Master‘s Thesis

Building a smart System to assist Medical Doctors in conducting Anamnesis

Thomas Kaul
of Zug, Switzerland (05-918-602)

supervised by
Prof. Dr. Harald C. Gall
Prof. Dr. med. Edouard Battegay

University of Zurich[UZH]

s.e.a.l.
software evolution & architecture lab

Master‘s Thesis

Building a smart System to assist Medical Doctors in conducting Anamnesis

Thomas Kaul

University of Zurich[UZH]

s.e.a.l.
software evolution & architecture lab

Master's Thesis

Author: Thomas Kaul, thomas.kaul@uzh.ch

Project period: July 10, 2012 - January 9, 2013

Software Evolution & Architecture Lab
Department of Informatics, University of Zurich

Acknowledgements

First of all, I am grateful to Prof. Dr. Harald Gall for supervising this ambitious and challenging topic. Secondly, I would like to thank my medical supervisors Prof. Dr. med. Edouard Battegay and Dr. med. Lukas Zimmerli. They provided very helpful expertise during this work.

A special thank goes to my brother Dr. med. Reto Kaul for coming up with the idea in this work as well as carefully handcrafting a medical ontology and Michael Würsch for his interesting feedback. Many thanks to all the test participants from the *UniversitätsSpital Zürich* for taking part in this study. Finally, I would like to thank the following people who supported my thesis during the last months: Anna & Hansueli Müller and my family.

Abstract

In the past years, medical doctors have increasingly been confronted with paperwork. In a medicine becoming more complex and a growing willingness on side of the patient to sue doctors, a precise documentation of the patient's complaint is inevitable. Recent studies have shown that a Swiss junior doctor spends up to two and a half hours a day with the documentation of patient data [G+11]. This time is missing in the direct contact with the patient.

In this thesis, we aimed to implement a system optimized for mobile touch devices with an effective and easily usable user interface to support doctors in their documentation work during the patient consultation. To reduce the typing effort, the system is based on a recommender system which suggests symptoms related to already recorded symptoms. In an experiment with a doctor-patient setting, the system achieves promising scores in matters of usefulness and usability.

Zusammenfassung

Während der letzten Jahre wurden Ärzte zunehmend mit Büroarbeiten konfrontiert. Die Medizin wird immer komplexer und gleichzeitig nimmt die Bereitschaft von Seiten der Patienten zu, Ärzte zu verklagen. Deshalb ist eine präzise Dokumentation der Beschwerden des Patienten unumgänglich geworden. Eine kürzlich publizierte Studie zeigt, dass ein Assistenzarzt in der Schweiz durchschnittlich bis zu zweieinhalb Stunden pro Tag mit der unattraktiven Patientendokumentation beschäftigt ist [G+11]. Diese Zeit fehlt ihm im direkten Kontakt mit dem Patienten.

Diese These befasst sich mit der Entwicklung eines für mobile Endgeräte mit Touchscreen optimierten Softwaresystems, welches Ärzte während der Patientenkonsultation in ihrer Dokumentationsarbeit unterstützt. Dabei setzen wir den Schwerpunkt auf eine effektive und einfache Bedienung. Um den Aufwand beim Eintippen zu reduzieren, verwendet das System ein Empfehlungssystem, welches verwandte Symptome der bereits eingegebenen Symptome vorschlägt. In einem Experiment mit Ärzten und Schauspielpatienten hat unser System erfolgsversprechende Ergebnisse in den Bereichen Nützlichkeit und Benutzerfreundlichkeit erzielt.

Contents

List of Figures

List of Tables

List of Listings

Chapter 1

Introduction

The process of patient documentation in hospitals and doctor's offices is reminiscent of the time when computers were not readily available. Medical doctors have to take notes, either type them into the computer system or record the findings with the aid of a dictaphone, and finally review the documentation.

In a medicine becoming increasingly complex and a growing willingness to sue doctors, a precise documentation of the complaint image is required these days. Thus, doctors need to make a major administrative effort of repetitive and tiring work. Recent studies have shown that Swiss junior doctors spend 24.7% of their workload with documenting patient data [G$^+$11]. In a 50 hours work week this makes up to two and a half hours a day. If this proportion is still rising slightly, junior doctors spend almost as much time with the documentation as with their patients (30.8%) [G$^+$11].

