and quality of developed applications?

*How it is answered in the thesis:*
The platform has several effects that have been identified through the work on this thesis:

- It reduces the number of lines of code that has to be written. This translates to shorter development time, or more time to be used on improving other parts of the application.
- The need to research and design the back-end is removed, leading to shorter time to delivery.
- It is easy to adapt existing solutions to the platform. This is important, so that existing solutions do not have to go through a big overhaul to adapt to the platform.
- The data that is stored in the DB can easily be shared between different applications. This makes collaboration easier and more accessible.

### Secondary Research Questions

#### Question one

What are common and easily measurable risk factors used in fall risk assessment?

*How it is answered in the thesis:*

From the literature analysis (see chapter 3 for details) several factors where identified - a short summary of some of the important ones are listed below:

- Medication: What type of medication (e.g. antidepressant) and how many different types of medication the person takes will affect fall risk
- Poor strength and balance: If the person is physically weak - the risk of falling will increase. The implication of the fall will also be more severe with low physique.
- Psychological: The state-of-mind of the person will affect the risk of falling (e.g. fear of falling, nervousness).

#### Question two

How can one design a meaningful data model based on the knowledge of which patient information/data is useful for predicting the level of risk of falling?

*How it is answered in the thesis:*

The most important aspects of modelling data, is identifying what kind of data that will be stored - and how it relates to each other. Based on the analysis done in chapter

3, the sort and structure of data typically gathered was identified. This analysis also helped to form relations between the different data, seeing how they relate to fall risk.

With the knowledge of what kind of data and how it should relate to each other, the design and architecting could begin. The database follows best practices and guidelines from the SQLite homepage. See chapter 7 for the full detailed design of the database.

**Question three**

How should the technical architecture of a platform be designed and built?

*How it is answered in the thesis:*

The design and architecture of a platform is slightly different from normal application design. The focus is on the developers and not end-users.

The analysis done in chapter 4 showed how other researchers and developers had solved their own problems related to fall risk and detection. The different solutions provided knowledge that helped me in the design and architecture of the platform. Such as the framework SPINE [32] was an excellent way to see how the design and architecture of a large-scale project for a framework was done.

It is important that the platform covers use cases, and if a certain use case is not supported, it should be easy to implement or adapt the platform to support it. This can either be done by the creator of the platform, or if the developer writes an extension class/method by themselves.

## 1.5 Results

From the literature analysis, it is expected to gain extensive knowledge in fall detection and preventative research. Specifically, how they gather data, what kind of data is gathered and the structure of the data. The State-of-the-Art analysis will grant insight into existing tools and technologies. Based on the resulting knowledge, a requirement specification will be formulated - covering the identified needs. This will be the basis on which the platform will be developed and evaluated. The platform will also be evaluated by creating several example applications that follow possible usage scenarios. The platform will be released under the Apache 2.0 license and as part of the UbiCollab.org project.

| |
|---|
| **D1** - An analysis of fall related research for the elderly |
| **D2** - A State-of-the-Art survey of existing solutions and technologies |
| **D3.1** - A data model based on the discoveries from D1 and D2<br>**D3.2** - An API serving as the access point for applications and the data model |
| **D4** - A set of example applications demonstrating the platform |

TABLE 1.1: *Deliverables*

## 1.6 Research Method

The methodology this dissertation followed was to initially research the problem area. This included a literature analysis to establish the foundation of the defined problem. The next step was to analyse the State-of-the-Art solutions in the respective field, to identify and create an overview of existing research and technologies that exists.

An iterative approach based on *Scrum* was adopted for the agile development process. Daily goals were set, with milestones to be completed every two weeks.

## 1.7 Report Outline

**Chapter 2 - Methodology**

This chapter presents what methodology that was used during the dissertation. It also summarizes the milestones set.

**Chapter 3 - Problem Elaboration**

This chapter presents the problem in a detailed way based on the current research in the field of fall detection and prevention.

**Chapter 4 - State-Of-The-Art Solutions**

This chapter presents and discusses several SotA solutions that relates to the problem definition.

**Chapter 5 - Proposed Solution**

This chapter presents the proposed solution for this dissertation.

**Chapter 6 - Requirement Specification**

This chapter details the functional and non-functional requirements for the NoFall platform.

**Chapter 7 - Design and Architecture**

This chapter presents the design and architecture of the platform and the database. It lists architectural drives and significant requirements, and the stakeholders are identified with their correlating concerns.

**Chapter 8 - Implementation**

This chapter presents the strategy used during the implementation, explaining what and how the different requirements where implemented, and the different tools and technologies used are discussed.

**Chapter 9 - Results & Evaluation**

This chapter summarizes the results of my work in light of the hypothesis, evaluating the results as well as listing some of the major challenges.

**Chapter 10 - Conclusion and Further Work**

This chapter concludes the results based on the initial problem definition. Lastly, some possibilities for further work are presented.

# Chapter 2

# Methodology

The dissertation methodology was a three part process; research - development - evaluation (see figure 2.1). The research provided knowledge that formed the basis for the development. The development was an agile process with several iterations during the project. Finally, the evaluation of the platform was done.

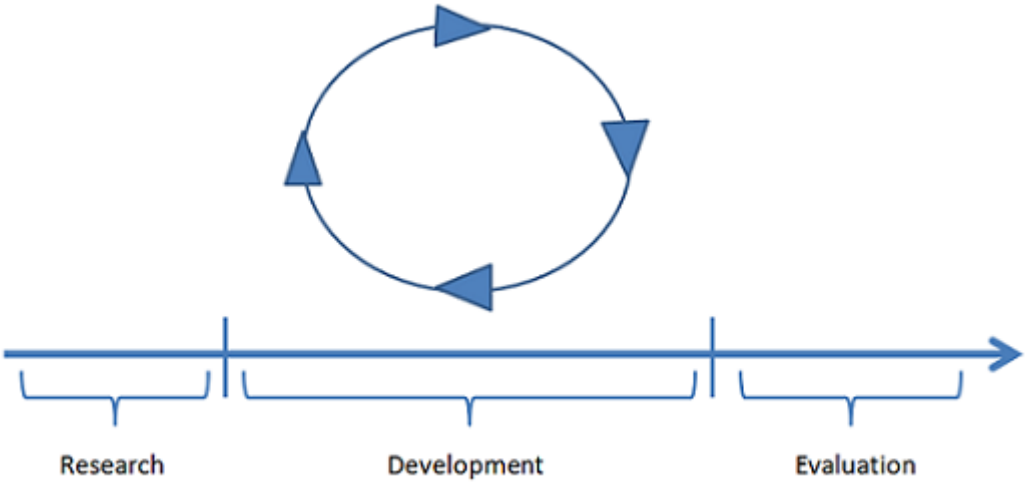



FIGURE 2.1: The Research Methodology

In this chapter, the methodology adopted for this project is presented with the milestones that lead to the completion of this thesis.

## 2.1 Research Methodology

**Literature Analysis**

The first thing that needed to be done, after an outline of the problem description was agreed upon with the advisor, was to perform a literature analysis.

The first step was to gather articles relevant to the problem definition. Based on different search criteria, 52 articles were selected depending on the abstract and summary. These articles where then skimmed through - to identify the better and more relevant articles. From these, 33 articles were selected for thorough reviewing.

It was important to find different articles that had similar results and different approaches to a hypothesis. This would enforce the findings, and make them more plausible to be true. Research that tried to disprove others findings, but failing or arriving at similar results was also important to include. Finally, reviewing large literature reviews done by researchers was a great source to see which methods and findings was prevalent in the respective field of research.

The result of this step is presented in chapter 3 - Problem Elaboration.

**State-of-the-Art Analysis**

By completing the Problem Elaboration, I had gained more knowledge and a better understanding of the defined problem. This made it possible to identify and select SotA solutions.

I wanted to gain a broad understanding of several different solutions and technologies, therefore a few solutions from many different categories where chosen. These categories where measurement techniques, tools, and technologies. Computer related solutions such as mobile applications and web pages were investigated. In addition, frameworks and platforms were looked at.

The analyses of a wide range of solutions, created the basis for the formulation of the systems requirement specification.

The result of this research analysis is presented in chapter 4 - State-Of-The-Art Solutions.

## 2.2 Agile Development Process

### 2.2.1 Scrum

As mentioned earlier, Scrum was adopted as the development process. Scrum's main goal of being flexible and adaptable to change was one of the main reasons for choosing this approach for the thesis. Doing research and development in tandem enabled for swift progress, however the need for flexibility was a concern. There could always emerge new results from the research, or new findings being discovered, that would alter the requirements for the development.

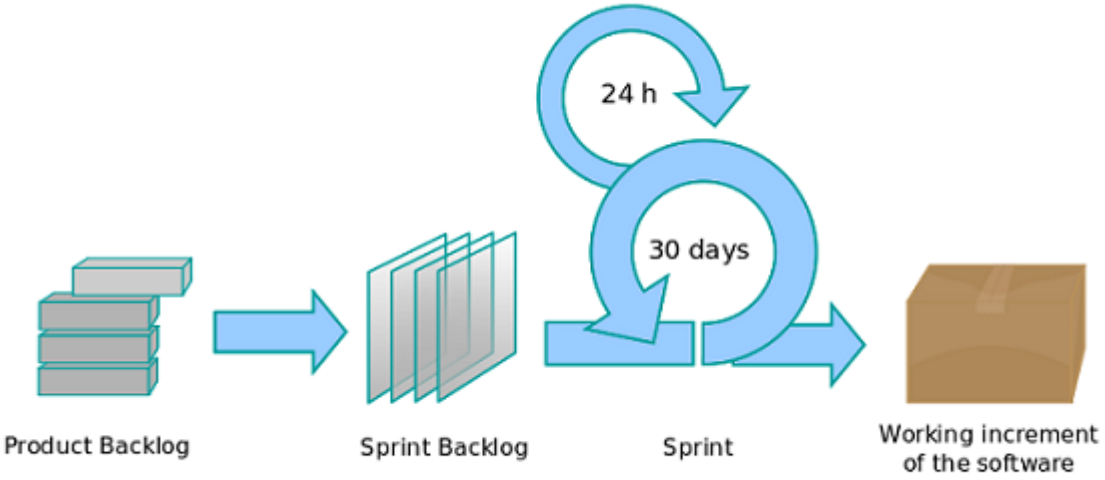



FIGURE 2.2: The development process in Scrum.

Scrum can be viewed as guidelines and recommendations, instead of a set of rules. To be truly iterative, Scrum in itself has to be adaptable and changeable. For this project, some changes were made to the original Scrum approach:

- After each sprint - instead of having a running and potentially shippable product, either a new chapter had to be written, or a new feature had to be working on the application.
- The daily Scrum meeting was omitted, since it was redundant to perform as a single member of the project "team".

- Some of the documentation has been dropped, since the thesis report in itself is more than sufficient.

## 2.3 Milestones

Three major milestones were identified in order to establish main measuring points for evaluation of the thesis progress and direction.

### 2.3.1 First Milestone

The first milestone was set to May the 30th. At this point several tasks was scheduled to be finished:

- Version 0.1 of the application
- Version 0.1 of the data model
- Literature Analysis
- SotA Analysis
- Chapter 3 - Problem Elaboration
- Chapter 4 - State-of-the-Art Solutions
- Outline Chapter 6 - Requirement Specification

The background and basis for the writing of chapter 2 and chapter 3 was established from the analysis done in the research phase of the dissertation.

*Chapter 3 - Problem Elaboration* and *Chapter 4 - State-of-the-Art Solutions* were at this stage at a satisfyingly level to be left alone until a more thorough re-reading and re-writing was to be done. *Chapter 6 - Requirement Specification* had the initial draft completed. The probability of having to revise and rewrite this chapter was high. This is because the requirements will change as more knowledge is acquired, and what is in the beginning thought of as the correct solution, no longer applies.

The application framework at this stage was at a level that could be considered version 0.1. This comprised of mostly a code skeleton, where all the major parts had been placed into the architecture, but minimal functionality had been implemented. The focus was on getting a strong foundation to continue to build on at this stage.

The DB model version 0.1 was completed at this time. It covered all the major parts of the application (medication, tests, surveys, sensors, standards). At this stage, an attempt at visualizing the standardisation of risk in relation to the different measurements had been made. This made it possible to see how the different applications would record and measure different data for risk and at the same time be able to share their results with each other.

The idea was to divide all the risk measurements into three different categories, low - medium - high. In relation to walking speed, one could separate them into e.g. 0 - 0,7 m/s = high, 0,7 - 1,3 = medium and 1,3 - = low. If it were possible to categorize all risks in such a manner, it would be possible to combine all the results, and give a satisfying estimate of risk.

### 2.3.2 Second Milestone

The second milestones deadline was set to October the 15th. The major tasks to be completed for this milestone were:

- Content Provider v. 1.0
- Content Contract v. 1.0
- DB model v. 1.0
- Example Applications
- Chapter 6 - Requirement Specification re-written
- Chapter 7 - Design and Architecture
- Chapter 8 - Implementation

This milestone was the major one in relation to producing results. The main focus were on production, and minor resources was focused on research. The application platform

with its database was to be completed at a satisfying level to be released for testing by those who would wish to use it. It was emphasized that this was version 1, and in most likelihood, several bugs would occur.

Before the implementation was performed, *Chapter 6 - Requirement Specification* was re-written. This was because the focus of the thesis shifted from the development of an independent fall prevention application to be focused on the development of a platform that future developers could build high-level applications on.

Based on chapter 6 the architecture was designed and documented in *Chapter 7 - Design and Architecture*. This was done to ensure that a well thought trough design was implemented, both to avoid technical debt and to implement a sound solution.

The DB model went through several re-factoring phases from the initial version completed in milestone 1. The biggest change was that the Standards specification was introduced, and all risk related data in all the other specification was moved to standards risk tables. The point of this was that there would exist a pre-defined NoFall standard (as mentioned in milestone 1). These values could then be mapped to risk standards that the developers of other applications might have. They would be required to divide their value ranges to work with the, low - medium - high, risk of NoFall. Other changes to the DB focused on increasing the modularity and separation of the different tables etc. so the DB can support several unique use cases.

The feedback from testers is invaluable for the improvement of software, since it is impossible to find all bugs or see inconsistencies (often developers get tunnel vision for spending so much time on a project, and therefore cannot see obvious design flaws).

The development of some example applications, using the platform was also scheduled. These would help other developers to see how to use the platform, and get inspiration for ideas of their own. Two of the examples are a pedometer application to gather some data, and a graphical data display application that used the data gathered and presented it.

Based on the results of the implementation, *Chapter 8 - Implementation* was written. This chapter covers the different technologies used, the strategy that was adopted and what was implemented with related issues.

### 2.3.3 Third Milestone

The last milestones deadline was 30th of November. The major tasks for this milestone was:

- Re-designed the data model
- Chapter 5 - Proposed Solution
- Chapter 9 - Evaluation & Results
- Chapter 10 - Conclusion and Further Work
- Polish the report - layout, structure, spelling, language

I continued to iterate on the application platform during this milestone, even though there was no specific task related to the coding. This consisted of bug fixing, optimizations and changes that became evident as time progressed.

The database went through a re-design late in this milestone. The biggest change in this version was that the aspect of *Standards* was toned down. I felt that by defining standards, I tried to bully the users of NoFall into using what I chose. This was against the goal of the platform - being open for as many use cases as possible. Therefore they where changed to *Risk Definitions*, which will enable the developers to put in their own measurements and risk levels related to a certain measurement. (e.g. walking speed; different levels of risk related to the speed). The *Risk Definitions* has almost the same relationships with the DB as the *Standards* had, but they will no longer contain predefined values that I chose, rather that the developer have the possibility to add their own. The relations between the different tables were corrected and improved.

Chapter 5 - Proposed Solution will describe the NoFall platform solution that became the end product to this thesis. This chapter creates an overview of the solution, while the details will be presented in the implementation and design chapters. It contains high-level figures of the DB and architecture with description. It will also show some of the example applications that where implemented using the platform.

Chapter 9 - Results and Evaluation summarizes all the results achieved for this project. The hypothesis is evaluated relative to the dissertation. The evaluation of the NoFall

platform is presented. Finally, the challenges that I met during the thesis are listed, and how I dealt with them.

Chapter 10 - Conclusion and Further Work summarizes the thesis and concludes based the hypothesis. Ideas for improvements and identified aspects that need to be improved are explained in the section of Further Work. This is a very important part of the thesis, to make it easier for others to continue the work on the NoFall platform.

The final touches were made on the report before delivery, which consisted of fixing layout, restructuring, spelling and re-writing.

# Chapter 3

# Problem Elaboration

The leading cause of injury-related visits to the emergency departments is fall-related. It is the leading cause of accidental deaths [3]. Each year over one third of persons over 65 years of age fall, and half of them are recurring incidents [4–6].

In this section, I will look more into the causes and possible preventions of falls, the economical implications of falls, and lastly the acceptance of technological assistance.

## 3.1 Causes

To understand why the elderly are at such a high risk of falling and be able to prevent or diminish the chance of them occurring, one has to look at what causes the falls. Several risk factors have been identified as causes for falls. They are medication, psychological, age-related changes and environmental [7, 8]. The chance for a fall increases drastically as the number of risk factors a person has increases (*[...]recurrent falls increases from 10% to 69% when the number of risk factors increases from 1 to 4 or more* [7]).

To more accurately determine the risk of falling, one has to be aware of which risk factors the person has. This is because some of them increase the risk more than others. In a study done by Davies and Kenny [9] they found: *Risk factors included: visual acuity defect (23%), gait abnormalities (35%), balance abnormalities (19%), depression (31%), four or more prescribed medications (27%), culprit medication (39%) and benign positional vertigo (8%).*

Another study done by A. S. Robbins et. al. [18] found that there were significant differences between fallers and non-fallers. They identified the biggest common factors in the fallers as weak hip, poor balance, and more than four medication. From their study they found that the risk of falling were 12% for those with none of these risks to 100% with all of them.

Even though the results from the different studies vary, one can see that there is a strong correlation between the number of risks and which risk factors that affect the person, and the chance of falling.

With age, muscle weakness is common and an increasing factor for falling. Studies have shown that as many as 48% has reduced strength in the lower limbs for non-institutionalized patients and 80% for patients of nursing homes [7]. Having weaker lower body strength makes it harder for the elderly to avoid falling, regaining balance etc. This has shown from many different studies to increases the risk of falling.

The elderly are often prescribed many different medications, and many of them are vital for their quality of life and health. Recurring for many of the studies is that there are a clear relation to medication and the risk of falling [4]. The medications have different degrees of impact on the risk of falling. Central Nervous System medicines, especially psychotropic drugs, seem to be associated with an increased risk of falling [10]. It is important to note that when a person is on medications, this is an indicator of the person having relatively poor health. Therefore, if the medication is a direct link to the risk of falling, or if it is a combination of several other factors is hard to tell. As stated by [10] they do not conclude directly that medication is a cause for falling, but rather say that from the results it is plausible. In addition, it is hard to determine if reducing the risk of falling would be more beneficial than having reduced the effects of the medication.

Falls are a major psychological burden on the elderly. A study asking the elderly to rank their biggest fears, the fear of falling was ranked as number one, in relation to other common fears [11]. This shows at which degree the fear of falling is a psychological burden that has to be taken seriously. If they have fallen before they often become scared of the chance of a recurring fall, and therefore becomes cautious and inactive. The inactivity leads to reduced physical abilities that increase the chance of falling [3]. This is one of the reasons why the people who has fallen before has an increased risk of falling again. The constant fear of falling, limiting oneself to inactivity, and in general

reducing ones quality of life can lead to depression, which from [9] is shown to be an increasing risk factor.