A recent study of the *University of Chicago Medicine* shows that medical residents equipped with a personal mobile computer increase their efficiency, reduce delays in patient care and enhances continuity of care. The majority of residents have reported that the portable computer allows them to complete tasks more quickly and enables them to spend more time on direct patient care. [PCL$^+$12]

The goal of this thesis is to explore the potential of mobile touch devices as an assistance for medical doctors in their patient documentation work. Therefore this thesis examines how mobile multi-touch interfaces can be used beneficially during a patient's consultation. We developed a mobile application focusing on an effective and usable user interface that demonstrates how mobile devices can be used live at the patient's bedside.

The most important part of the documentation is that a third person is able to understand how the doctor has made the diagnosis. In this context, the diagnosis is the name of a certain disease which can be described as a combination of symptoms, signs, and results of laboratory and imaging investigations. So that a third person comprehends the diagnosis, particularly these problems needs to be accented which distinguish similar diseases. The goal of this work is to discover related symptoms and point them out. With the help of already developed ontologies, we build a set of adequate symptoms for each disease. Having these generated disease profiles, we are able to calculate the similarity of two symptoms. Our implemented recommender system in the *Mobile Application* then has the capability to suggest related symptoms which should be asked during the consultation. By suggesting related symptoms, the typing effort which arises from searching manually for specific symptoms can be minimized.

However, to make the system applicable for a medical doctor, it still needs more functionality than entering symptoms. In addition, the recording of patient history, medications, allergies and family history needs to be realized in a flexible way since the conversation with the patient may hop between several subjects and complaints.

The realization of the problems mentioned above is a major part of this thesis and is explained

in Chapter 3 and 4. To be able to understand the problems with all the facets we provide a theoretical background of *Medical Records*, *Data Representations* and *User Interface Design* in Chapter 2. The empirical evaluation in Chapter 5 shows that the implemented prototype can meet these requirements.

Chapter 2

Background

This chapter starts with a short overview of support systems for medical doctors through information technology followed by a description of how a typical *Medical Record* (Section 2.1) is structured. Section *Data Representations* (2.2) covers different variants of data representation. Principles of *User Interface Design* are treated in Section 2.3.

The first clinical *decision support systems* (or *expert systems*) were created in the 1970s [Tay06]. These systems were designed to assist doctors in their decisions by reasoning about knowledge. Nowadays, decision support tools are at least on a limited scale capable of annotating clinical, laboratory, or electrocardiogram reports with an interpretation. Sometimes, *decision support systems* are capable to give advice on what to do next in examination or treatment. This may help doctors with their decisions and actions to improve patient outcomes. [SW09] These systems are often based on aggregated data of thousands of medical records which are discussed in the next section.

2.1 Medical Records

The terms *medical record*, *health record*, or *patient record* are commonly used to specify a systematic documentation of medical information on a particular patient. The medical record of a patient has traditionally been a paper record of data which has been accumulated over a single or multiple consultations. An example of a typical medical record can be found in the Appendix (Listing 7.1 in Section 7.6). The paper-based medical record is typically bundled and stored in a centralized archive. It usually contains details of hospital admissions, hand-written file notes from medical doctors, hospital nurse, and health staff, results of laboratory, pathology, and imaging investigations, reports of operations, and copies of correspondence to local medical officers. [Fen07]

Nevertheless, in the current complex medical environment the patient record is often incomplete because departments in the hospital run their own databases. Thus, the paper-based medical record is an inconvenient solution.

The progress in information technology coupled with the increasing volume of data needed to be stored in a medical record have recreated the interest in researching an electronic solution to manage these critical data. An effective management, processing, and exchange of patient data in medical records improves the quality of healthcare. *Electronic Medical Records* (EMR) have the following benefits: easy access, secure storage, and safe transmission of patient data. However, designing an EMR is a complex task. Important challenges are the complexity and quantity of the data, diversity in the information infrastructure and databases within and across hospitals. [Fen07]

15% of the general practitioners in Switzerland maintain an EMR [Kra12]. The latest progression shows that EMR will be entirely integrated into health care in future.

Medical records are usually recorded manually with little support from computer systems. On the market there are currently systems, which provide at least static text blocks. This allows to write medical reports in less time. For example *Practice Fusion*[1], a web-based EMR software. Symptoms and predefined clauses can be selected in a tree-like structure to build a report by clicks. The problem is that the user is restricted to a set of static clauses. Searching for a symptom is also hardly faster than formulating a sentence.

Our approach described in this thesis has the potential to dynamically generate medical reports by a computer system.