When the elderly have fallen, they appreciate advice from health care professionals, but they were reluctant in seeking it themselves [11]. This might be related to the fear of becoming a burden on society, and therefore it is important that healthcare professionals take the initiative to educate and advise the elderly.

One of the leading causes for unpredictable falls are environmental barriers. They are responsible for 30–50% of the falls [7]. Such falls are more prevalent at home, this can be related to them being more comfortable and sure of themselves at home, and therefore not exerting such care that is needed. Nursing Homes are designed with such implications in mind, and have fewer possible barriers. Barriers come in many different forms such as stairs, slippery surfaces (bathroom floor), poor lighting or excessive lighting, thresholds and more.

Although the results vary from study to study, they all agree on what a risk factor is, and that having more risk factors will increase the risk of falling drastically. Since there are so many different factors that come into play in relation to the risk of falling, and studies have shown that the elderly are reluctant to seek help in regards to such matters, it will be useful to have applications that provide relevant suggestions and education based on the patients own registrations. This can be from basic advice as "be careful when moving around at night" to more extensive exercise programs, or that they should seek medical assistance.

## 3.2 Economical Implication

In regards to economical implications for the healthcare system, falls are one of the biggest, ranking number two, after motor vehicle accidents [12]. For the elderly, falls are ranked as number one, and there were recorded as many as 10 300 fatal and 2.6 million non-fatal fall related injuries in 2000 in USA [13].

Based on findings from [12] they estimated the costs for fall victims in the US: Costs per fall victim, per fall and per fall-related hospitalisation ranged from 2,044 to 25,955, 1,059 to 10,913, and 5,654 to 42,840 USD PPP and depended on fall severity. With such

high costs, the price tag for falls is reaching upwards to billions of dollars per year in treatments. This shows that not only is it important to prevent falls for the sake of the faller, but for economical reasons to.

To reduce the chance of people falling, good fall prevention initiatives has to be in place. *A recent meta-analysis of the intervention literature found that fall prevention programs, analyzed as a group, effectively reduced the risk of falling by 11%, and a systematic review reported that multicomponent interventions for community dwelling seniors reduced fall risk by 27%.* [13]

With the elderly population increasing, it becomes important to reduce the number of fall related injuries. Developing innovative and cost efficient solutions, which can help preventing falls, can be a big economical beneficial factor.

## 3.3 Prevention

There have been many studies on the effects of different prevention initiatives ranging from medication regulation; simple, tailored or comprehensive exercise programs, to home proofing. The studies focuses on single- and multifactorial assessments, where multifactorial has proven to be the most beneficial [4, 14–16]], but this is a more extensive intervention and will be more costly and time consuming. It is suggested that if the intervention program is not properly targeted it will fail to achieve maximum potential [14]. It has been shown from studies that there is a need for specialized and tailored programs for the individuals [17, 18]. This is related to the big difference in relative health to each elderly.

Introducing the elderly to an exercise program with strength training combined with balance training has shown to be an efficient way to reduce the risk of falling [4, 7, 8, 15, 16, 19, 20]. This is a cheap and easy initiative, which makes it one of the more attractive measures to take. Faber et. al. in [20] points out that the health of the patient has to be taken into account when considering exercise. If the patient is frail, exercise induces more harm than good.

There is a difference in opinion if the exercise program needs to be tailored for the individual or if a more general approach can be used to achieve good results. In [17],

[18] and [8] they argue that it is necessary with a tailored program, because the elderly usually have big differences in health. While many others suggest that a general program will be helpful [4, 15, 19, 20], but they do not argue if one is better than the other is, they just say that exercise has proven to be beneficial and should be used as a single factorial intervention or in multifactorial intervention. Gillespie et. al. points out that a specialized program from a health professional will be more effective than a general program [16]. In [7] they conclude that exercise can be beneficial, but what, how much and at what intensity is unclear, and more research is needed. This again relates back to what Faber et. al. pointed out in regards to the elderly being frail, and that some might not be able to exercise at all, or need really low intensity programs.

Balance training, such as Tai-Chi, may be effective [7, 16, 19]. This might be a good single intervention for some of the elderly if they cannot perform strength training. Acquiring better balance might reduce the risk of falling.

Many of the studies also recommend modifying the home for the elderly, especially those that are considered at a high risk of falling, by removing hazardous objects [4, 7, 15, 16, 19, 21]. If performed by a professional, this can reduce the risk by up to 1/3 [15]. It is an easy and cheap measure to take to improve the quality of life for the elderly, making it easier for them to live by themselves and be safer.

If the patient is going to receive a multifactorial intervention, a professional doctor can assess the medication prescribed and over a period of time reduce the amount taken [4, 7, 21]. Which drugs are directly linked to the increased risk of falling vary from study to study. In [7] they point out drugs for antihypertensive, diuretics and benzodiazepines. A broader recommendation is made in [4], saying reduction or withdrawal of psychotropic medications. As stated earlier, this is one of the harder parts of reducing the risk of falling, and should be done over time, with the supervision of a health professional.

Even though it has shown as ineffective as a single intervention, the education of fall prevention is important. This can be related to the fact that many of the elderly are aware of the risk and consequence of falling, but they do not see themselves as one of the people that are at risk [22, 23]. If they are properly educated, they might change their view and by default become more conscious of dangerous situations, and exert the needed care.

Based on the studies done on fall risk related to different risk factors and how to prevent them, the development of an application that use this information can be a powerful tool to support the elderly, in a cheap and easy way. Tracking medication, movement patterns and providing education for the elderly, and simultaneously bypassing the "I do not want to be a burden" mentality, since there is no time consumption from other people, just the usage of an mobile application.

## 3.4 Technological Assistance

New technology can often be overwhelming and daunting. For the elderly who did not grow up surrounded by ever changing technology, computers and the like can be hard to grasp. Some of the elderly chooses to reject them instead of assimilate, as shown from [24] several of the elderly refused to participate in an experiment only because they heard the word "computer".

Although many of the elderly are reluctant with technology, there are always two sides of a coin, and Nyman and Yardley developed a web page for the elderly that focused on balance training. This site was, despite having one usability problem, well received among the older people and sheltered housing wardens [25]. This shows that there is potential in developing simple computer assisted systems for the elderly.

Two studies, one focusing on computer-assisted instruction in medication recall training [26] and one focusing on seeing if computers can be used to learn patients to cope with osteoarthritis [27], both found that using computers had a positive effect. The results from [26] showed a reduction in the medication non-adherence rate to 10% from 32%, which indicates that as a recall training tool it was rather successful. From [27] the subjects showed significant increases in knowledge and significant self-reported, beneficial behaviour changes, including increased exercise, use of heat, and rest.

This shows that there is a big potential for both acceptance for the use of an application, and that it will be useful as a supportive tool for the elderly. The big challenges will be to be able to design and customize the application in such a way that it will be easy to use for the elderly, and at the same time be effective as a tool.

It should also be noted that in 10-20 years, many elderly will be proficient with computers. This means that the fear of computers and technology is a relatively short-lived phenomenon. With an increasing target group that are accepting to computers, the research in this field will have an increasing value.

## 3.5 Chapter Summary

| |
|---|
| **Causes for falls**<br>• Medication.<br>• Poor Strength & Balance.<br>• Barriers in homes.<br>• Psychological. |
| **Preventative Measures**<br>• Strength & Balance training.<br>• Change\Reduce Medication.<br>• Educate about risks of falling.<br>• Home-Proofing.<br>• Risk assessment and early screening |
| **Technology**<br>• Elderly are reluctant, but willing to learn new technology.<br>• Big potential to develop solutions that are cheap and cost efficient. |

TABLE 3.1: *Summary of Chapter 3*

Although the results vary from study to study, they all agree on that having more risk factors will increase the risk of falling drastically. And one of the biggest challenges to reduce the risks would be to try and reduce the medication intake. It is hard to determine if reducing the risk of falling would be more beneficial than the effects of the medication.

Fall proofing a home would be an economical cost, but one that is rather easy to implement. The application can point out the dangerous areas of a house, which things to take extra care with, and try to avoid etc. Giving them reminders of using correct lighting, e.g. if the elderly wakes at night, the mobile can register that they have gotten out of bed and give a notification/reminder of such things.

There are different opinions on exercise; if it works, or how much is needed. Many factors play into this, since the physical health of the elderly vary a lot, tailored training programs are needed. This is something that can be provided through an application, through collaboration with health professionals.

One easy but important thing is that just by educating the elderly and making them conscious of the different risk factors and how to limit them, in itself will reduce the risk of falling. Therefore having an application that will, based on various input, tell the user if they are prone to falling, can reduce the risk of falling.

Last, research has shown that the elderly can be reluctant to use new technology, but that this is not final, and many are in fact eager to learn. Creating an application for the elderly, designing it so they can take full advantage of it will be a big challenge. But from this review one can see that there exists room for such research, and that it can be successful if done right.

Providing the developers of future fall preventative applications with an underlying back-end platform, with a thorough and well-defined DB based on the most resent research in fall prevention, will make it easier to develop high-level applications. The developer can then focus more on creating an application that the elderly will use, focusing on the GUI and logical flow of the application. This will reduce the cost and time of delivery of the applications, and it will increase the standards of the applications.

# Chapter 4

# State-Of-The-Art Solutions

In the field of medical science and fall related research there exists many solutions for preventative measures. These range from simple questionnaires to extensive computer programs that will automatically track and calculate the risk of falling for patients. Based on the problem elaboration and analysis for this thesis, relevant research and technologies have been selected.

This will help to create an overview and map out the existing solutions, and see how they related to this research. This will also make it possible to choose some of them to help further progress this research, both as an inspiration and directly using them in the solution.

## 4.1 Fall Risk Assessment Measures and Tools

To accurately select the correct assessment tool, one has to consider in which context it is going to be used. From a survey done by V. Scott et. al. [28] they looked at 38 different tools that met the inclusion criteria that was set for the review. They concluded that there exists several reliable and good tools that deliver good results, but most of the tools were only tested ones or in a single setting. Based on this, a single tool cannot be recommended to be used in all settings, and a more thorough selection process needs to be performed.

Depending on the purpose for the tool, such as a quick screening to determine the risk of falling, a quick and easy-to-apply tool is necessary. If the purpose is to reduce the risk

of falling, a test in which risk factors can be identified, to make it possible to focus on the correct area of prevention is needed. Choosing an appropriate tool is a complicated task. The lack of consistency in methods and interpretation of fall-risk assessment tools in the published literature today is the cause for the complexity.

In this section an assessment tool, a sensor to determine risk of falling, a test to determine risk of falling and a model that takes into account assessment parameters to identify the risk of falling are summarized and valued. These solutions where selected to give a broad approach to the field of existing solutions and different approaches that has been made.

### 4.1.1 ADAPT Fall Assessment Tool

The ADAPT Fall Assessment Tool is a documentation system that is intended to ensure fall risk reassessment every 12 hours. The initiative to create such a tool came from the increasing rate of elderly, and decreasing rate of health care personnel. ADAPT are embedded into routine assessment documentation, eliminating added charting time [1].

| |
|---|
| *Create a computerized documentation solution that would ensure fall risk reassessment no less than every 12 hours.* |
| *Develop an evidence-based fall assessment tool that would accurately reflect the fall risk of the patient. Indicators would be embedded into pre-existing assessments, providing timely, accurate (rather than pasted) assessment without increasing charting time.* |
| *Design a program that would tailor interventions to specific patient risks rather than giving multiple intervention options to patients with significantly different fall risks.* |
| *Integrate the fall risk information into care plans, report sheets, audits, and care conferences to produce an interdisciplinary communication network.* |

TABLE 4.1: *The goals of the MHS Fall Committee [1]*

The risk factors that are incorporated in the system are based on the current recommendations in the literature on fall prevention. The falls are categorized into 4 broad areas (Disorientation, Activity, Post medication and Toileting), where each category has

several risk factors specified for each category. 30 of the most common risk factors are included in the system; some of these are confusion, disorientation, depression and history of falls. Having categories makes it easier to discern what measures to take considering the different risk factors identified, improving usability.

The tool allows for tailored interventions for each patient, where the risk factors are integrated into the pre-existing documentation that is in use today, such as the report sheet. This makes the integration of the new system easier, and the workers will see it as an addition, and not an entirely new task.

The basis for this tool is consistent with the findings in chapter 3. In regards to the risk factors that are included in this tool, the same categories and specific risk factors were found. As summarized in chapter 3, a tailored intervention program for each individual is the best approach, since often the risk factors vary notably. One of their goals was to provide tailored intervention, which will produce the best possible preventative measures.

The ADAPT tool is a good solution, but it is not designed for the patient (the user in regards to this thesis), but rather for the healthcare personnel. Therefore, a slightly different approach has to be made, but several commodities can be taken advantage of, such as the idea of fall risk reassessment at certain intervals and the categorization with standards.

### 4.1.2 Fallarm

Fallarm [29] is a solution for preventing falls that is suitable for usage in hospitals, care facilities and home settings. It is a wrist-worn sensor that continuously tracks and assesses the individuals risk of falling. It gives constant feedback about the actual risk, and thus increases the awareness of the wearer and therefore reduces the risk.

From the studies reviewed in chapter 3, it showed that awareness and alertness reduced the risk of falling. Subjects that participated in the testing of the Fallarm stated that they felt safer when wearing the equipment. As found in chapter 3.1: fear is a psychological burden that increases the risk of falling. Removing or at least reducing their innate fear will reduce the risk.

Even though several studies suggest that having a sensor on the arm will produce less accurate results than a sensor worn around the waist, the results from the Fallarm

was satisfying. The subjects stated that wearing a waist sensor was embarrassing and obstructive for activities, and preferred the wrist-worn device. This corroborates that the elderly will use technological assistance, but the solution has to be non-patronizing and well made, as found in chapter 3.4.

Having such a device as the Fallarm increases the patients privacy, since there is no need for continuous monitoring from health personnel. Such monitoring has shown to increase stress with the patient, which should be avoided. It is important to note that the elderly rarely seek help in such manners, because they do not want to be a burden (see chapter 3.1). Having technology performing such tests and monitoring will increase the probability of the elderly seeking help, if they are discovered to be at risk. This is a cost-efficient solution, since there is no need for human time-consumption.

### 4.1.3 Timed up-and-go Test

The timed up-and-go test (TUG) [30] is a simple test done on patients to check if they are prone to falls or not. The test requires a subject to stand up, walk 3m, turn, walk back, and sit down. Based on the time it takes the subject to complete the test, one can determine the functional mobility. Functional mobility is a term used to reflect everyday life movement and the balance, such as the ability to walk alone or walk up stairs.

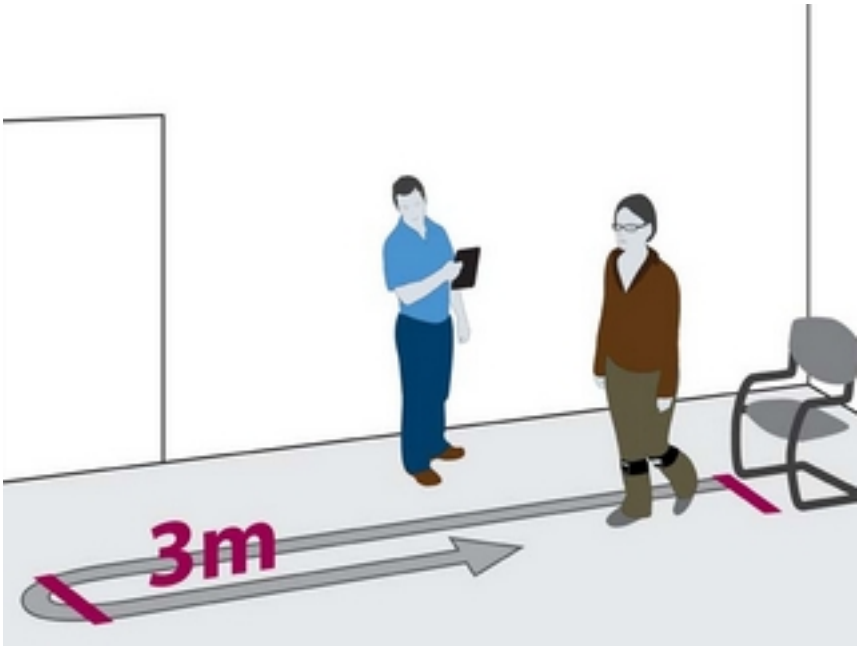



FIGURE 4.1: Image of the TUG Test.

| | Cutoff Score (s) | Sensitivity (% Fallers) | Specificity (% Nonfallers) | Overall Prediction | Predicted Probability |
|---|---|---|---|---|---|
| TUG | ≥13.5 | 80% | 100% | 90% | .77 |
| $TUG_{manual}$ | ≥14.5 | 86.7% | 93.3% | 90% | .5 |
| $TUG_{cognitive}$ | ≥15 | 80% | 93.3% | 86.7% | .5 |

FIGURE 4.2: Sensitivity, Specificity, and Predicted Probability for TUG [30].

From the study, they found that the TUG test was both a sensitive and specific method to identify elderly individuals who are at risk for falling.

The results show that subjects who took 14 seconds to complete the test had an 83% probability of being a faller, which means that individuals that take longer than 14 seconds have a high risk for falls. The cut-off value for the test varies some from study to study, which can be because they used different test subjects.

Poor balance and weak lower body strength are both indications of risk for falling, as found in chapter 3.1. Having a test that in less than a minute can determine the risk of falling for an elderly person, based on these risk factors, is invaluable. With some technological assistance, it could become possible to perform the TUG test without the assistance of others. This would be a great benefit given the fact that the elderly does not want to be seen as a burden on society, and will often avoid seeking help.

### 4.1.4 Hendrich II Fall Risk Model

The Hendrich II fall risk model [31] is a simple model that takes into account eight assessment parameters which results in the identification of risk for falling for a patient. The tool can easily be inserted into existing documentation forms, or as a part of an electronic record with risk analysis.

The study assessed more than 600 risk factors, which led to several statistical values for the increase risk of falling. A patient with altered elimination needs was 1.67 times more likely to fall than someone with normal needs. A subject who could not rise in a single movement during a TUG test was shown to be 2.16 times more likely to fall than someone who did it in a single movement. The statistical model is important, but harder to use, and is more useful when converted into a risk point system. Several other factors were looked at, and led to the model shown in Figure 4.3.

| Risk Factor (≥5 = High Risk) | Risk Points |
|---|---|
| Confusion/disorientation (mixed definition)* | 4 |
| Depression (mixed definition)† | 2 |
| Altered elimination‡ | 1 |
| Dizziness/vertigo (subjective definition)§ | 1 |
| Gender | 1 |
| Any prescribed antiepileptics | 2 |
| Any prescribed benzodiazepines | 1 |
| Get-up-and-go Test Item #2: | |
| "Rising from Chair" | |
| Able to rise in single movement | 0 |
| Pushes up, successful in one attempt | 1 |
| Multiple attempts but successful | 3 |
| Unable to rise without assistance | 4 |

*Charted as confused or disoriented or scored <17 on Mini-Mental Examination.
†Charted as depressed or scored >8 on depression test.
‡Charted with altered elimination needs or answered "yes" to any BET questions.
§Charted with dizziness or vertigo.
©2002 Ann Hendrich & Associates, Inc. All rights reserved.