2.2 Data Representations

In this work, we make use of semantic data to model medical information. This section gives an overview of different types of data representations. The most common ones include *Tabular Data* (2.2.1), *Relational Data* (2.2.2), and *Semantic Data* (2.2.3). These data representations are characterized in the next sections.

2.2.1 Tabular Data

One of the simplest kind is *tabular data* as shown in Table 2.1. The data is kept in a table with rows and columns. Such a placement gives each piece a particular meaning. The advantage of tabular data is the ease of reading and manipulating. The data stored in a table has also limitations. In the last column, all open days are strung together. For a computer it is not simple to find all restaurants open on Monday and it cannot understand that the fields in the *Open* column are used to store multiple distinct information values. The problems get more complicated when there are several tables with references pointing to each other.

Restaurant	Cuisine	Open
Mama Africa	South African	Mon, Tue, Wed, Thu, Fri, Sat, Sun
Nooch	Asian	Mon, Tue, Wed, Thu, Fri
Outback Lodge	Australian	Mon, Tue, Wed, Thu, Fri, Sat, Sun
Pizzeria Pippone	Italian	Tue, Wed, Thu, Fri, Sat, Sun

Table 2.1: Table of restaurants

2.2.2 Relational Data

To overcome the problems of tabular data, *relational data* has been introduced. Many years of research have made relational databases become very fast and powerful tools for storing large sets of data. The data model is now well understood and widely used in industry. A "relational database allows multiple tables to be joined in a standardized way" [SET09]. In the example with restaurant data, we define four tables: *Restaurant* (2.2) includes the name and cuisine type, *Cuisine* (2.3) contains the name of the cuisine type, *Day* (2.4) covers all week days, and *Open* (2.5) defines the connection of the restaurant and the week day. Now it is possible to do more

[1] `http://www.practicefusion.com`

sophisticated queries like finding all restaurants that are open on Monday. The meanings of the values are described by the schema. Even though the database does not understand the meaning of "restaurant", it can respond to requests to display restaurants with given properties.

ID	Name	CuisineID
0	Mama Africa	0
1	Nooch	1
2	Outback Lodge	2
3	Pizzeria Pippone	3

Table 2.2: Table 'Restaurant'

ID	Name
0	South African
1	Asian
2	Australian
3	Italian

Table 2.3: Table 'Cuisine'

ID	Name
0	Monday
1	Tuesday
2	Wednesday
3	Thursday
4	Friday
5	Saturday
6	Sunday

Table 2.4: Table 'Day'

In general, relational databases fit for datasets where the data model is clear from the beginning and there is some understanding of how the data will be used in future. Lots of applications (e.g. product catalogues or address directories) fit well into relational schemas because there is generally a fixed set of fields needed.

But relational databases have their limits due to the fact that there is no consistent way to get the meaning of a relation. As long as the data types are matching, any columns can be joined without checking the semantics. Humans are usually able to guess the meaning of a relation, while computers are not capable of doing so. Therefore it is necessary to encode a significant amount of implicit knowledge into applications to make use of the data. [Wür12]

In a context such as the Internet with rapidly changing contents, one can hardly know what will be available and how people want to use it. Relational data reaches its limits because of a centralized architecture where data cannot be easily distributed.

Going back to the example: if new information about bars needs to be added (independent of locations and bars in already captured restaurants), existing data of the old data model needs to be transformed to the new data model. This process is called *schema migration* and is usually

ID	RestaurantID	DayID
0	0	0
1	0	1
2	0	2
3	0	3
4	0	4
5	0	5
6	0	6
...	...	...
24	3	6

Table 2.5: Table 'Open'

very cumbersome. Not only the data must be migrated, but all queries and dependent code need to be adapted. Another problem which arises from relational databases is that schemas can get drastically complex when dealing with many different kinds of data. [SET09]

2.2.3 Semantic Data

To overcome the limitations of explicit semantics, there is a different approach referred to as *Semantic Web* which is a collaborative movement advanced by the *World Wide Web Consortium* (W3C). Data is enriched with metadata describing the semantics to make them computer-processable.