FIGURE 4.3: The Hendrich II Fall Risk Model.

The categories that are identified in this model correspond well with the findings in chapter 2. They address balance and strength by doing a TUG test. Depression, experience of confusion or disorientation, and dizziness are accounted for. The specific medication the user takes. This study shows, through statistical data, that the evaluation of a limited number of factors can give a highly predictive indication of a persons risk of falling. Even though the results are not entirely specific, they can easily be used to deduce risk, and from that suggest measures to take.

## 4.2 Computer Assisted Fall Prevention

### 4.2.1 Web Page for Balance Training Advice

As previously noted (Chapter 3.4), Nyman and Yardley developed a web page for the elderly which focused on balance training. This site was, despite having one usability problem, well received among the older people and sheltered housing wardens [25]. This shows that there is potential in developing simple computer assisted systems for the elderly.

The web page follows a step-by-step sequence where a profile of the user is built (See figure 4.4 for the steps). During the profiling, questions such as age, gender, how many medicaments the user takes, if they have osteoporosis and more are asked. These questions are based on the same factors that where found in chapter 3. Based on the results on the various questions; advice, recommendation and training suggestions are given.

The result from this is a semi-tailored training program for the user, with focus on balance training. They suggest several different exercises and activities the user can perform, such as Tai-Chi, walking and swimming.

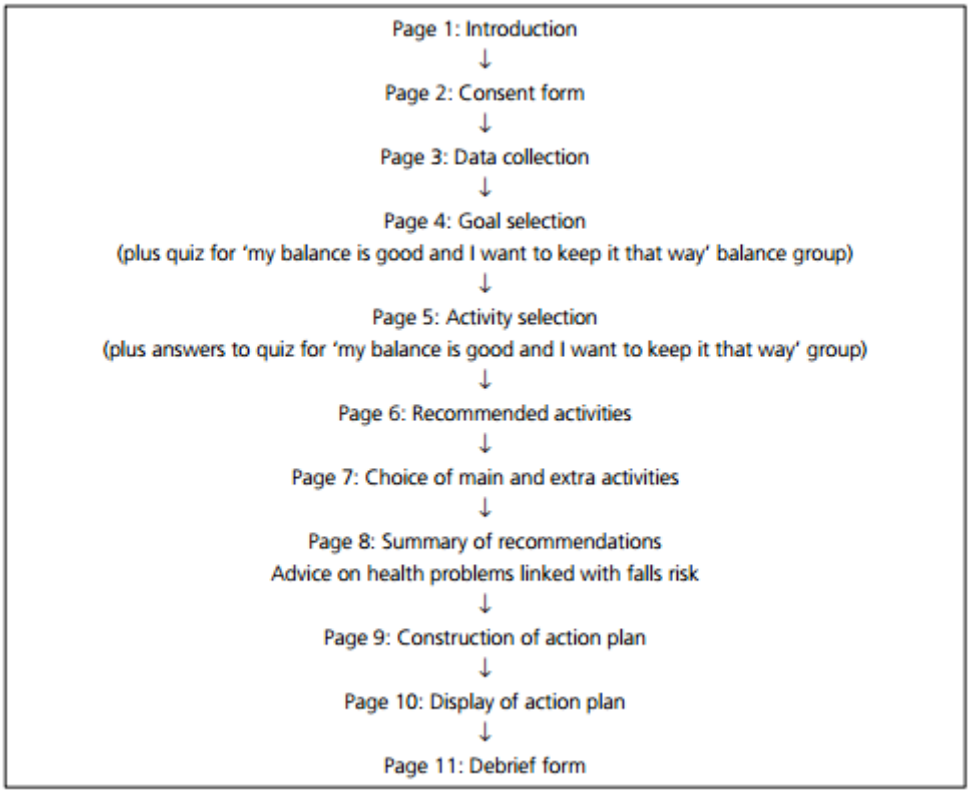



FIGURE 4.4: Sequence of web pages.

Guiding the user through steps that are simple, clear and easy to understand is important, since many of the elderly have low or no understanding of computers. It becomes important to avoid the usage of intuitive designs that are aimed at experience users (e.g. symbols that have gained certain meaning through continuous usage in web page design).

As seen in figure 4.5, the interface is simple and clean, with no excessive design clutter. But it should be noted that acceptability and perceived usability is a challenge when

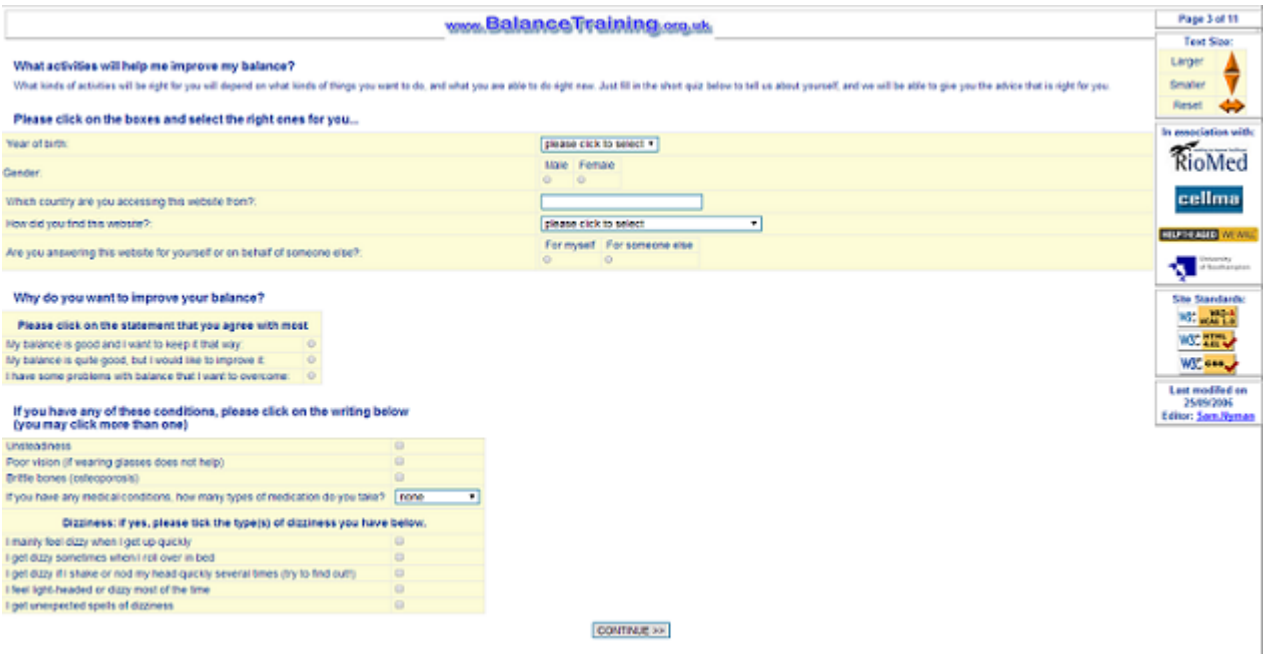



FIGURE 4.5: Screenshot of one of the web pages. (http://www.balancetraining.org.uk/)

designing for the elderly. Some previous work with prevention advice has been reported as either common sense and potentially patronizing, or frightening and oppressive [25].

### 4.2.2 Mobile Questionnaires for Elderly and Partially Sighted People

Through a project Holzinger et. al. built a mobile system for questionnaires [24]. The incentive for this was that often it is hard for the elderly and partially sighted to complete a written questionnaire, and therefore a better and simpler solution was wanted. This would also eliminate the need to type the answers into a computer at a later stage.

The system was developed by applying a User Centered Design including four levels [24]:

- Paper mock-up studies.
- Low-fidelity prototypes.
- High-fidelity prototypes.
- The system in real life.

Based on the results from the various iterations they managed to overcome several of the challenges that HCI have, such as the appropriate interplay between user interface design, the device and social context. An example would be that they built a special help function that activates after some time of inactivity, which displays a red arrow where the user is intended to press, to be able to take the next step.

From their user testing one example protrudes: *One elderly lady within an experiment first were very frightened and said that she does not like computers at all – after she touched-through the application, she asked carefully: "This was a computer?", as we replied yes, she said "... but this was really funny".* This shows that if the design comes across as genuine and respectful, but at the same time, it is easy to use, the user experience will be good.

FIGURE 4.6: User-test for mobile questionnaires [24].

The results from this project clearly show that there is a lot of potential in the field of medicine and mobile computing. The challenges are to develop useful applications that the elderly can, and more importantly will use. If one manages to develop a good user interface that works for the elderly, and that the application is perceived as something serious and helpful, the odds of it being used will increase.

## 4.3 Frameworks

Frameworks provide tested methods, guidelines and rules to follow. Software frameworks are universal, reusable software platforms for development. The main goals of software frameworks are to facilitate development by providing the low-level details of the system, allowing the users of the framework to focus on meeting requirements, leading to reduced

development time. Evaluation frameworks strive to give guidelines on how to evaluate research results, creating commonalities between different research results, making it easier to share and compare results.

Rapid development of prototypes are vital to progress research at a steady pace. It is also important to have commonalities between the different research results, so it becomes easier to compare and evaluate the results.

### 4.3.1 SPINE

SPINE is a project that aims at providing software instruments for rapid prototyping of WBSN-based applications. It is a lightweight and flexible framework, offering the developers a java API, which allows them to focus on problem solving algorithms and avoid using low-level programming languages for the sensor nodes, leading to a more rapid and robust process of development.

SPINE is a domain-specific framework, which is based on the following principles [32]:

- *Open Source. SPINE is an open source project to establish a broad community of users and developers. Its source code is released under the LGPL 2.1 license 21.*
- *Interoperability through APIs. SPINE provides local and remote applications with lightweight Java APIs that can be used to manage sensor nodes or issue service requests. The APIs are easily portable to devices of various capabilities, such as PCs or mobile phones.*
- *High-level abstractions. SPINE provides high-level programming libraries including protocols, utilities, data processing functions and specific support to easily specify new services and features. The layer defined by the SPINE service libraries allows application designers to program at higher levels of abstraction than the currently available operating environments (e.g. TinyOS).*
- *Rapid prototyping. SPINE helps designers to efficiently prototype distributed BSN data classification algorithms with respect to the use of energy and channel bandwidth.*

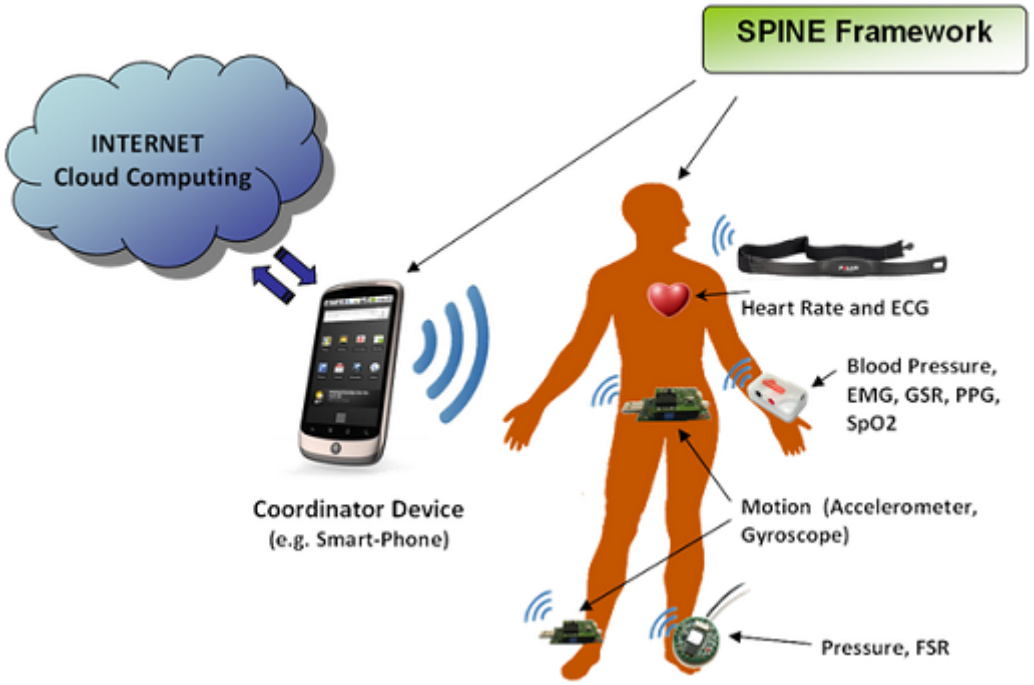



FIGURE 4.7: SPINE Architecture (http://spine.deis.unical.it/spine.html).

SPINE has been used in several projects focusing on WBSN applications. It has successfully been used to monitor human activity (e.g. walking or jumping) and recognize postures (e.g. sitting or standing). It has been used to detect falls, and if the wearer cannot get up. The prototypes have shown great efficiency and accuracy (97% correct recognition of posture/movement and almost no undetected falls).

Fast, easy and cheap prototyping can often be a factor in the successfulness of research. In the field of fall detection and prevention, a lot of research is done, with new ideas needing to be tested. The SPINE framework works as a step in the right direction of making it easier and cheaper to perform experiments.

Having a framework that can easily adapt and add new sensors to an existing system is of very high importance. It increases the longevity of a prototype. It broadens the possible usage areas and adaptability. All these points also increase the value of prototypes that are made, ensuring that they will not be a one-time use and then discarded. This might make it easier to fund certain research, or get the approval to develop a given prototype.

**Main findings from analysis**

- ADAPT Fall Assessment Tool
  - The risk factors used in this tool corresponds with the ones found in chapter 3.
  - Categorization of standards (Activity, Disorientation etc.) are excellent ways to separate risk factors and create an easier model of the data.
- Fallarm
  - A sensor does not necessarily need to be located on the waist. Wrist-worn or other placements can bring in accurate readings.
- TUG Test
  - Short, simple and easy testing can give good indications of risk of falling
- Hendrich II Fall Risk Model
  - It is possible to weight different types of risk factors. This makes it possible to calculate risk easier.
  - A limited number of evaluations can give a strong implication of risk.
- Web Page
  - Questionnaires used to establish a basis for how a tailored program should be formed.
  - Tailored programs for users can be effective
- Mobile Questionnaire
  - Questionnaires are often used in assessing risk of falling
- SPINE
  - Others are working to solve the issue of providing framework/platform solutions
  - Adaptability is really important for success

TABLE 4.2: *Summary of Chapter 4*

## 4.4 Chapter Summary

In this chapter some of the existing research in the field of fall prevention and related work has been explored. It shows that there exists a vacancy in easily accessible fall prevention measures or equipment, since most of the existing solutions require special equipment or the involvement of a healthcare professional.

The development of a device or software that can be used without the need for extra sensors or special equipment and/or the assistance of healthcare professionals could be very useful, both in regards to cost and the increased quality of life for the elderly.

As shown, frameworks and platforms are excellent ways to reduce development time, increase the quality of the end product and make reuse of code easy. However, it is hard to develop good frameworks or platforms that suit several use cases, and not becoming a niche tool.

# Chapter 5

# Proposed Solution

This chapter presents the proposed solution NoFall at a high level. Personas and use scenarios for NoFall is described, an applicable example is detailed for the usage of the platform. Finally, the top-level design for the architecture and data model is presented.

## 5.1 NoFall

In the field of fall detection and prevention, to this date there exists a multitude of mobile solutions for detecting and preventing falls. The problem is that they are all mostly standalone solutions. They have their own set of gathered data and results. NoFall is an attempt at uniting the different solutions that emerge in the mobile field. By providing a shared database, the individual applications can unite and share their gathered data.

Some of the problems NoFall attempts to improve:

- By having a shared dataset, the effectiveness and results of the different applications can be enhanced.
- Reduce the time it takes to develop a finished solution.
- Reduce redundancy: Reuse of code.

- Many teams develop several different solutions. They might have a sensor for pressure reading, a light detector, a questionnaire and a test. Having a ready and well-defined foundation, which they can build upon, can simplify their work, reduce time to release and improve the results.

## 5.2 Design Implications

In regards to the design of the platform, several factors has to be regarded, one of them are requirements for the solution. It is important to define what the solutions is required to support, as to not develop something that is outside of the scope of the solution.

The findings in chapter 3 and 4 create the basis for defining the requirements for the platform.

**Implications for design: Chapter 3**

**Causes for falls**

The causes for falls lead to several implications for design:

- Medication tracking.
- The user (elderly).
- Testing.

**Preventative Measures**

To keep track of preventative measures and the data gathered these factors has to be taken into account:

- Testing.
- Surveys.
- Sensors.

**Technology**

To reduce cost and increase the value, a back-end platform will make it easier to make high-level applications fast. This will rely on measurement standards, making sense of

the data that is gathered. This will also help other researchers understand the results, if they are shared.
Factors to consider for the design are:

- Measurement standards.

**Implications for design: Chapter 4**
From the analysis done in chapter 4, it became evident that there exist numerous solutions to the same problem. The number of solutions is an indication that NoFall has to be able to support several different solutions and answers to the problem. It should also be able to work for new solutions that has not yet been made. The implication to design from this is:

- The different parts of the database will support *Specification* - where the developers can specify their own solution. E.g. they can specify an entire fall test and save it to the database, with measurements and risks associated with the data it gathers. If they add a description and instructions to it, other users of the NoFall platform can easily make sense of this test, and use it themselves.

Depending on the solution, the data they gather can differ and how they measure and makes sense of it varies. This is something NoFall should support, by providing easy specification of measurements, data ranges, data type, data value, risk levels and more. Discovered from chapter 4 was that categorization of standards (Activity, Disorientation etc.) are excellent ways to separate risk factors and create an easier model of the data. This shows that, as mentioned earlier, using specifications to make sense of the data is a good solution.

It is desired to make many solutions collaborate - to further progress research and improve the solutions (if they share findings, more reliable predictions can be made). NoFall has to be accessible by many different solutions, and Android provides an excellent API for this - the Content Provider. It makes databases accessible for outside sources (i.e. other applications).

## 5.3 Applicable Example

To demonstrate how the NoFall platform can be used; several example applications have been developed using the platform as a basis.

- One application is a pedometer, where the pedometer code-base was acquired through an open source project and adapted to the NoFall platform.
- Another example is a graphical application that uses a simple XY-plotting graph or pie chart. The data it displays is from the pedometer application via the NoFall platform.
- A widget application was developed that can display different types of data it gets from the NoFall platform on the main-screen of the phone.
- A simple TUG-test application works simply by recording the time a user takes to perform a TUG-test.
- To show a medication profile, a medicament registration application was made. This enables the user to specify how many medications they are on, and if they know, which ones.