To describe information with metadata, "an *ontology* has to be defined that formally describes the concepts (classes) found in the domain of discourse, the relations between these concepts and the properties used to describe them" [WRDG10]. The ontology has an important role in knowledge management and knowledge discovery, especially in text mining applications. Since the information is described in formal semantics, data can be exchanged between applications as long as they are supporting the same ontology [WRDG10]. In the biomedical field many ontologies have been developed for controlled vocabularies and categorization of concepts. Besides that, there are also different terminologies for genes and proteins. Most of this data is open data which is freely available to everyone's usage and can be downloaded from the Web. [CFFH05]

The *Resource Description Framework* (RDF) is a standard to express graphs of data and share them with other people or machines. The goal of *Semantic Web* is to transform the Internet currently dominated by unstructured and semi-structured documents into a "web of data". [SET09]

This approach is based on making statements about resources in the form of subject-predicate-object expressions. It is similar to classic conceptual modeling approaches such as *class diagrams* or *entity-relationship*. These expressions are known as triples. The subject in a triple corresponds to an entity for which a conceptual class exists. The predicate is the property of the entity to which it is attached. Objects can be divided into two categories: literal values such as strings or numbers, and entities which can be the subject in other triples. Selected triples related to the restaurant example are provided in Table 2.6.

"Multiple triples can be tied together by using the same subjects and objects in different triples" [SET09]. Once these triples have been assembled to chains of relationships they form a directed graph. Many reusable techniques have been developed to query, explore, and use large graphs.

Subject	Predicate	Object
Mama Africa	cooks	South African
Nooch	cooks	Asian
Outback Lodge	is operating	a bar
Pizzeria Pippone	is open on	Tuesday

Table 2.6: Subject-predicate-object relationship

2.3 User Interface Design

User interface design has an important role in this work in the context of usability. It is not primarily about buttons and menus. *User interface design* is rather about the user interaction with the system. It is not how a product looks, but about how it works. This is known as *Usability* [BD09]. Users only see and interact with the *Graphical User Interface* (GUI). What goes on in the underlying back-end architecture, they do not notice. Therefore getting the user interface right has a great impact on how much a user enjoys working with an application.

Before building a user interface it is important to first understand what makes a usable user interface. Good user interfaces share eight characteristics or qualities.

1. Clarity
The interface makes the usage clear by flow, hierarchy, language, and metaphors for visual elements. Ambiguous information should be avoided in order that all users interpret the display in the same way [Joh10]. A clear interface does not require a manual and prevents users from making mistakes.

2. Concision
The interface should not be made clear by over-clarifying and labelling all elements. If the interface is cramped, users are not able to find what they need. "The real challenge in making a great interface is to make it concise and clear at the same time" [Fad09].

3. Familiarity
Even if a user interacts with an interface for the first time, certain design elements can still be familiar. Often real-life metaphors are used like folder-style tabs for navigation. People then recognize the items and know how they work.

4. Responsiveness
An interface should respond quickly on interaction and not feel laggy. Additionally, the user needs to be informed about the current status of the application.

5. Consistency
Keeping the interface consistent over the entire system allows the user to recognize usage patterns. Acquired knowledge can be applied to new use cases. For example people may expect to find similar functionality in similar places. [Tid05]

6. Aesthetics
An attractive interface does not make the application do its job better, but the user is happier using the software. "Interfaces actually become more usable when people enjoy using them" [Tid05].

7. Efficiency
"Time is money, and a great interface should make the user more productive [...]" [Fad09]. One of the core benefits of technology is that it allows user to perform tasks with less effort and time.

8. Forgiveness
An elaborated interface should not punish the user for making mistakes. On the contrary,

an application could offer the possibility to undo an action.

The principles mentioned above often affect each other. This makes it difficult to create an interface that takes all eight principles into account. [Fad09]

Chapter 3

Design

This chapter deals with the fundamental design and architecture of the *Crawler* (Section 3.1) to generate data describing the similarity between symptoms. This data is later used in the *Mobile Application* (Section 3.2) to recommend related symptoms.

3.1 Crawler

The goal of the *Crawler* is computer-aided preparation of accurate disease profiles with associated symptoms. We use text mining as a technique to create these profiles. "Text mining aims to extract useful knowledge from textual data or documents" [Hea99] [CFFH05]. Text mining is often considered a subfield of data mining. However, some text mining techniques have originated from disciplines like *information retrieval*, *information visualization*, *computational linguistics*, and *information science*. Text mining applications include examples such as document classification, document clustering, information extraction, and summarization. [CFFH05]

Two examples of (manually compiled) disease profiles are shown below.

Achalasia of Cardia (ICD-10 K22.0)
Achalasia of Cardia is a "failure of relaxation of the *lower oesophageal sphincter* (due to degeneration of the *myenteric plexus*)" [LWTC07]. The *lower oesophageal sphincter* is a ring of muscles that separates the stomach from the esophagus.