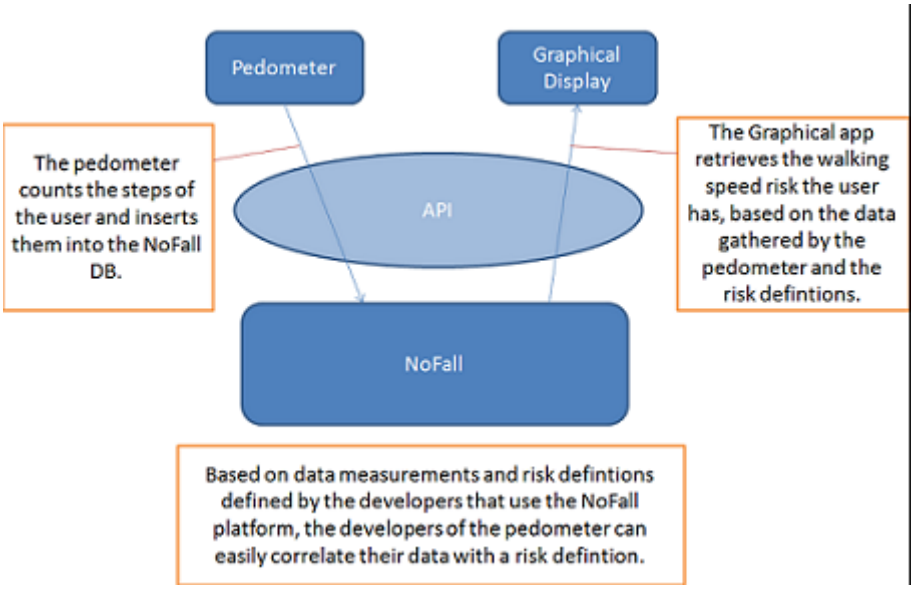



FIGURE 5.1: NoFall - Data-flow between applications

The adaptation of the pedometer application shows that the potential for using the NoFall platform on existing solutions is quite possible, and rather simple to accomplish

- albeit time consuming depending on the size of the pre-existing database and its complexity. This is important because a usage scenario might be that someone wants to develop a new application using the platform, but would be hesitant to use it - if they had several other applications up and running, but would also like them to work together. Instead of creating their own solution, they can then use the NoFall platform as a basis.


FIGURE 5.2: Example App - Pedometer

The graphical and widget applications demonstrate the advantage of having a shared database. They do not need to have any knowledge of how the data was gathered. They will only need code to retrieve it and display it in a logical fashion. This removes the need for having code for both a database and functionality for sensors or other means of data gathering. The complexity of the applications becomes reduced.

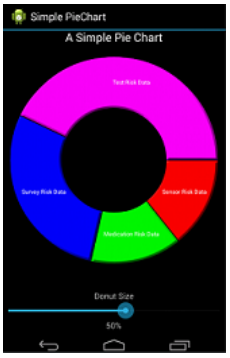



FIGURE 5.3
Example App - Pie Chart

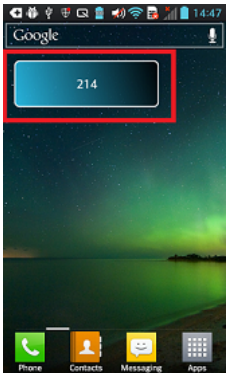



FIGURE 5.4
Example App - Widget

The TUG-test application is a simple stopwatch to record the time for a performed test. The code for this application is simple, needing one activity class to display the functionality and one query to store the data. Without the NoFall platform this application would need several other classes such as classes to create the database (This is considering that the design process of the database is excluded. Often this step is more time-consuming than the actual coding.), DB helper class and others.

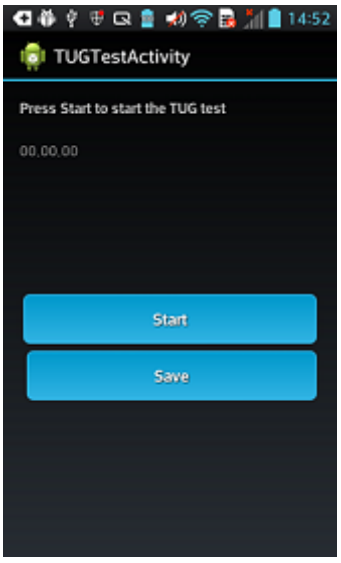



FIGURE 5.5: Example App - TUG

This application is a good example to demonstrate how different specifications connect to each other. See figure 5.6 for a coding example of how the TUG-Test results relates to different specifications in the database.

By using the NoFall platform, the applications have a reduced code-base, since all of the database code is separated from the native code. This makes it easier to perform code reviews and re-factoring, it also reduces the time it takes to implement the applications.

```
public Uri insertTUGResults(int time){
    Cursor cursor = null;
    Cursor cursor2 = null;
    Cursor cursor3 = null;
    try{
        String selection = MeasureStandards.MEASURE_TYPE + " = \"" + "WalkingSpeed" + "\"";
        String[] projection = {MeasureStandards._ID};

        cursor2 = myCR.query(MeasureStandards.CONTENT_URI,
                        projection, selection, null,
                          null);
        boolean temp;
        if(cursor2.moveToFirst() != false){
            temp = cursor2.moveToLast();
            String selection2 = TestMeasureSpec.FK_RISK_DEF + " = \"" + cursor2.getString(0) + "\"";
            String[] projection2 = {TestMeasureSpec._ID};
            cursor = myCR.query(TestMeasureSpec.CONTENT_URI,
                     projection2, selection2, null,
                       null);
        }
        String selection3 = TestSpec.NAME + " = \"" + "TUGTest" + "\"";
        String[] projection3 = {TestSpec._ID};
        cursor3 = myCR.query(TestSpec.CONTENT_URI,
             projection3, selection3, null,
               null);
        int id = -1;
        if(temp = cursor3.moveToFirst()){
            ContentValues values = new ContentValues();
            values.put(TestLog.FK_TEST_SPEC, Integer.parseInt(cursor3.getString(0)));
            // here the risk can be set based on calculations
            Uri testLogUri = myCR.insert(TestLog.CONTENT_URI, values);

            String path = testLogUri.getPath();
            String idStr = path.substring(path.lastIndexOf('/') + 1);
            id = Integer.parseInt(idStr);
        }
        ContentValues values = new ContentValues();
        values.put(TestMeasureLog.VALUE, time);
        if(temp = cursor.moveToFirst()){
            values.put(TestMeasureLog.FK_MEASURE_SPEC, Integer.parseInt(cursor.getString(0)));
        }
        if(id != -1)
            values.put(TestMeasureLog.FK_TEST_LOG, id);

        return myCR.insert(TestMeasureLog.CONTENT_URI, values);
    }catch(SQLiteException e){
        Log.i("SQLiteException when getting Standards and Test IDs", "error: " + e);
        return null;
    }
}
```

First one has to find the standard for measurements - in this case it is WalkingSpeed which is used in a TUG-test

Second one has to find the Test Specification to insert the correct values - in this case the TUGTest Specification

Third one insersts the values recorded by the TUGTest: Linking the data to the Measurement Specification, the Test Specification and the Test Log.

FIGURE 5.6: Example Code - TUG-Test

## 5.4 Top-level Architecture

The architecture at the top-level for the NoFall platform is shown in figure 5.7. The idea is that several different applications (independent or coherent) connects and uses NoFall through an API: Content Provider.

The Content Provider grants applications that are external access to the database of NoFall. It provides inserts, writes, deletes and updates. The Content Provider is a standard method Android provides for developers, and it is recommended to use when working with databases in many use cases. Since this is a recommended method, there exist several examples and tutorials of how to use the API. This is important, since one of the issues NoFall tries to improve is development time, and therefore needs to be easy to learn and understand. To make it simpler for developers to use the platform, NoFall will provide several *content contracts* - which are utility classes with constant variables that point to the different attributes in the database. This is also a standard way Android recommends to handle Content Contracts and its contents.

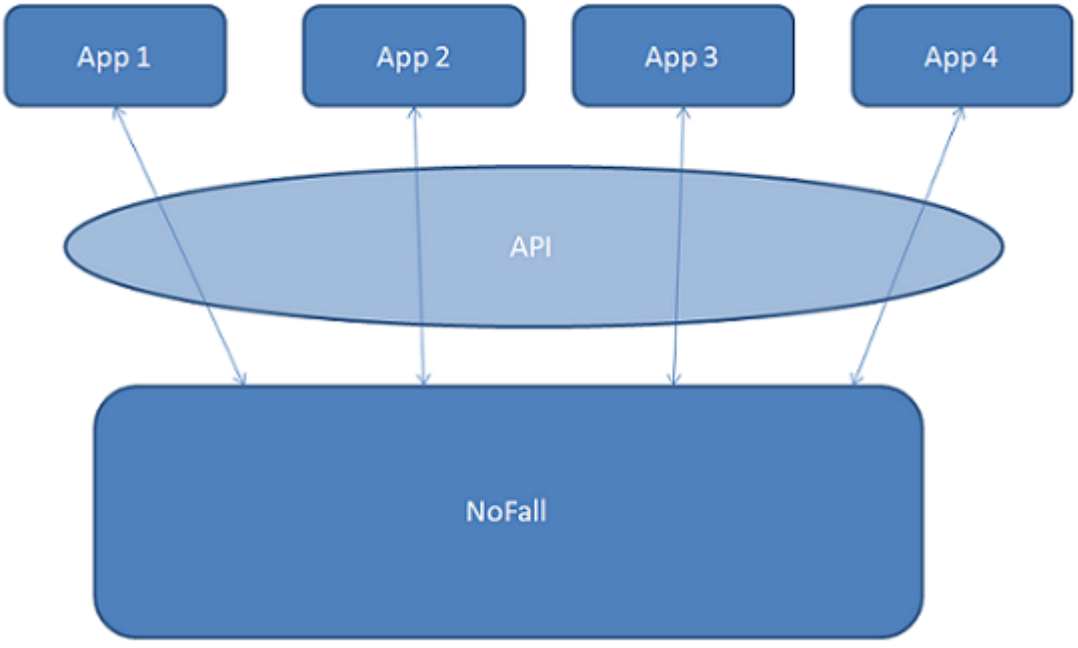



FIGURE 5.7: Overall Architecture

The most important part of the platform is the database and how it is modelled based on the findings in chapter 3 & 4. The model is divided into six different areas (see

figure 5.8 for a simple overview). They consist of data definitions (Measurement/Risk definition) and data gathering (survey, test, sensor, medication, user). See table 5.2 for the rationale behind each specification.

The idea behind the specification is to ensure that even if new methods for testing, new surveys are made or new types of sensors are invented, they can easily be registered in the database as a new specification. By modelling the database in such a fashion, it can accommodate a large amount of different use-cases.

An example of the table sensor specification:

| Specification example for a sensor: | |
|---|---|
| Table Name | Contents |
| Sensor Spec | This table will contain the sensors name and accuracy. It will also have an ID tag for which application registered this sensor. E.g. Name: Gyroscope, Accuracy: 95%, Owner: PedometerApp |

TABLE 5.1: *Sensor Specification Example*

This specification could then in turn collect data on movement. All of this data would be logged to a table that has a foreign key linked back to the specification. This way, it is easy to figure out from who and where the data came from, and how reliable it is. The data gathered would have a relationship to the measurement for e.g. movement speed, where movement speed would be related to a risk definition.

As discovered from the previous chapters, there exist several ways to collect fall risk data and several ways to measure it. The measurement/risk definition is where the developers specify which data that is gathered and what kind of measurement they use for it - as for the pedometer example:

- The data measurement gathered are steps and steps per minute, which can calculate speed.
- The data unit used is *int* and *long*.

They can also relate this measurement to different risk definitions. A simple way to do this is relating data-intervals to risk increments:

- The walking speed can be divided into intervals which correlates to increased risk of falling: 0 - 0,7 m/s = high, 0,7 - 1,3 m/s = medium and 1,3 m/s - = low.

| **Specification** | **Rationale** |
|---|---|
| User | When defining risk for a person, their age and gender will affect this. It is also important for flexibility to collect and store the total risk for the user in one place. Based on findings in chapter 3. |
| Medication | Studies have shown that the use of medication will increase the risk of falling. This should therefore be taken into account. Based on findings in chapter 3. |
| Survey | A good way to find out if a person is in risk of falling is a quick survey. The questions are meant to identify several important factors for the person that could relate to fall (e.g. history of falls). Based on findings in chapter 3 & 4. |
| Test | Tests can be quick and easy way to screen a person to check if they are in risk of falling (e.g. the TUG-test). Based on findings in chapter 3 & 4. |
| Sensor | Sensors are excellent ways to monitor a person. This can be used to measure walking speed, movement patterns, pressure points etc. Based on findings in chapter 4. |
| Measurement/Risk Definitions | To make sense of the data that is gathered, it has to be stored with a correlation to some sort of measurement and risk definition, otherwise it will only be some random number for other people than those who actually gathered and stored it in the NoFall DB. The basis for this was from findings in chapter 3 & 4. |

TABLE 5.2: *DB Specification Rationale*

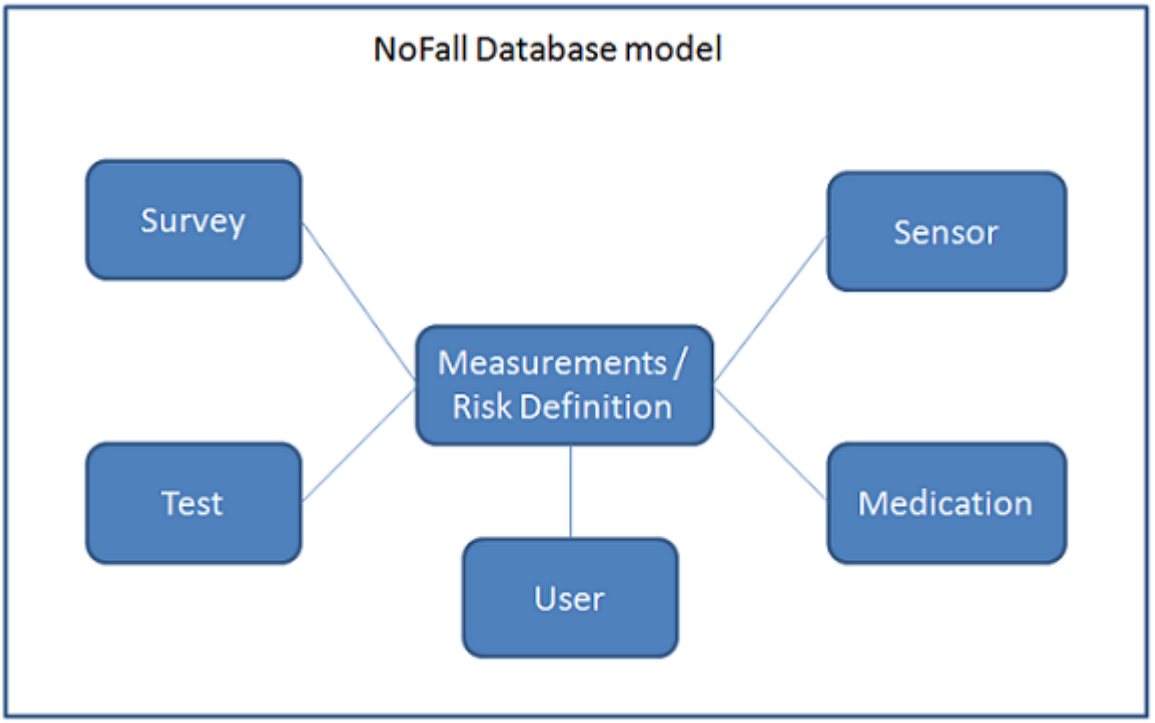



FIGURE 5.8: Simple DB overview

# Chapter 6

# Requirement Specification

To be able to create a good solution, having a well-built foundation and understanding of what requirements and goals the solution has to support is important. Based on the findings in chapter 3 and chapter 4 the requirements were established. They were iteratively updated during the lifetime of the project, since the implementation was done by following an agile methodology.

This chapter lists the component and technical constraints, details the functional and quality requirements.

## 6.1 COTS - Components and Technical Constraints

When defining a solution it is important to know which constraints one has to work with. By knowing them - one will not try to envision a solution that is in theory excellent, but impossible because of certain constraints with hardware or the technology.

Android is a Linux-based operating system. It is designed primarily for use on smartphones and touch screen devices. Android is open source and is released under the Apache license [33]. The constraints for the solution manly relate to Android and the devices.

### 6.1.1 Android Platform

The solution will be developed for Android, and will therefore only work on Android devices.

### 6.1.2 Android Device

The Android devices has limited amount of processing power and memory. Therefore, it is important to develop a solution that can run on several different devices and still perform well.

### 6.1.3 API level

The solution will be developed for Android 4.0 and target API 19. Some APIs will not work or work incorrectly if a particular device runs a lower version of Android. This will affect back-wards compatibility.

## 6.2 Requirements

In this section, the functional and quality requirements are detailed.

### 6.2.1 Functional Requirements

FR1 - Content Provider

*The content provider must provide access to all the tables that exist in the DB.*

Priority: High

FR2 - Content Contract

*Each table in the DB with its attributes must have a corresponding content contract for it. These classes will be used as convenience classes, making it easier to use the DB.*

Priority: High

FR3 - SQLite DB

*The platform must have a consistent, working DB with all the important tables (from the model).*

Priority: High

FR4 - DB Helper

*Predefined standard queries such as insert/delete/update should be available for the developers through the API.*

Priority: Low

FR5 - Persistent Data

*The platform must retain all information registered during usage, so it is available at next launch.*

Priority: High

FR6 - Data security

*The solution must secure the data (since it will be available for several sources), by sanitising the inputs to prevent SQL injection attacks.*

Priority: High

### 6.2.2 Quality Requirements

In development, there exist many different quality requirements (often referred to as non-functional requirements). For this project the two most important are *maintainability* and *documentation*. In this section the rationale for focusing on these two are made, and some requirements are detailed. Each requirement has scenario(s) associated with it.

#### 6.2.2.1 Maintainability

The applications platform should ensure that it is easy to build easily maintainable applications on top of it. This is because if using the platform results in an application that is hard to maintain, the benefit of using it, over creating the application from scratch is diminished. How easy it is to maintain in order to cope with a changed environment, and maximize efficiency and reliability are factors that have to be taken into account when assessing maintainability. In this respect, especially reliability, since the framework will be used with fall prevention technology, and humans are at risk.

U1 - Re-factor a class to improve reliability

*Re-factoring is a task that has to be done regularly when developing. This is to improve the work done, and ensure simple and readable code, which is easy to understand and maintain.*

**Scenario M1**

| | | |
|---|---|---|
| 1. | Source of stimulus | Developer. |
| 2. | Stimulus | Input from co-worker. |
| 3. | Environment | Lifetime of application. |
| 4. | Artifact | System. |
| 5. | Response | Re-factor class. |
| 6. | Response measure | Depending on complexity: maximum two workdays. |

U2 - Fix an error

*Unforeseen errors occur while an application is in use (release). There should be good error reporting in place, to ensure that it will be easy in the future to debug errors and fix them.*

**Scenario M2**

| | | |
|---|---|---|
| 1. | Source of stimulus | End user. |
| 2. | Stimulus | Input from user. |
| 3. | Environment | Usage of application. |
| 4. | Artifact | System. |
| 5. | Response | Debug and fix error. |
| 6. | Response measure | Depending on complexity: maximum a workweek. |

### 6.2.3 Documentation

To ensure that the NoFall platform (API) is used, good and thorough documentation is necessary. In many cases, if the documentation is lacking, the developers will chose another solution to reduce challenges, frustration and surprises.

D1 - Map the NoFall platform
*The development team has decided to use the NoFall platform. To ensure that everyone in the team knows what the API provides, how to use it, and its limitations, they create a mapping of the platform based on the documentation.*

**Scenario D1**

| | | |
|---|---|---|
| 1. | Source of stimulus | Developers. |
| 2. | Stimulus | Project team. |
| 3. | Environment | Usage of application platform. |
| 4. | Artifact | System. |
| 5. | Response | Map the platform. |
| 6. | Response measure | One workday. |

D2 - Look into the application examples
*A developer knows what he wishes to accomplish, but not entirely sure how to do it. He looks into the example applications to see if this has been solved previously, or if similar solutions exist that can be adapted to his problem.*

**Scenario D2**

| | | |
|---|---|---|
| 1. | Source of stimulus | Developer. |
| 2. | Stimulus | Idea from developer. |
| 3. | Environment | Usage of application platform. |
| 4. | Artifact | System. |
| 5. | Response | Solve a problem related to the API. |
| 6. | Response measure | One workday. |

# Chapter 7

# Design and Architecture

Giving some thought into the planned solution by performing design activities and defining the architectural choices, ensures that some effort have been put in before the implementation begin. Typically implementing something without design, results in a finished product that does not meet the requirements, is delivered late or is of poor quality.