- chest pain
- cough
- dysphagia
- odynophagia
- pyrosis
- regurgitation
- vomiting
- weight loss

Pneumonia (ICD-10 J18.9)
"*Pneumonia* is an infection of the *pulmonary parenchyma*" [FBK+08] (infection of the lung).

- chest pain
- chills
- confusion
- cough
- dyspnea
- fatigue
- fever
- loss of appetite
- sputum
- tachycardia
- tachypnea

These manually compiled disease profiles act as gold standards (reference symptoms) to compare and measure the performance of machine generated profiles. The gold standards have been

compiled by Dr. med. Lukas Zimmerli, medical doctor at the *UniversitätsSpital Zürich* in Switzerland.

To limit the complexity in this work, the *Crawler* is restricted to the most important diseases presenting with either *abdominal* or *chest pain*. Technically, there is no such restriction in the *Crawler*. The *Crawler* is able to evaluate any randomly chosen disease.

The following list shows common diseases presenting with *abdominal pain* considered by the *Crawler* according to [Pro10].

- Abdominal aortic aneurysm (dissecting)
- Abdominal cancer
- Abdominal trauma
- Adrenal crisis
- Anthrax, GI
- Appendicitis
- Cholocystitis
- Cholelithiasis
- Cirrhosis
- Crohn's disease
- Cystitis
- Diabetic ketoacidosis
- Diverticulitis
- Duodenal ulcer
- Ectopic pregnancy
- Endometriosis
- Escherichia coli
- Gastric ulcer
- Gastritis
- Gastroenteritis
- Heart failure
- Hepatic abscess
- Hepatic amebiasis
- Hepatitis
- Herpes zoster
- Insect toxins
- Intestinal obstruction
- Irritable bowel syndrome
- Listeriosis
- Mesenteric artery ischemia
- Myocardial infarction
- Norovirus infection
- Ovarian cyst
- Pancreatitis
- Pelvic inflammatory disease
- Perforated ulcer
- Peritonitis
- Pleurisy
- Pneumonia
- Pneumothorax
- Prostatitis
- Pyelonephritis (acute)
- Renal calculi
- Sickle cell crisis
- Smallpox (variola major)
- Splenic infarction
- Systemic lupus erythematosus
- Ulcerative colitis
- Uremia

Common diseases presenting with *chest pain* considered by the *Crawler* as mentioned in [Pro10].

- Angina pectoris
- Anthrax (inhalation)
- Anxiety
- Aortic aneurysm (dissecting)
- Asthma
- Blast lung injury
- Blastomycosis
- Bronchitis
- Cardiomyopathy
- Cholecystitis
- Coccidioidomycosis
- Costochondritis
- Distention of colon's splenic flexure
- Esophageal spasm
- Herpes zoster (shingles)
- Hiatal hernia
- Interstitial lung disease
- Legionnaires' disease
- Lung abscess
- Lung cancer
- Mediastinitis
- Mitral valve prolapse
- Muscle strain
- Myocardial infarction
- Nocardiosis
- Pancreatitis

- Peptic ulcer
- Pericarditis
- Plague
- Pleurisy
- Pneumonia
- Pneumothorax
- Psittacosis
- Pulmonary actinomycosis
- Pulmonary embolism
- Pulmonary hypertension (primary)
- Q fever
- Rib fracture
- Sickle cell crisis
- Thoracic outlet syndrome
- Tuberculosis
- Tularemia

3.1.1 Requirements of Medical Ontologies

This section lists the requirements of medical ontologies the *Crawler* utilizes grouped by the *Symptom Ontology* and the *Disease Ontology*. Both ontologies need to be structured in a machine interpretable language like *Web Ontology Language* (OWL).

Symptom Ontology

We need an extensive ontology with well structured symptoms. The symptoms must be selected properly so that it make sense that they can be asked during the medical history taking. To accomplish the crawling task, we also require related synonyms of the symptoms. An example in hierarchical order is shown below with synonyms in squared brackets.

Symptom
→ Nervous system symptom
→ sensation perception
→ pain
→ abdominal pain [abdo pain, abdominal and pelvic pain]
→ chest pain
→ headache [cephalalgia, cephalgia, cephalodynia, head ache, head pain]

Disease Ontology

We are in need of an entire ontology with diseases in a hierarchical structure. Again we require synonyms to locate a particular disease over various sources. Furthermore, the diseases should be annotated with the corresponding *International Statistical Classification of Diseases and Related Health Problems Version 10* (ICD-10) code. This code consisting of a letter, a number, a dot, and another number, orders diseases in a standardized medical classification developed by the *World Health Organization* (WHO).