The importance of good design increases with the lifetime of the application. There is no such thing as a perfect code-base, and it will always be susceptible for change. In this context, design is measured in regards to how easily it accommodates change.

Based on recommendations from IEEE 1471, this section contains a detailed description of architectural tactics and patterns based on the architectural drivers, the identified stakeholders and their concerns.

## 7.1 Architectural Drivers\Architecturally Significant Requirements

The main drivers that affect architecture and the system:

**Developer Inexperience**

I have done small projects before with android and mobile application development. This is the first time I work on a large-scale project alone, and not in a team, which affects the process. I have also never worked with the development of platforms.

**Project Lifetime**

By wishes from both my project advisor and me, the work done in this thesis can be used in other research or further improved. Therefore, it should always be kept in mind that this will be looked at by others.

**Technical Constraints**

The application platform is for Android and minimum 4.0 Android version.

**Project Deadline**

The thesis is done over two semesters. This makes it important to plan well to ensure that the project is completed by the deadline and satisfying work is delivered.

**Agile Development Process**

The project followed an agile development process. By having short deadlines with incremental progress, the risk of not having something working and completed by the deadline is reduced.

**Modifiability**

The code has to be easily modifiable because:

- There is new research emerging in the respective field, and the application platform has to be able to adapt easily.
- The requirements will change and evolve during the development process.

**Modularity**

The application platform should have low coupling, so it is easy to remove, change or add new functionality to the platform. This is both related to the need for an easily modifiable code base, and that further work might be done on it in the future.

## 7.2 Stakeholders and Concerns

The stakeholders of the system, and their concerns:

**Developer**

Constructability: The solutions complexity must be kept within borders of what is possible to complete within the deadline.

Quality: The report and code for the application must have a high level of quality, to achieve a satisfying grade.

**Advisor**

Reviewability: Easy to review and give feedback.

**Future developers**

Modularity: It is important that it is easy to change or remove parts without making ripple effects in the code.

Documentation: Well documented code, so it becomes easier to understand.

## 7.3 Architectural Description

In software architecture, a common approach is to follow already existing design patterns, which are reusable solutions to common problems. They conform to best practices and are in general a great way to tackle a problem. Many of the design patterns are excellent ways to guide architecture, but most of them were made before mobile development.

For the architecture in this project, some patterns and rules were adopted, such as naming convention and package structure. Otherwise, the code follows the Android framework architecture, which is a model view-controller architecture.

The Model-view-controller [36] pattern is a common pattern when developing a system that has both a GUI and some data handling. It separates the logic and the rendering of the GUI, which increases decoupling. The design pattern is less specific than many other patterns. It is more an idea for design, rather than a strict list of rules, leaving a lot of room for alternative implementations. In Android, each Activity is usually treated as a user interface making it act as both the controller and the view.

**File Organization**

When organizing files for an android project there are two sensible approaches:

- Organize all files by type: All activities in one package, all DB files in one package, all providers in one package etc.

- Organize all files by functionality: By this, all the files for one given functionality would be stored in the same package. E.g. for the start-up screen, the activity, adapter, DB files etc. would be stored in the same package.

For this project the first suggestion was chosen. This was based on previous experience with android development and sampling of existing projects on the web. See figure 7.1. for the structure used in this project.

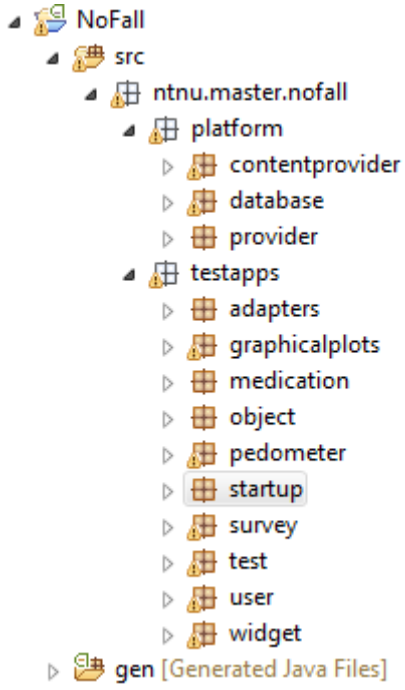



FIGURE 7.1: Package Structure

### Naming Convention

For naming, it is recommended to use prefixes to easily separate and identify files and components.

- Naming for XML files: The files should begin with what type they are. E.g. activity_NAME, dialog_NAME.
- Naming for components: A component should begin with which type it belongs to, and what type it is, followed by a name. E.g. activity_main_btn_login.

Following these will optimize auto complete, organization and avoid name collision. It will also make it easier for other developers to read and understand the code.

#### Base Classes

The android framework comprise of four main components that are the building blocks for an application [37].

- Activity - The activity is a class that represents a single screen with UI.
- Service - The service is a class that runs in the background to perform different types of operations, or remote processing.
- Content Provider - The Content Provider class handles data storing and sharing.
- Broadcast Receiver - The Broadcast Receiver is a class that handles and responds to system-wide broadcast announcements. E.g. battery is low.

It is recommended to use these types for everything the application will do, mainly because this is how the android framework has been designed, and is intended to be built.

#### System Architecture

Based on the previously mentioned methods and patterns, the main layout of the system architecture can be seen in figure 7.2.

### 7.3.1 System Goals

To realize the system a set of main goals and sub goals where defined:

- Provide storage for different types of fall related applications.
  - The DB should provide tables for the most common types of data that is related to fall.
  - It should provide recommendations/suggestions for standards to use, to make it easier to share the data with others. This ensures that third parties can make sense of the data.
- Possibility to share data
  - A content provider that provides access to the storage.
  - Content contracts for the different content that the provider has. They will function as convenience classes.

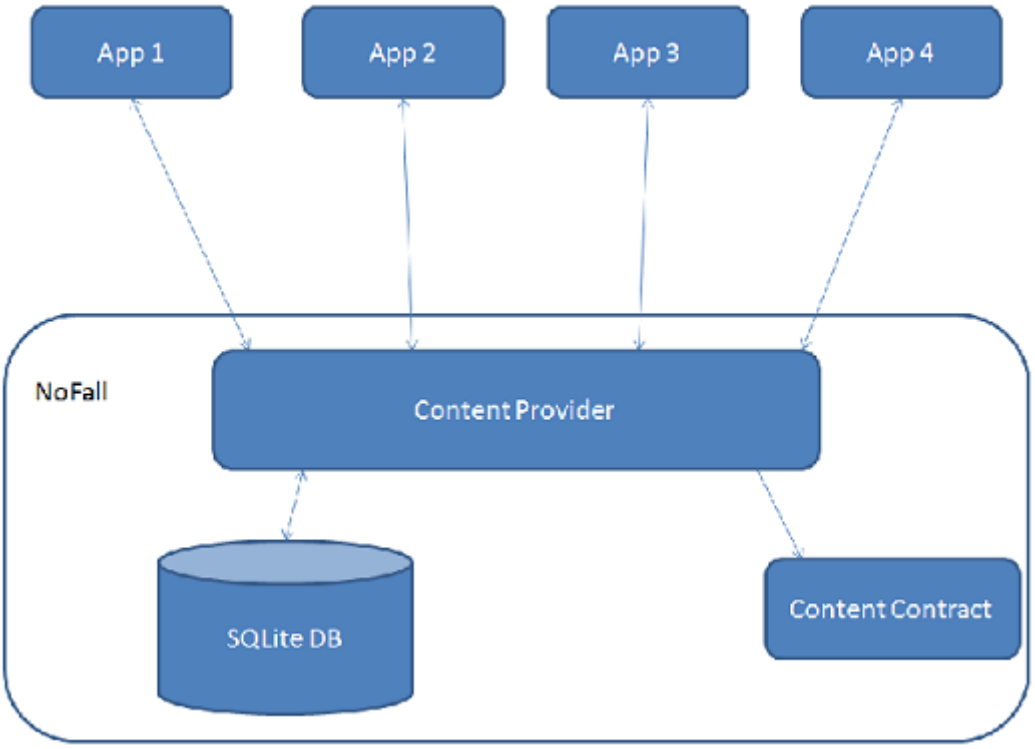



FIGURE 7.2: System Architecture

## 7.4 Data model

In this section, the SQLite DB model will be described in detail.

### 7.4.1 NoFall Data model

The DB is divided into separate sections, which corresponds to each major area that affects the risk of falling - as found in chapter 3, see appendix B for the full model. This is done to retain the modularity of the software, ensuring that it will be simple to update the DB in accordance with new functionality. Each section is divided into *Specification* and *Log*. Tables ending with *Spec* are used to store the information that e.g. a survey consists of, while the tables ending with *Log* are what the user and/or application registers during use.

### 7.4.2 User

The User specification will mainly be used to store risk calculations and results from the other different specifications. It will also be important to register the gender and age of the user, since this will also affect the results. The main idea with this specification is to enable developers an easy way to store data in a single table, to use it for data representation, or give warnings about risk levels etc.

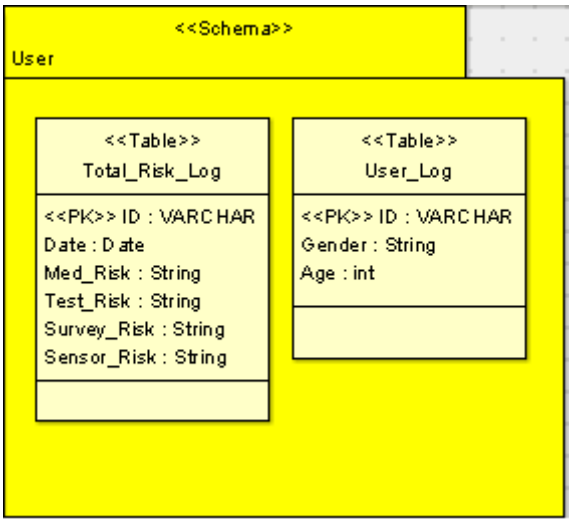



FIGURE 7.3: DB: User

*User_Log*

This table is for the storage of general information about the user. It will contain the age and gender, since they both will affect the risk of falling.

| Example of data stored in User Log: | |
|---|---|
| ID | 1 |
| Gender | Male |
| Age | 67 |

*Total_Risk_Log*

This table will be a log of all the risks registered in the DB. At given date intervals it will be of interest to store the current risk, so it becomes easy to see the progression. If it has gone up or down, if there are spikes at certain dates. By having this kind of data and possibilities, one can identify causes for risk, and mitigate them.

| Example of data stored in Ttl Rsk Log: | | |
|---|---|---|
| ID | 1 | 2 |
| Date | 10.05.14 | 21.05.14 |
| Med Risk | Low Risk | Low Risk |
| Test Risk | Low Risk | Low Risk |
| Survey Risk | Med Risk | Med Risk |
| Sensor Risk | Low Risk | Med Risk |

### 7.4.3 Risk Definitions

To make it possible to evaluate, compare, and collaborate data that is registered in the DB, the data has to be correlated to some risk definitions. This means that the different types of measurements the data will be gathered as, has to be related to risk definitions. Otherwise the different results might become unintelligible.

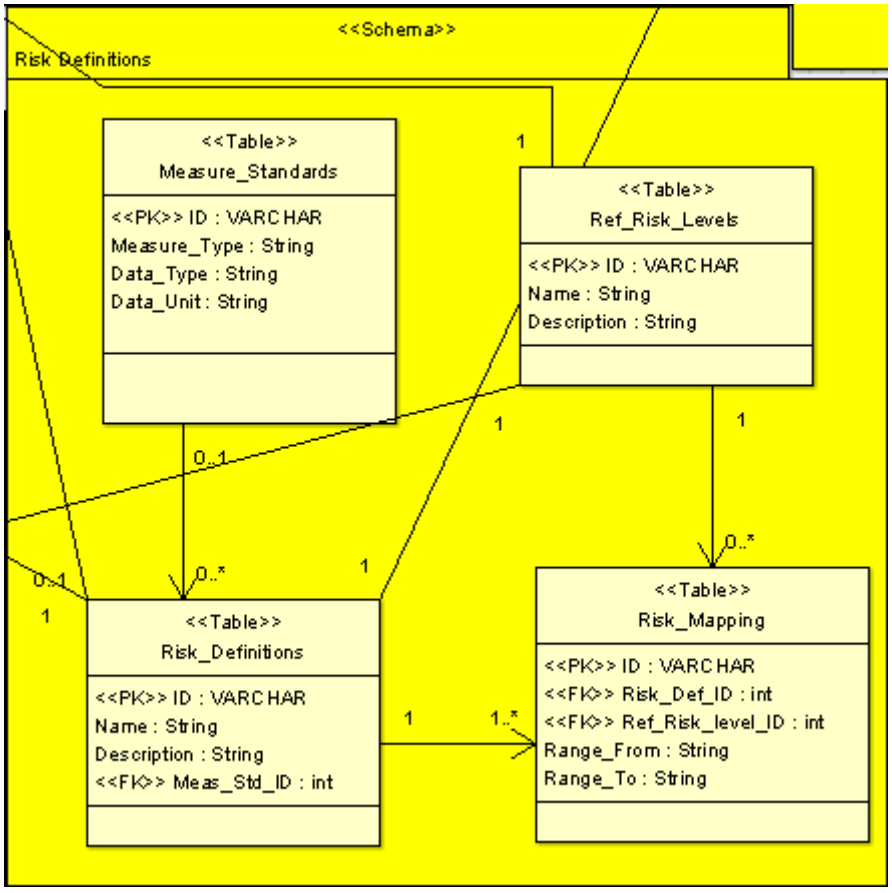



FIGURE 7.4: DB: Risk Definitions

*Measurement_Standards*

This table will contain measurement standards for different types of measurements that can be made, depending on the application. The measurement type (walking speed, step

count, medication consumption), the data_type (integer, long, string) and the data_unit it stores (e.g. meter/second, pressure).

| Example of data stored in Meas Std: | | | |
|---|---|---|---|
| ID | 0 | 1 | 2 |
| Measure Type | Walking Speed | Time Measurement | Medication |
| Data Type | long | long | int |
| Data Unit | meter/second | seconds | Number Of |

*Risk Definitions*

This table will contain the different types of risk definitions the measurement standards can have. Several different risks have been found through research, and these will be stored here (E.g. WalkingSpeed can consist of AverageWalkingSpeed, maximum walking speed, sprint).

| Example of data stored in Risk Def: | | | |
|---|---|---|---|
| ID | 0 | 1 | 2 |
| Name | Walking Speed | Medication | TUG Basic |
| Description | The average walking speed of the user can be used to calculate risk. | Number of medication taken. | Basic TUG test measurement. |
| FK: Meas Std | 0 | 2 | 1 |

*Reference Risk Levels*

This table will contain reference risk levels. This enables the developers to define some easy levels of risk for their gathered data. By doing this, it makes it easier for the application to indicate some level of risk for the user. Examples of such levels could be; No increment, Low, Medium and High or 0 - 1 - 2 - 3.

| Example of data stored in Ref Rsk Lvls: | | | | |
|---|---|---|---|---|
| ID | 0 | 1 | 2 | 3 |
| Name | No increment | Low | Med | High |
| Description | No increased risk | Minor increased risk of falling | — | — |

*Risk Mapping*

This table will contain a mapping between the reference risks levels and the risk definitions. A given measurement can often have a value range associated with it. If there is no range, or if it is hard to define a range, the attribute Range_To will be empty.

| Example of data stored in Risk Mapping: | | | | |
|---|---|---|---|---|
| ID | 0 | 1 | 2 | 3 |
| FK: Risk Def | 0 | 0 | 2 | 2 |
| FK: Ref Risk Level | 1 | 2 | 0 | 2 |
| Range From | 0 | 0,7 | 0 | 14 |
| Range To | 0,7 | 1,3 | 14 | - |

### 7.4.4 Medication

The aspect of medication and increased risk of falling is still quite unspecific. Most of the research can tell that certain medications might have an increased affect and they also say that if the patient is on more than four different medications, it is an indication of increased chance of falling. The medication specification tries to consider this, but still be able to support new findings from research. The *Medication Spec* table can be used to specify different types of treatments (e.g. heart-decease or depression).

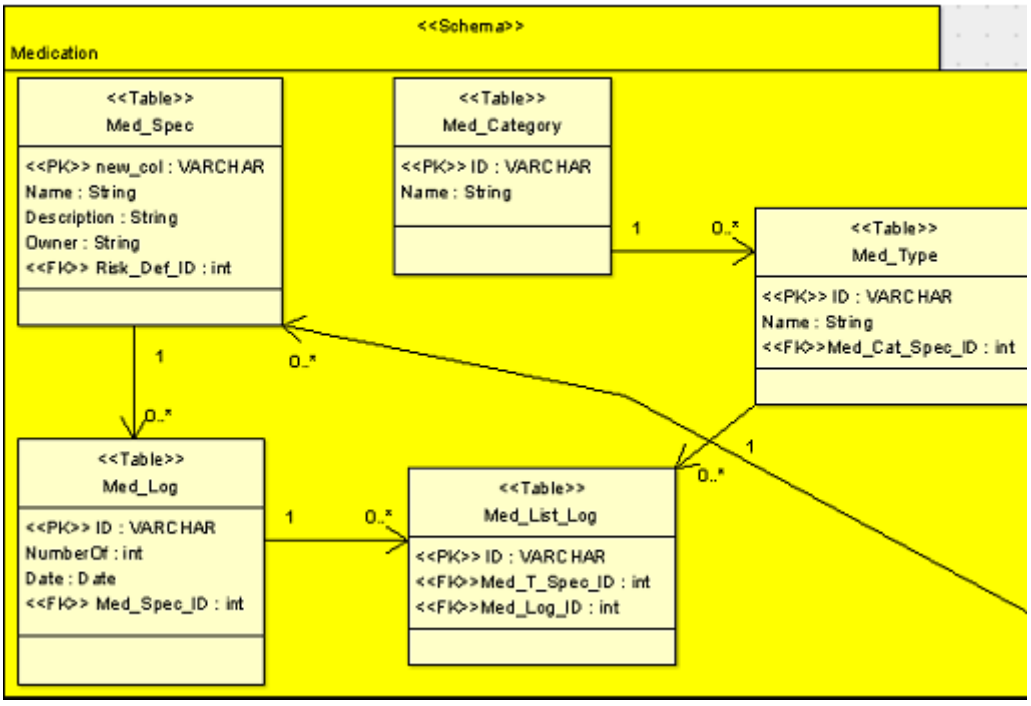



FIGURE 7.5: DB: Medication

*Medication_Specification*

This table will contain a medication spec for medications that has been identified as risk increasing. It will contain a name and description. It will also have a foreign key to a risk definition, which could be the number of medications that is taken. The Owner_ID attribute is to identify where the data came from.

| Example of data stored in Med Spec: | |
|---|---|
| ID | 0 |
| Name | Standard Med |
| Description | Standard recording of medication that calculates risk based on the number of medications, and not the specific medication |
| Owner | - |
| FK: Risk Def | 1 |

*Medication_Category*

This table is used for the storage of all the different categories of medication that have been identified as associated with increased risk of falling. (e.g. Antidepressant).

| Example of data stored in Med Cat: | | | |
|---|---|---|---|
| ID | 0 | 1 | 2 |
| Name | Antidepressant | Anticonvulsants | Antihistamines |

*Med_Type*

This table is used for the storage of all the different types of medication that have been identified as associated with increased risk of falling. (e.g. Amitriptyline). The foreign key binds the table to the category, so all the antidepressant medications are linked to this category.

| Example of data stored in Med Type: | | | |
|---|---|---|---|
| ID | 0 | 1 | 2 |
| Name | Amitriptyline | Bupropion | Brompheniramine |
| FK: Medication Category | 1 | 1 | 3 |

*Med_Log*

This table is used for the storage of how many medicaments the user are currently taking. Through research it has been shown that by knowing how many medicaments a person takes, one can deduce a risk of falling. This risk is associated with a risk measurement standard defined in the DB (see measurement for details). It is not needed to know which medications they are taking.

| Example of data stored in Med Log: | |
|---|---|
| ID | 0 |
| Number Of | 3 |
| Date | 10.05.14 |
| FK: Med Spec | 0 |

*Med_List_Log*

This table will store which medications the user is taking, if they know and registers this

through the application. The foreign keys link this list to the medication log, indicating from which date they are currently prescribed the different drug(s), and which drug(s) it is.

This table can be empty, because it is not necessarily that the user knows which medications they are taking. Many will just take what the doctor prescribes and be content with that.

| Example of data stored in Med List Log: | | | |
|---|---|---|---|
| ID | 0 | 1 | 2 |
| FK: Medication Type | 0 | 1 | 2 |
| FK: Medication Log | 0 | 0 | 0 |

### 7.4.5 Survey

Several different surveys have been made through different research in fall risk, e.g. Hendrich Fall Risk Model II contains simple questions with a point scores that will indicate risk of falling. To enable the users of NoFall to register surveys the want to use, they can specify all the different specifics such as the questions and the risk factors related to different answers.

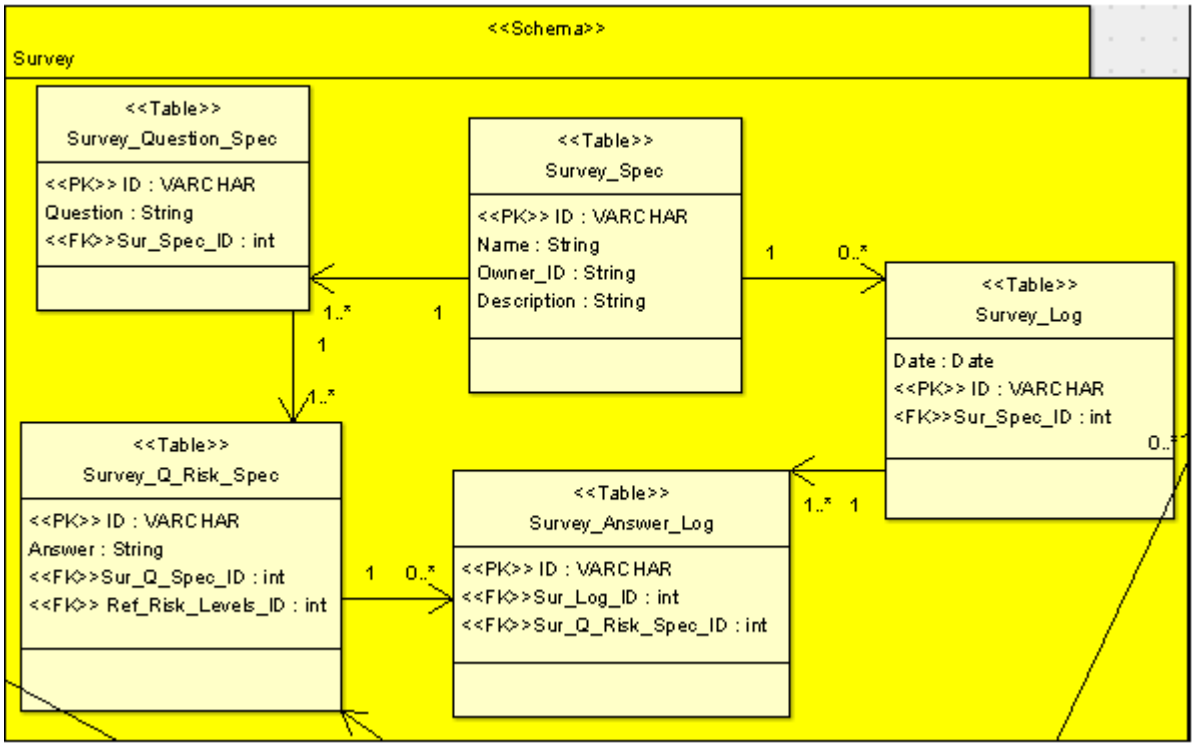



FIGURE 7.6: DB: Survey

*Survey_Spec*

This table will contain the specification for a given survey. This will consist of the name of the survey and a description. The Owner_ID attribute is to identify from where the data came from; if it was from an external or internal application.

| Example of data stored in Survey Spec: | |
|---|---|
| ID | 0 |
| Name | Hendrich Fall Risk Model II |
| Description | A model consisting of some risk categories |
| Owner ID | NoFall |

*Survey_Question_Spec*

This table stores all the questions associated with a given survey. Each survey can have one-to-many questions. The foreign key links the question to the correlating survey.

| Example of data stored in Surve Q Spec: | | | |
|---|---|---|---|
| ID | 0 | 1 | 2 |
| Question | Are you often confused or disoriented? | Are you depressed? | Do you experience dizziness/vertigo? |
| FK: Survey | 0 | 0 | 0 |

*Survey_Question_Risk_Spec*

Often the questions can have several different answers to a question (yes, no, maybe) and each answer has its own risk. This risk is associated with a risk level. This is linked to risk levels, because an answer will always have no increased or increased risk, and each question will affect this. The foreign key link the risk and answer to the correct question.

| Example of data stored in Survey Q Rsk Spec: | | |
|---|---|---|
| ID | 0 | 1 |
| Answer | Yes | No |
| FK: Ref Risk Levels | 2 | 0 |
| FK: Survey Question | 2 | 2 |

*Survey_Log*

This table contains the date at which the user completed the given survey.

| Example of data stored in Survey Log: | |
|---|---|
| ID | 0 |
| Date | 14.05.14 |
| FK: Survey Spec | 0 |

*Survey_Answer_Log*

This table contains all the answers the user made. The foreign keys link the answers to

the date at which they were given. It also links the answer the user made in the survey to the risk table, so it is easy to look up the risk for that given answer.

| Example of data stored in Survey Ans Log: | |
|---|---|
| ID | 0 |
| FK: Survey Question Risk | 1 |
| FK: Survey Log | 0 |

### 7.4.6 Sensor

When using sensors in relation to fall risk detection, several different types are used. These can be accelerometers, gyroscopes, pressure readers etc. The sensor specification enables developers to specify which sensor they use, register relevant info such as accuracy (this is so other users can see at what degree they can trust the data). They can also save all the data they gather with a given sensor.

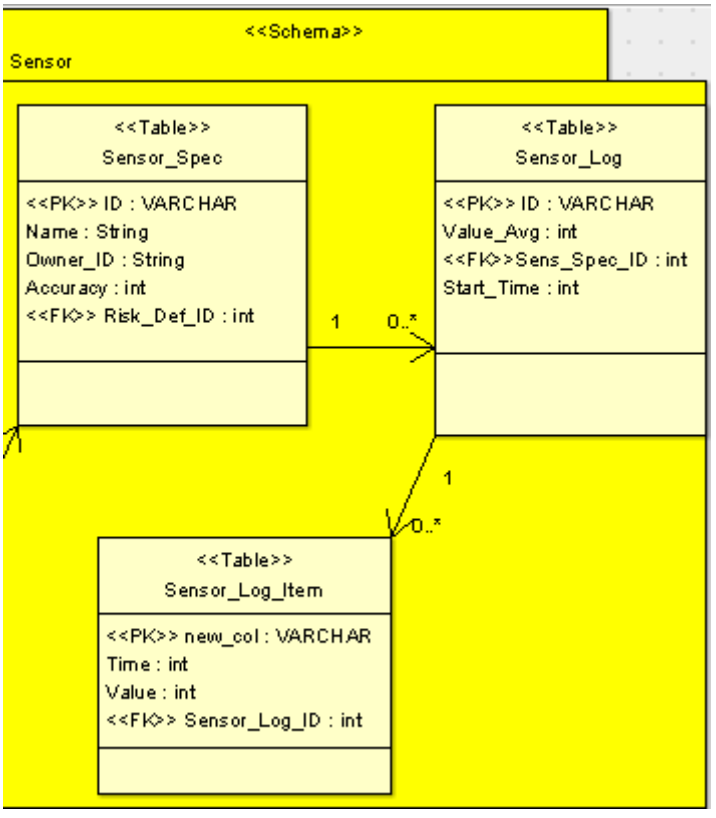



FIGURE 7.7: DB: Sensor

*Sensor_Spec*

This table contains all the information about a given sensor. It will have the name of the sensor (e.g. accelerometer). The sensor will also gather some type of data, which

will be identified through the foreign key Risk_Def_ID. The Owner_ID attribute is to identify where the data came from; if it was from an external or internal application. The type of data that is stored will have to have some sort of risk definition for the measurements; this will be stored in the risk definition specification.

| Example of data stored in Sensor Spec: | |
|---|---|
| ID | 0 |
| Name | Accelerometer |
| Owner ID | NoFall |
| Accuracy | 95% |
| FK: Risk Def | 0 |

*Sensor_Log*

This table will store the average value (Value_Avg) that is recorded by the sensor if this is of interest (e.g. the average movement speed of the user over the last week). It will contain when the recording started. The foreign key ensures that the values that are stored here are linked to the correct type of sensor.

| Example of data stored in Sensor Log: | | |
|---|---|---|
| ID | 0 | 1 |
| Value Average | 0.75 M/S | 0.72 M/S |
| FK: Sensor Spec | 1 | 1 |
| Start Time | 10.05.14:10.45 | 16.05.14:12.30 |

*Sensor_Log_Item*

This table will store each reading that is made during the logging. It will contain the value and time of the reading. Based on the data stored here, the average can be calculated and stored in the Sensor Log table. Having each item stored like this, one can for example see changes in movement over two months by comparing the data.

| Example of data stored in Sensor Log Item: | | |
|---|---|---|
| ID | 0 | 1 |
| Time | 10.05.14:10.45 | 16.05.14:12.45 |
| Value | 0.75 M/S | 0.72 M/S |
| FK: Sensor Log | 0 | 0 |

### 7.4.7 Test

To determine risk of falling there exists many different types of tests that a person can take (e.g. timed-up-and-go(TUG) test or sit-to-stand test). Some consists of a test and questions related to the test (e.g. for the TUG: Did you use more than one attempt

to rise out of the chair?). The test specification covers the most relevant attributes identified from research analysis. Enabling them to specify their own tailored tests, or register known established tests.

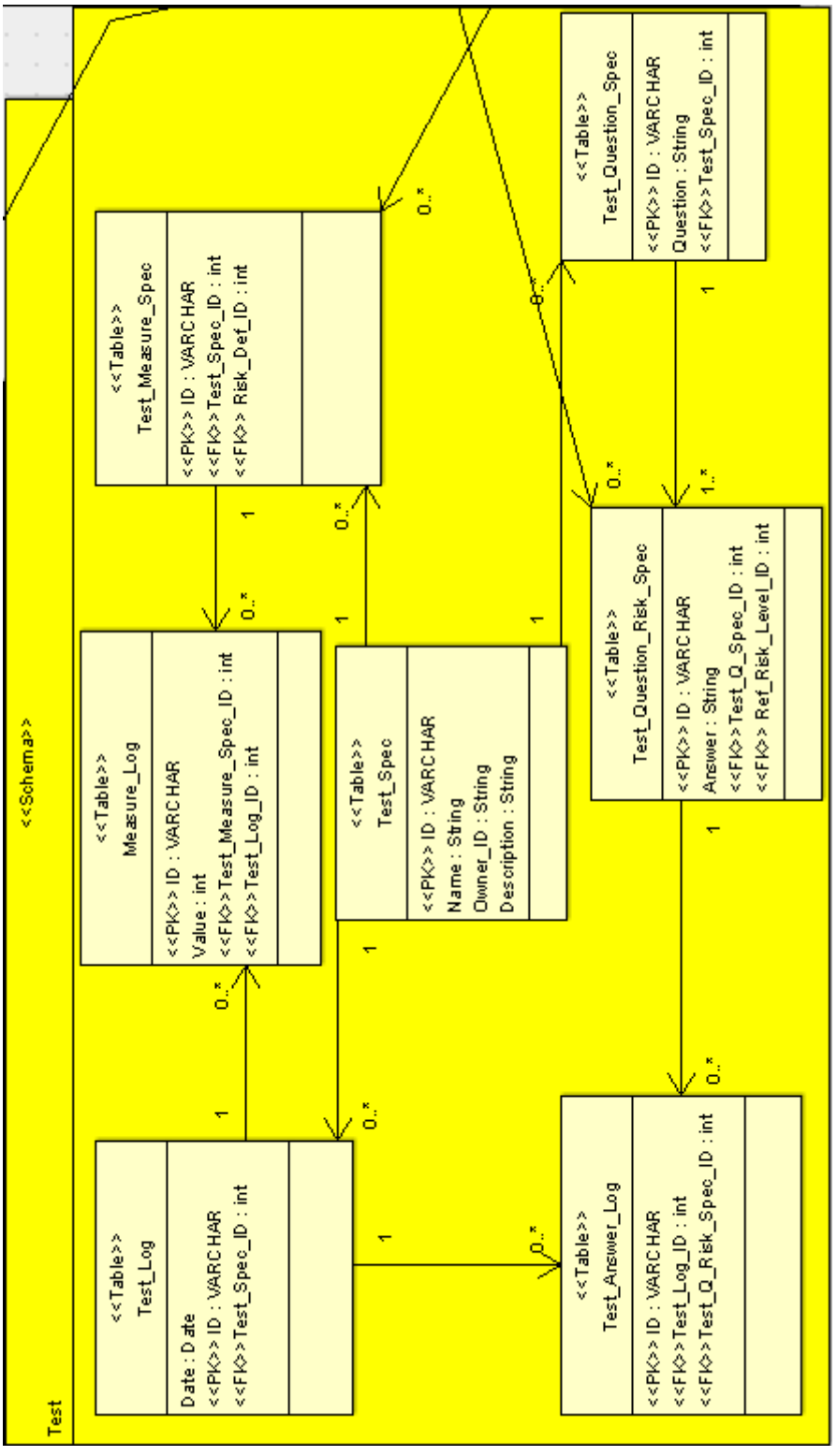



FIGURE 7.8: DB: Test

*Test_Spec*

This table will contain the name of a specific test. It will also store an Owner_ID which will indicate who stored the data for this certain test; if it was an internal or an external application.

| Example of data stored in Test Spec: | |
|---|---|
| ID | 0 |
| Name | TUG |
| Owner ID | NoFall |
| Description | A standard TUG test... |

*Test_Measure_Spec*

Many tests will have some form of measured values (e.g. the TUG test will record the time a person takes to walk 3 meters, turn, walk back and sit down). This table will contain which type of measure is to be recorded for this test, what kind of data type it records and what kind of data unit. These sorts of measures will follow defined risk definitions that are stored in a risk definition table, the foreign key will point to this table. The foreign key Test_Spec_ID ensures that the measure is linked to the correct test.

| Example of data stored in Test Meas Spec: | |
|---|---|
| ID | 0 |
| FK: Risk Def | 1 |
| FK: Test Spec | 0 |

*Test_Measure_Log*

The test measure log will contain the results of an executed test that has some sort of measurements.

| Example of data stored in Test Meas Log: | |
|---|---|
| ID | 0 |
| Value | 13 seconds |
| FK: Measure Spec | 0 |
| FK: Test Log | 0 |

*Test_Question_Spec*

Some tests also have questions that will be asked in relation to the test. (e.g. in the TUG test, it can ask if the person used one, two or several attempts to rise out of the chair).

*Test_Question_Risk_Spec*

The questions will often have several answer options, and each option might have a risk

| Example of data stored in Test Q Spec: | |
|---|---|
| ID | 0 |
| Question | Did you take several attempts to get out of the chair? |
| FK: Test Spec | 0 |

value. This risk is associated with a risk level defined in the DB (see measurements for details). This table will contain such information, where the foreign key ensures that the risk and answers are connected to the correct question.

| Example of data stored in Test Q Risk Spec: | | | |
|---|---|---|---|
| ID | 0 | 1 | 2 |
| Answer | No | One | Two |
| FK: Ref Risk Level | 0 | 2 | 3 |
| FK: Test Question ID | 0 | 0 | 0 |

*Test_Log*

The test log table will contain at which date the test was completed.

| Example of data stored in Test Log: | |
|---|---|
| ID | 0 |
| Date | 15.05.14 |
| FK: Test Spec | 0 |

*Test_Answer_Log*

This table will contain all the answers the user gave during a test. The foreign key will ensure it is connected to the correct test and date (Test_Log table).

| Example of data stored in Test Ans Log: | |
|---|---|
| ID | 0 |
| FK: Test Q Risk Spec ID | 1 |
| FK: Test Log ID | 0 |

## 7.5 Chapter Summary

In this chapter the main driving points and design for the architecture is presented. Stakeholders and concerns that affect the choices made in relation to architecture and design are explained. The goals that the system is aimed to achieve are listed. Finally, the database model design and rationale are described.

To avoid technical debt, which is the consequence of poor software design, it is important to begin the design process of the software architecture and database before the

implementation begins, to avoid coding from scratch, blind and bewildered. Since the development is an iterative process, changes will occur to the design, almost at each new sprint. Good design takes into account that changes will occur during the lifetime of the project, and they need to be easily addressed.

The entire process removes normal pitfalls developers find themselves in, if they do not follow any sort of plan or design. Planning and designing will also remove some unneeded changes that would otherwise occur, since they have already been predicted and overcome. One final benefit is that the risk of having to revert to previous versions of the software is diminished, because the chance of starting to develop something that will ruin the platform or be totally wrong is small.

# Chapter 8

# Implementation

This chapter details the strategy followed during the implementation. The tools and technologies that were used will be described, and the rationale for their usage explained. The process for how each major part of the platform was implemented, with issues, will also be presented.

## 8.1 Strategy

As mentioned in chapter 2 this project followed an adoption of Scrum. The implementation tasks were divided as:

- Requirements: These are larger parts of the platform, which also has to cover the requirements specified in chapter 5. E.g. complete the implementation of the *Survey Spec* in the DB.
- Requirement tasks: Tasks are smaller portions of the entire requirement. The requirements are divided into small, manageable assignments, which is never larger than a workday. E.g. write the SQLite code for the table *Survey Log*.

All the requirements and tasks were gathered in the backlog, see figure 8.1 for some of the requirements set for the implementation. Each requirement was then divided into several tasks, see figure 8.2 for the tasks for requirement 1. For each sprint, the amount of time to complete a requirement were estimated, and moved to *active*. Based on how

many hours of work was completed in comparison to how many hours in a work week were available, the burn down was calculated.

| 1 | Requirements | Estimated Time in hours |
|---|---|---|
| 2 | The Survey Specification has to be accesses through a Content Provider to make it possible for several applications to access the same | 10 |
| 3 | The Test Specification has to be accesses through a Content Provider to make it possible for several applications to access the same | 10 |
| 4 | The Medication Specification has to be accesses through a Content Provider to make it possible for several applications to access the same | 10 |
| 5 | The User Specification has to be accesses through a Content Provider to make it possible for several applications to access the same | 10 |
| 6 | The Standards Specification has to be accesses through a Content Provider to make it possible for several applications to access the same | 10 |
| 7 | The Sensor Specification has to be accesses through a Content Provider to make it possible for several applications to access the same | 10 |
| 8 | To make it easier to use the content provider, each specification in the DB has to have a Content Contract: Survey | 10 |
| 9 | To make it easier to use the content provider, each specification in the DB has to have a Content Contract: Test | 10 |
| 10 | To make it easier to use the content provider, each specification in the DB has to have a Content Contract: Medication | 10 |
| 11 | To make it easier to use the content provider, each specification in the DB has to have a Content Contract: Standards | 10 |

FIGURE 8.1: Partial Backlog

| Requirement | Task | Estimated Time in hours | Status |
|---|---|---|---|
| 1 | Write inserts | 2 | Completed |
| | Write updates | 2 | Completed |
| | Write queries | 2 | Completed |
| | Write deletes | 2 | Completed |
| | Write Uris for urimatching | 2 | Completed |

FIGURE 8.2: Backlog - Requirement 1 tasks

Using an iterative methodology with rigorous follow-up each sprint made it possible to complete the implementation in time. This also mitigated the risk of not managing to complete in time. Another risk mitigation that was used, where the usage of a personal rule in regards to the report: *Each week a minimum of four pages was to be written, or an equal amount of time to revise, restructure or plan new chapters was to be directed at the report.* This removed the danger of becoming to focused on the development,

and ignoring the report. Other risks were taken into account, such as sickness, personal issues that obstructed progress, delay in receiving/obtaining vital assets.

Testing was performed on two different Android devices: LG-P990 and Huawei-U9000. Since the platform only provides back-end functionality, and the test applications written are only made to provide developers with examples of how to use the platform, the need to test if the GUI scaled well on many screen sizes did not exist. That is why the use of such a limited number of devices was sufficient.

## 8.2 Tools & Technologies

### 8.2.1 Java & SQLite

#### Java

NoFall was developed using Java programming language, this relates to the wishes of my advisor that the platform was to be developed for Android devices. Android would also have been my choice to develop for, since it has the largest community, and it is focused on open source.

Java [38] is an object-oriented programming language first released by Sun Microsystems in 1995, the languages syntax is based on C and C++. Java is the programming language that Android applications are based on.

#### SQLite

*SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. SQLite is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file.* [39]

Working with sensitive and private information in relation to the storage of fall risk data, there were no possibilities for any external storage. This is because a major process of applying for permission to store this kind of information, and the need for such functionality did not outweigh the tedious process of getting it granted, or if it would be granted.

This led to the usage of an internal database for each mobile phone and no external web-storage. SQLite works great with android, providing an easy to use API, quick read/writes and since SQLite requires little to no administration, it works well for internal databases that have to work unattended.

### 8.2.2 Eclipse

I did the development in Eclipse with Android SDKs installed. I chose Eclipse as my development tool because it is familiar, easy to use and has good support for Android and Java development. It provides easy debugging support, with great logging that can be viewed with Logcat.

Eclipse [40] is one of the most used open source IDE. By supporting various plug-ins the Eclipse makes it possible to develop mobile applications, web-applications and even develop in different languages then Java.

By using the Android SDK with Eclipse, the API libraries and developer tools needed to build, test and debug Android applications in Eclipse is provided. This includes *Logcat*, which is an excellent way to use logging to track data flow for debugging.

Another IDE that I could have used was Android Studio, which is an IDE based on IntelliJ IDEA, and will be the official IDE for Android when it is released. Android Studio is still in Beta, which is the reason I decided not to use it, since I prefer not to risk bugs and problems related to an unfinished product.

### 8.2.3 SQLitebrowser

To view the SQLite DB to verify that the correct tables were created, SQLitebrowser [41] was used. This is a light GUI editor for SQLite DBs.

### 8.2.4 TeXstudio

The report was written in TeXstudio [42]. This is an integrated writing environment for creating LaTeX documents. It provides several features to make it easier to manage a document, such as syntax-highlighting, reference checking and integrated viewer.

### 8.2.5 Android

Android is an OS designed primarily for touch screen devices such as tablets and smartphones. The OS is released under the Apache license, which means it is open source and anyone can modify it or develop their own products. The user interface focuses on logical interaction such as touch, swiping and pinching. This makes the interaction suitable for novice users, and the user learning curve is minimal.

## 8.3 Coding

This section will give a short description of the implementation, and issues that occurred during this process.

### 8.3.1 SQLite Database

The implementation of the DB was straightforward, by following the design from chapter 7, and looking up examples and best practices from the SQLite homepage. The specifications with its corresponding tables was implemented in a systematic order, completing one section at a time, and testing it to see that the tables were created correctly by using SQLitebrowser. Simple test queries were written to test inserts, deletes and updates.

**Issues**

An issue that occurred during the implementation was syntax mishaps with the strings that create tables (see figure 8.3 for an example). These are not forgiving when it comes to syntax, and minor errors create problems. They are also hard to find, since the IDE does not detect any error when it is a string of text.

Another issue was that it became troubling to keep track of the entire DB as it continuously grew and the design changed. It happened that some changes to the design did not get into the current build at the correct time, because it was overlooked.

```
private static final String DATABASE_CREATE = "create table "
        + MedicationType.TABLE_NAME
        + "("
        + MedicationType._ID   + " integer primary key autoincrement, "
        + MedicationType.NAME + " text not null, "
        + MedicationType.FK_RISK_STAND_MAP + " integer, "
        + MedicationType.FK_MEDICATION_CATEGORY + " integer, "
        + MedicationType.CREATED_DATE + " integer,"
        + MedicationType.MODIFIED_DATE + " integer,"
        + " FOREIGN KEY ("+MedicationType.FK_RISK_STAND_MAP+") REFERENCES "
        +StandardsRiskMap.TABLE_NAME
        +" ("+StandardsRiskMap._ID+") ON DELETE CASCADE"
        + " FOREIGN KEY ("+MedicationType.FK_MEDICATION_CATEGORY+") REFERENCES "
        +MedicationCategorySpec.TABLE_NAME
        +" ("+MedicationCategorySpec._ID+") ON DELETE CASCADE );";
```

FIGURE 8.3: SQL String example

### 8.3.2 Content Provider

The content provider was developed by following the guidelines and looking at the example applications provided by Android. For each database specification that was completed, the content provider was updated, so it could be accesses through it.

#### Content Contracts

When the DB and Content Provider version 0.1 was completed, the content provider was re-factored. The result was content contracts for each specification and its tables. The content contracts for the provider are utility classes that are created for the convenience for developers. A contract class defines constants that help applications work with the content URIs, column names, intent actions, and other features of a content provider.

#### Issues

An issue with the content provider was the sheer size of it. It became rather hard to establish an overview of it at times, especially if it had been a while since I had been working with it. This problem became less when I managed to form good contract classes, which contains much of the information that was previously in the content provider.

There was also some initial problems, related to inexperience, such as not fully understanding what or how the Uri and Urimatching, and authority granting worked.

### 8.3.3 Example Application

Example applications was developed to ensure that the DB and the content provider fulfilled the requirements and scenarios established, and to provide the potential developers that will use NoFall with some examples. An easy and intuitive way to become familiar with a platform or framework is to view examples made by the creator(s).

The focus of the example applications was to show possible applications that could use the platform, give inspiration or ideas, and not on designing good and robust applications. Therefore, several short cuts were taken during the development of them; such as good GUI design was neglected.

- A pedometer application made by Levente Bagi [43] was incorporated and customized to work with the NoFall platform. This was to demonstrate that existing complete applications could be adapted to the NoFall platform.
- A graphical displaying application was developed to have some way to present the data gathered from the pedometer application. This application was developed to demonstrate the power of sharing data, and how simple the implementation of an application can be - other applications and APIs handled the majority of the complexity.
- A simple listing of medicaments that can be selected. This application was developed as more a suggestion of functionality that applications can have, rather than demonstrating the NoFall platforms functionality.
- A widget application was developed, this shows data based on the pedometer application as well, but it could be used to display e.g. the current level of risk of falling based on several risk factors.
- A simple testing application was developed. It is a simple TUG-testing application that explains the TUG-test and how to perform it. It also saves the data from the test to the NoFall platform.

#### Issues

An issue, or more a challenge arose during the development of the different example applications; the process of figuring out which applications to develop. They had to be

relevant to the usage of NoFall and in turn be relevant to fall prevention or risk detection. They also had to demonstrate how to use the content provider with its contract classes.

Initially the TUG-test application was intended to be semi-automatic. The user was supposed to be able to press start, place the phone in a pocket, wait for the phone to beep and begin the test. The phone would automatically beep again after 3 meters, signalling for the user to turn and walk back. It would also notice when the user stopped moving, saving the result of the test. This became too complex and time-consuming, and in the end there was not enough time to make it. Therefore, the TUG-test application is now a glorified stopwatch that saves the timed results.

## 8.4 Chapter Summary

In this chapter, the strategy used for the implementation and the writing of the report has been explained. Which main tools and technologies that were used has been discussed, and the main parts of the implementation and related issues has been presented.

When approaching a new project, selecting the correct tools and technologies is an important process. Therefore putting some time into research and comparison can in the end improve the result, make life easier and earn the time put into finding the correct tools back.

A well-defined and rigorous strategy, that has weekly and daily goals, ensures continuous results. Using an iterative methodology will push the developer to always improve and enhance the product, pushing out new or improved functionality on time. This helped me to avoid what is called a *zero-day*, which is a day where nothing is achieved on the dissertation. Everyone can have a bad day, but a day that is *non-zero* is easily achievable, and this accumulates over such a long process and leads to a lot of work getting done.

By breaking the entire solution up into smaller, more manageable sizes (such as the requirement tasks), makes it more plausible to execute. Another benefit of such a process is that to be able to define these requirements and tasks, the entirety of the solution and process must be thoroughly thought through. By doing this, one gains a

better overview of the task at hand, which can lead to changes, improvements or removal of unnecessary complexity.

# Chapter 9

# Results and Evaluation

This chapter will present the final implemented solution. An evaluation of the hypotheses presented in chapter 1.2 relative to the final implemented solution. The evaluation of the results is presented. Finally, some of the major challenges that occurred during the dissertation are listed.

## 9.1 Final Solution

All the implementation requirements that were set during the planning and designing phase for the NoFall platform was completed. This includes the implementation of the database foundation and the content provider that grants access to it. The test applications had their own requirements and tasks, in which several of the planned test applications were implemented, albeit not all of them because of lack of time.

### 9.1.1 Database

The database was implemented following the design from chapter 7. The DB covers each major important aspect of fall risk research discovered and presented in chapter 3.

### 9.1.2 Content Provider

A content provider was implemented following Androids tutorial and guidelines. It allows for access to the entire DB with read, inserts, deletes and updates. To ensure

that it is simple to use and user-friendly, each specification with its tables has a content contract class. These classes contain constants that are used to reference the different table names, attributes and URIs.

### 9.1.3 Example Applications

The example applications implemented are listed below:

- Pedometer - an application that records number of steps and calculates the movement speed.
- Graphical display - this application displays data in a normal graph, and in a pie chart.
- Medication registration - this application has many different medications and categories that have been identified with correlation to fall risk. It shows how one could list these, register them to a user and calculate risk from it.
- TUG-Test - this application is a simple TUG-test.
- Widget - a simple widget with some queries for different data to present.

## 9.2 Hypothesis

The focus of this thesis has been fall prevention applications with the hypothesis:

**Hypothesis**: Having a platform for fall detection applications will make it easier for future developers to build high-level applications that are well defined and achieve a higher level of quality.

Through my work with this thesis; designing and implementing, it is my opinion that providing future developers with a platform will increase the speed at which prototypes or even fully functional applications made for different application markets are developed. This will also make it easier for the developers to build high-level applications, since they can focus solely on functionality, usability and other aspects, than the process of defining the underlying storage and how this is supposed to be designed.

Based on my results, I believe that the hypothesis is true. This is however my opinion, which is inevitably biased, based on my experience using the NoFall platform to develop example applications. Further work, testing and feedback by other developers are mandatory to be able to fully evaluate the hypothesis.

## 9.3 Research Questions

### Main Research Question

What is the effect of a platform on the ease and quality of developed applications?

*How it is answered in the thesis:*

The platform has several effects that have been identified through the work on this thesis (See chapter 5.3 for a demonstration of these findings).:

- It reduces the number of lines of code that has to be written, which translates to shorter development time, or more time to be used on improving other parts of the application.
- It allows for sharing of code (e.g. queries) - which reduces work time.
- It is easy to adapt existing solutions to the platform. This is important, so that existing solutions do not have to go through a big overhaul to adapt to the platform.
- The data that is stored in the DB can easily be shared between different applications, making collaboration easier and more accessible.

### Secondary Research Questions

*Question one*

What are common and easily measurable risk factors used in fall risk assessment?

*How it is answered in the thesis:*

From the literature analysis (see chapter 3 for details) several factors where identified - a short summary of some of the important ones are listed below:

- Medication: What type of medication (e.g. antidepressant) and how many different types of medication the person takes will affect fall risk

- Poor strength and balance: If the person is physically weak, the risk of falling will increase. The implication of the fall will also be more severe with low physique.
- Psychological: The state-of-mind of the person will affect the risk of falling (e.g. fear of falling, nervousness).

*Question two*

How can one design a meaningful data model based on the knowledge of which patient information/data is useful for predicting the level of risk of falling?

*How it is answered in the thesis:*

The most important aspects of modelling data, is identifying what kind of data that will be stored - and how it relates to each other. Based on the analysis done in chapter 3, the sort of data that is typically gathered was identified. This analysis also helped to form relations between the different data, seeing how they relate to fall risk.

With the knowledge of what kind of data and how it should relate to each other, the design and architecting could begin. The database follows best practices and guidelines from the SQLite homepage. See chapter 7 for the full detailed design of the database.

*Question three*

How should the technical architecture of a platform be designed and built?

*How it is answered in the thesis:*

When building a platform or a framework, the focus is on the developers and not end-users, and this is something to take into consideration when designing the architecture.

The analysis done in chapter 4 showed how other researchers and developers had solved their own problems related to fall risk and detection. The different solutions provided knowledge that helped me in the design and architecture of the platform. Such as the framework SPINE [32] was an excellent way to see how the design and architecture of a large-scale project for a framework was done.

It is important that the platform covers use cases, and if a certain use case is not supported, it should be easy to implement or adapt the platform to support it. This can either be done by the creator of the platform, or if the developer writes an extension class/method by themselves.

## 9.4 Evaluation

NoFall has three problems that it attempts to improve in regards to development; time, interoperability, and reuse. It wants to reduce the time it takes to finish an application. It tries to make it easy for systems to work together and share data. Finally, it wants to increase the amount of code that can be reused. The three problems are all closely related, and by improving one of them, the others will benefit. Such as reuse of code between solutions could also be seen as interoperability.

To evaluate how successful the platform is, values that the NoFall platform could be measured against were set for the three issues.

- Time: Number of hours saved when developing.
- Interoperability: How easy it is to share data between applications.
- Reuse: How much code is reusable.

**Problem to Improve - Time**

The time to develop an application is reduced when using the NoFall platform. If the use case fits the platform correctly, almost half of the solution is already in place and ready to be used.

This became evident during the development of the different test applications. When I measured the amount of work and time that went into the different test applications, I saw a big improvement in time consumption versus other projects I've worked on when developing mobile applications.

**Problem to Improve - Interoperability**

The NoFall platform makes it easy to share the data between applications. The Content Provider API is easy to learn and has several examples online on how to use it. Therefore, it is pretty straightforward to write an application that access the NoFall DB trough the API.

This showed to be simple when I was developing the graphical display application. When I wanted to get some data, it was simple to query the DB and retrieve the data the pedometer application had stored.

However, from the evaluation process it was not evident how easy it would be to make sense of the data that is shared, even though it is simple to access. This comes from lack of other testers than myself, and the inability to have access to data that was retrieved by an outside source. Therefore, it is hard to determine if interoperability has become easier.

**Problem to Improve - Reuse**

Many of the queries used to access the database can be reused by several applications, especially queries that are only interested in e.g. all the data stored in certain tables. Several inserts are also reusable, since they will be the same, only change which values that are inserted. This makes it possible to copy queries written by other users, reducing the amount of time needed to develop their own solution. If one views the entire NoFall solution as "reusable code", the amount of code that is reusable multiplies for each application that uses the platform.

## 9.5 Challenges

During the dissertation, I encountered several challenges, both minor and major. The challenges with the largest impact and how I approached them is listed below:

- **Changed direction of the dissertation**

  The initial goal of the thesis was to explore the possibilities and develop an application for fall prevention. This application was supposed to be stand-alone, diminishing or removing the need of having to interact or get help for healthcare professionals.

  From guidance from my advisor, the shift to explore and rather develop the NoFall platform came relatively early on in the first semester, but it still required some adjustment and re-evaluation of several aspects.

  I accomplished this by researching more into platforms and frameworks, and incorporating this knowledge with the knowledge I had already acquired. It was hard to develop a platform since this was the first time I had ever done this. In addition, it was rather different from the normal application thought process.

- **Designing the database**

  The database went through three larger overhauls. The overhauls were motivated by the fact that I felt the design was moving in the wrong direction. It was very hard to get a firm grasp on what kind of design would accomplish the desired outcome. The biggest issue was how risk definition and measurements should be handled. What kind of data would come in? What sort of risk would this data indicate? How should this data relate to the source? How should it be stored, as to make sense for other users of the NoFall platform?

  This issue was addressed by getting feedback from my advisor and my father (Helge Johansen) on the current design. I would not say this issue was entirely solved, because I believe there is room for a lot of improvement on the current database design.

- **Example Application**

  - **Pedometer**

    The customization of the pedometer was challenging, because it was implemented targeting an older release of Android than what I was targeting. This lead to some implications, such as several of the methods that existed was *deprecated*, and had to be changed. These changes led to ripple effects in the code, leading to new bugs and errors.

  - **TUG-test**

    The example application for the performance of a TUG-test was supposed to be able to start and stop the timer automatically. The user was supposed to press start test, the timer would then start when it noticed movement, make some noise after 3 meters - to signal the turn, and stop when no motion was noticed.

    This was not implemented, because the sensor in the mobile phone was to prone too false-positives, and the results became untrustworthy. This relates to my lack of skill with writing algorithms in relation to human motion. Therefore, the application is now a manual press start/stop timer.

# Chapter 10

# Conclusion & Further Work

## 10.1 Conclusion

The goal of this dissertation was to investigate challenges and solutions related to fall prevention with focus on mobile applications. Based on the investigation, an issue of research was formulated; *(1) Having a platform for fall detection applications will make it easier for future developers to build high level applications that are well defined and achieves a higher level of quality.* This led to the formulation of one main research question and three secondary research questions, which helped to guide the direction of the thesis:

**Main Research Question**

What is the effect of a data-storage platform on the ease and quality of developed applications?

**Secondary Research Questions**

*Question one*

What are common and easily measurable risk factors used in fall risk assessment?

*Question two*

How can one design a meaningful data model based on the knowledge of which patient information/data is useful for predicting the level of risk of falling?

*Question three*

How should the technical architecture of a platform be designed and built?

In order to achieve this, a platform was implemented which provides the foundation for a fall related application. It provides a well-defined database based on the existing research in the field of fall prevention with a content provider to access it.

To make it easier to use and diminish the learning curve, some example applications were developed to show the usage of the platform. They are all related to fall prevention to increase their relevance with the platform, and to serve as inspiration for application ideas. The applications are small and simple, focused on showing what I have thought of as "high probability use case", and not on best practices on application implementation with high level of complexity and completeness.

The lack of real-life usage of the platform is the largest downside, limiting the possibilities to uncover faults, bugs, needed or unneeded functionality. This could in turn have led to a better result for both the report and the NoFall platform. Although this is the case, my experience with the platform when developing test applications has been positive, and I believe it is a good approach.

Solely I, the developer of Nofall, did the evaluation without any outside sources affecting the results. Therefore, the results are biased and plausible at best. This was done because there was no available and suitable test person, someone who was going to develop an application related to fall prevention that would take the time to test a prototype solution. This is understandable, because it is not always desired to test someone's prototype and impeding their own research when it is not a necessary risk. It should be pointed out that this is acceptable because the result of this thesis is a proof-of-concept, and not a final solution.

My results led to strong implications towards the hypothesis being true. This being the case, I feel that the goals set for this dissertation has been fulfilled, and that my result can contribute to further research and work in the field of fall prevention with focus on mobile applications.

## 10.2 Further Work

The content provider is at the moment the only access-point to the DB, and it gives access to the entire DB. It is not necessarily the case that a developer will have the need

to access the entire DB, and therefore the development of new content providers, which gives access to less, could be useful. This could come in the form of read-only access, or access to a limited number of tables or specifications. This could be a performance improvement and it could reduce the complexity the new developer has to adhere to. The security of data could also be an improvement, since less data or actions to manipulate the data are available.

The DB is at the moment satisfying and ready to be put into use. The DB is currently at version 1.0 and will be updated and improved as user feedback is given, and as the maintainers matures, learns and understands more in relation to fall risk and prevention. New use cases might also emerge during the lifetime of the platform, and the DB must be modified accordingly as to support the eventual new usages.

One change that was considered, but excluded from the current DB version was a Medication mapping table between Medication Specification and Medication Type for more detailed medication risk definitions. This was omitted because as the current research stands, it is rather vague on what kind of medications that would lead to increased risk of falling, or if there is something that can be done (would it be more beneficial to reduce medication vs. the reduced risk of falling?). This is something that should be added to the design in a later phase.

Several example applications have been developed, such as a pedometer, graphical display of data and a widget, but there are certainly room for more good test examples. There will definitely be problems and new usage areas for NoFall that will become evident when other developers start using it. Therefore, it is important to update the existing example applications, and develop new ones based on feedback from users, making sure the applications are current and up-to date. This will benefit and ensure that it will be easier for the next developer.

Porting the platform to a cross-platform solution could also benefit the research and developer community. That will make it available to several mobile platforms, and thereby making it possible to develop for more devices, reaching more users.

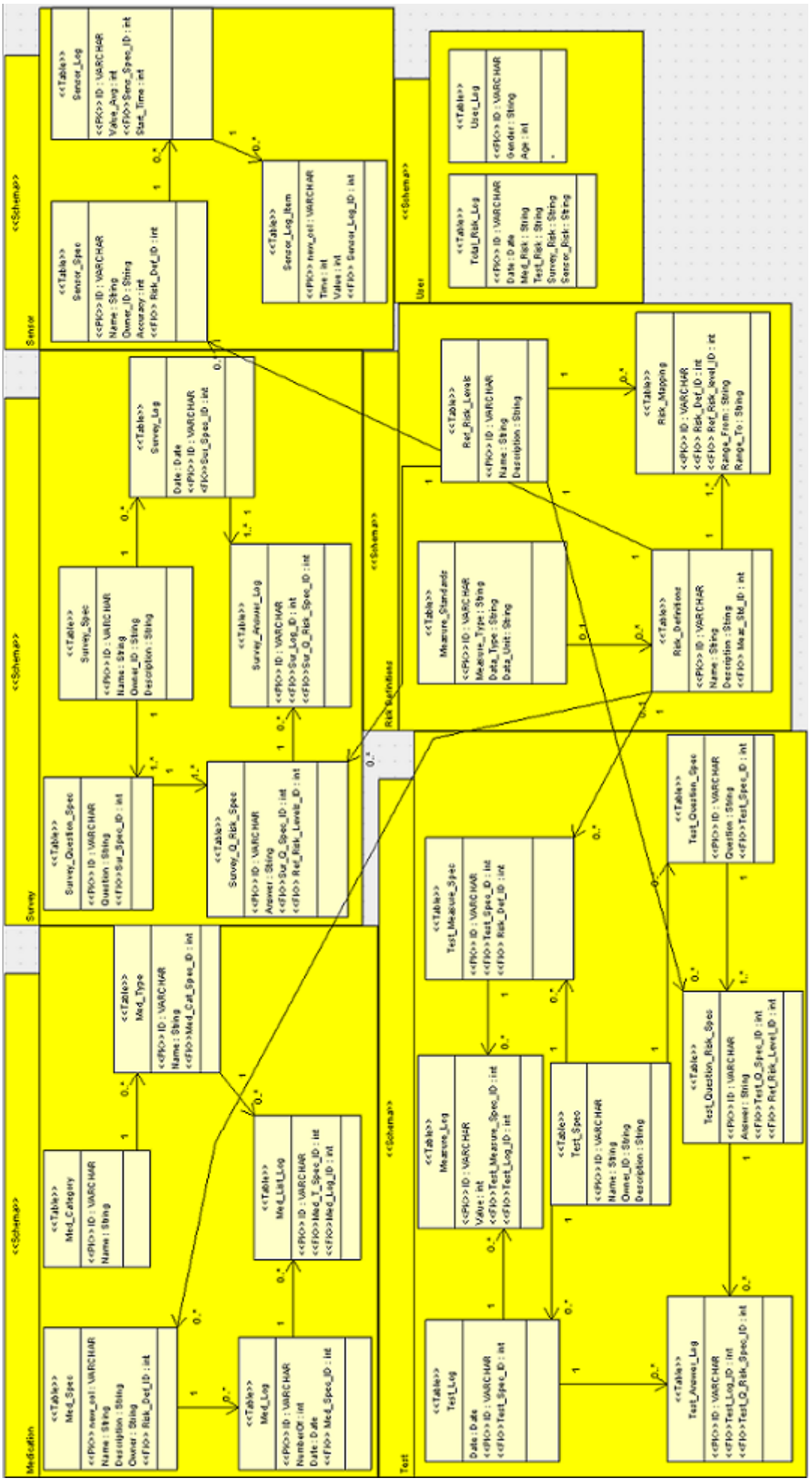
Step one after the delivery of this project could be to place the NoFall platform in a testing environment. A possible opportunity to test and improve the NoFall platform is to make it part of a bachelor or master project. This would be a good way to verify

that this is the way we want to do it, and this is how it should look like. The project could be based on a two-part step; build applications with the platform, and improve the platform as improvements are discovered during usage. Based on the findings, a solution that would be put into production could be made - with the proof-of-concept from this project and the results from the testing as a starting-point.

# Appendix A

# SQLite DB Model

Below is an image of the entire DB model. This image might be a bit hard to read the small details, but it is intended to show how all the different specifications are connected. See chapter 7.4 for the individual specifications and details.


Medication
Sensor
Survey
User
Test
Risk Definition
<<Schema>>
<<Table>>
Med_Spec
Med_Category
Med_Type
Med_Log
Med_Cat_Log
Sensor_Spec
Sensor_Log
Sensor_Log_Item
Survey_Spec
Survey_Question_Spec
Survey_Q_Risk_Spec
Survey_Log
Survey_Answer_Log
User_Log
Total_Risk_Log
Ref_Risk_Levels
Risk_Definitions
Risk_Mapping
Measure_Standards
Test_Spec
Test_Question_Spec
Test_Measure_Spec
Test_Question_Risk_Spec
Test_Log
Measure_Log
Test_Answer_Log

# Appendix B

# NoFall Access

The NoFall platform can be found at:

- https://github.com/UbiCollab/NoFall or
- https://github.com/finnjohansen/NoFall

To get the platform just click Download ZIP (see figure B.1)

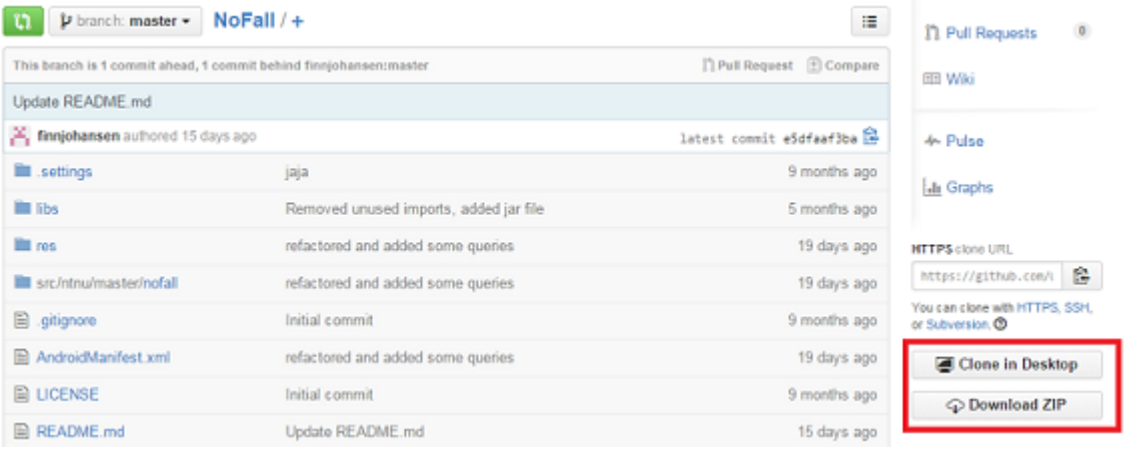



FIGURE B.1: Git Screenshot

The Wiki can be found at:

- https://github.com/UbiCollab/NoFall/wiki or
- https://github.com/finnjohansen/NoFall/wiki

# Bibliography


[1] Browne et. al. Using information technology to assist in redesign of a fall prevention program. *Journal of Nursing Care Quality*, 19(3):218–225, 2004. URL http://journals.lww.com/jncqjournal/Abstract/2004/07000/Using_Information_Technology_to_Assist_in_Redesign.8.aspx.

[2] K. Petersen et. al. Systematic mapping studies in software engineering. URL http://robertfeldt.net/publications/petersen_ease08_sysmap_studies_in_se.pdf.

[3] G.F. Fuller. Falls in the elderly. *Am Fam Physician*, 61(7):2159–2168, April 2000. URL http://www.aafp.org/afp/2000/0401/p2159.html?wvsessionid=wvf503391fbb7c408b927404ac346c989a.

[4] Mary E. Tinetti. Preventing falls in elderly persons. *N Engl J Med*, 348(2):42–49, January 2003. URL http://europepmc.org/abstract/MED/12510042.

[5] Tinetti ME. Clinical practice. preventing falls in elderly persons. *The New England Journal of Medicine*, 348(1):42–49, 2003. URL http://europepmc.org/abstract/MED/12510042.

[6] M. E. Tinetti and C. S. Williams. Falls, injuries due to falls, and the risk of admission to a nursing home. *N Engl J Med*, 337:1279–1284, October 1997. URL http://www.nejm.org/doi/full/10.1056/NEJM199710303371806#t=articleDiscussion.

[7] A. Ungar et. al. Fall prevention in the elderly. *Clinical Cases in mineral and bone metabolism*, 10(2):91–95, October 2013. URL http://europepmc.org/articles/PMC3797008.

[8] Cwikel J et. al. Gait and activity in the elderly: Implications for community falls-prevention and treatment programmes. *Disability and Rehabilitation*, 17(6):277–280, 1995. URL http://informahealthcare.com/doi/abs/10.3109/09638289509166647.

[9] Davies AJ and Kenny RA. Falls presenting to the accident and emergency department: types of presentation and risk factor profile. *Age and Ageing*, 25(5):362–366, 1996. URL http://europepmc.org/abstract/MED/8921140.

[10] Louhivuori K. Hartikainen S, Lönnroos E. Medication as a risk factor for falls: critical systematic review. *The Journals of Gerontology*, 62(10):1172–1182, 2007. URL http://europepmc.org/abstract/MED/17921433.

[11] K. S-W. Kong et. al. Psychosocial consequences of falling: the perspective of older hong kong chinese who had experienced recent falls. *Journal of Advanced Nursing*, 37(3):234–242, February 2002. URL http://onlinelibrary.wiley.com/doi/10.1046/j.1365-2648.2002.02094.x/full.

[12] S. Heinrich et. al. Cost of falls in old age: a systematic review. *Osteoporosis International*, 21(6):891–902, June 2010. URL http://link.springer.com/article/10.1007/s00198-009-1100-1.

[13] J. A. Stevens et. al. The costs of fatal and non-fatal falls among older adults. *Inj Prev*, 12:290–295, 2006. URL http://injuryprevention.bmj.com/content/12/5/290.short.

[14] E. McInnes and L. Askie. Evidence review on older people's views and experiences of falls prevention strategies. *Worldviews on Evidence-Based Nursing*, 1 (1):20–37, March 2004. URL http://onlinelibrary.wiley.com/doi/10.1111/j.1741-6787.2004.04013.x/full.

[15] P. Kannus et. al. Prevention of falls and consequent injuries in elderly people. *The Lancet*, 366(9500):1885–1893, November 2005. URL http://www.sciencedirect.com/science/article/pii/S0140673605676040.

[16] Gillespie LD et. al. Interventions for preventing falls in elderly people (review). *The Cochrane Library*, October 2003. URL http://onlinelibrary.wiley.com/doi/10.1002/14651858.CD000340/pdf/standard.

[17] Lee HC et. al. Effects of a multifactorial fall prevention program on fall incidence and physical function in community-dwelling older adults with risk of falls. *Archives of Physical Medicine and Rehabilitation*, 94(4):606–615, 2013. URL http://europepmc.org/abstract/MED/23220343.

[18] N. M. Sjösten et. al. A multifactorial fall prevention programme in home-dwelling elderly people: A randomized-controlled trial. *Public Health*, 121(4):308–318, April 2007. URL http://www.sciencedirect.com/science/article/pii/S0033350606002952.

[19] Gillespie LD et. al. Interventions for preventing falls in older people living in the community. *The Cochrane Database of Systematic Reviews*, 9, 2012. URL http://europepmc.org/abstract/MED/12510042.

[20] Faber MJ et. al. Effects of exercise programs on falls and mobility in frail and pre-frail older adults: A multicenter randomized controlled tria. *Archives of Physical Medicine and Rehabilitation*, 87(7):885–896, 2006. URL http://europepmc.org/abstract/MED/16813773.

[21] S. Mayor. Nice issues guideline to prevent falls in elderly people. *BMJ*, 329(7477):1258, February 2004. URL http://www.ncbi.nlm.nih.gov/pmc/articles/PMC534478/.

[22] B. L . Braun. Knowledge and perception of fall-related risk factors and fall-reduction techniques among community-dwelling elderly individuals. *Physical Therapy*, 78 (12):1262–1276, December 1998. URL http://www.physicaltherapyjournal.com/content/78/12/1262.short.

[23] K. Hughes et. al. Older persons' perception of risk of falling: Implications for fall-prevention campaigns. *American Journal of Public Health*, 98(2):351–357, February 2008. URL http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2376900/.

[24] A. Holzinger et. al. Mobile computing in medicine: Designing mobile questionnaires for elderly and partially sighted people. *Computers Helping People with Special Needs Lecture Notes in Computer Science*, 4061:732–739, 2006. URL http://link.springer.com/chapter/10.1007/11788713_107.

[25] S.R. Nyman and L. Yardley. Usability and acceptability of a website that provides tailored advice on falls prevention activities for older people. *Health informatics*

*journal*, 15(1):27–39, 2009. URL http://jhi.sagepub.com/content/15/1/27.full.pdf+html.

[26] Leirer et. al. Elders' nonadherence, its assessment, and computer assisted instruction for medication recall training. *Journal of the American Geriatrics Society*, 36(10):877–884, Octoberl 1988. URL http://psycnet.apa.org/psycinfo/1989-23420-001.

[27] R. M. Rippey et. al. Computer-based patient education for older persons with osteoarthritis. *Journal of the American Geriatrics Society*, 30(8):932–935, August 1987. URL http://onlinelibrary.wiley.com/doi/10.1002/art.1780300814/abstract.

[28] A. Scanlan V. Scott, K. Votova and J. Close. Multifactorial and functional mobility assessment tools for fall risk among older adults in community, home support, long term and acute care settings. *Journal of Age and Ageing*, 36(2):130–139, 2007. URL http://ageing.oxfordjournals.org/content/36/2/130.short.

[29] N. Caporusso et. al. A pervasive solution for risk awareness in the context of fall prevention. *Pervasive Computing Technologies for Healthcare*, pages 1–8, 2009. URL http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.ieee-000005191215.

[30] Shumway-Cook et. al. Predicting the probability for falls in community-dwelling older adults using the timed up and go test. *Physical Therapy*, 80(9):896–903, 2000. URL http://physicaltherapyjournal.com/content/80/9/896.short.

[31] A. L. Hendrich et. al. Validation of the hendrich ii fall risk model: A large concurrent case/control study of hospitalized patients. *Applied Nursing Research*, 16(1):9–21, 2003. URL http://www.sciencedirect.com/science/article/pii/S0897189702109025.

[32] Fortino G. Giannantonio R. Gravina R. Guerrieri A. Bellifemine, F. and M. Sgroi. Spine: a domain-specific framework for rapid prototyping of wbsn applications. *Softw: Pract. Exper.*, 41(3):237–265, 2011. URL http://onlinelibrary.wiley.com/doi/10.1002/spe.998/full.

[33] The Apache Software Foundation. Apache licence. Accessed 22.02.2014. URL http://www.apache.org/licenses/.

[34] A. Cooper. The inmates are running the asylum. *SAMS*, 1999.

[35] J. M. Carroll. Making use: scenarios and scenario-based design. *Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques*, page 4, 2003. URL http://dl.acm.org/citation.cfm?id=347652.

[36] James Bucanek. Model-view-controller pattern. *Learn Objective-C for Java Developers*, pages 353–402, 2009. URL http://link.springer.com/chapter/10.1007%2F978-1-4302-2370-2_20?LI=true.

[37] Android. Application fundamentals. Accessed 12.05.2014. URL http://developer.android.com/guide/components/fundamentals.html.

[38] Java homepage. 2014. URL http://java.com/en/download/whatis_java.jsp.

[39] Sqlite homepage. 2014. URL http://www.sqlite.org/about.html.

[40] Eclipse homepage. Accessed 2014. URL https://www.eclipse.org.

[41] Sqlitebrowser homepage. Accessed 05.02.2014. URL http://sourceforge.net/projects/sqlitebrowser/.

[42] Texstudio homepage. Accessed 25.01.2014. URL http://texstudio.sourceforge.net/.

[43] Pedometer github. Accessed 04.06.2014. URL https://github.com/bagilevi/android-pedometer.