Another example with synonyms in squared brackets is as follows.

Disease
→ Diseases of the respiratory system (J00-J99)
→ Influenza and pneumonia (J10-J18)
→ Pneumonia, organism unspecified (J18)
→ Bronchopneumonia, unspecified (J18.0) [bronchopneumonia]
→ Lobar pneumonia, unspecified (J18.1) [lobar pneumonia]
→ Hypostatic pneumonia, unspecified (J18.2) [hypostatic pneumonia]
→ Pneumonia, unspecified (J18.9) [pneumonia, pneumonitis]

3.1.2 Overview of Medical Ontologies

This section highlights various medical ontologies freely available on the Internet.

Gemina Symptom Ontology

The *Gemina Symptom Ontology*[1] by the *University of Maryland* consisting of 936 classes was designed around the guiding concept of a symptom being: "A perceived change in function, sensation or appearance reported by a patient indicative of a disease" [sym12]. The ontology includes various cross references, definitions and synonyms.

Disease Ontology

The *Disease Ontology*[2] authored by the *Northwestern University* is a community driven, open source ontology that is designed to link datasets through disease concepts. The 8656 classes include various cross references, definitions and synonyms. However, the ontology does not contain *ICD-10* codes.

ICD-10 Ontology

The *Data & Knowledge Management (DKM) Unit at Fondazione Bruno Kessler (FBK)* has authored the *ICD-10 Ontology*[3], a formalization of the ICD 10th edition, published by the WHO in 2004. The ontology includes 14502 classes with *ICD-10* codes, but no other data.

Diseases Database

The *Diseases Database*[4] is a database that underlies a website and provides information about the relationships between medical conditions, symptoms, and medications. This is not an ontology in the classical sense because there is no semantic structure. Since the quality of the relations between diseases and symptoms and synonyms is high, the website is still valuable. As a drawback, there are no ICD-10 codes for diseases.

3.1.3 Conclusion

The following section describes the process of building the *Symptom Ontology* and *Disease Ontology* in this work.

Symptom Ontology

As a basis we take the *Gemina Symptom Ontology* from the *OBO Foundry* as illustrated in Figure 3.1. This ontology includes a lot of definitions, synonyms and cross references. A disadvantage is that it contains a set of entries declared as symptoms which are diagnoses. Among others, these are *arthritis*, *bronchitis*, *cellulitis*, *hepatitis*, *meningitis*, *myocarditis*, *stroke* and so on. In our ontology we only need symptoms the doctor may ask for conducting the anamnesis. Therefore, manual editing is needed to a large extent to refine the ontology. We have used various sources of information like the *Diseases Database* [dis12], *dict.md* [dic12] and *DiagnosisPro* [dia12] for enrichment to improve the quality especially of the synonyms.

[1] `http://symptomontologywiki.igs.umaryland.edu/wiki`
[2] `http://diseaseontology.sourceforge.net`
[3] `https://dkm.fbk.eu/index.php/ICD-10_Ontology`
[4] `http://www.diseasesdatabase.com`

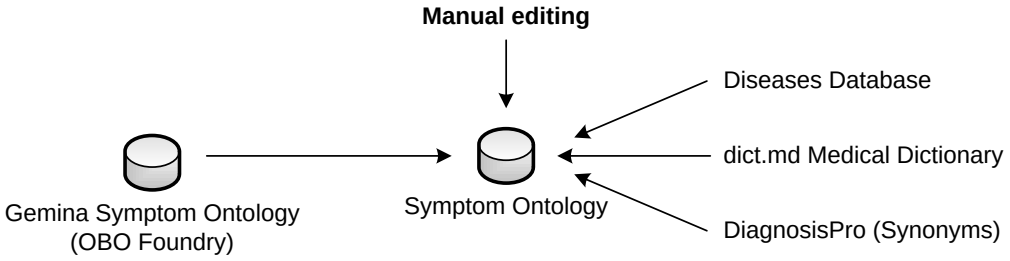

Figure 3.1: Symptom Ontology Process

Disease Ontology

For our disease ontology we have used the *ICD-10 Ontology* from the *Data & Knowledge Management (DKM) Unit at Fondazione Bruno Kessler (FBK)* as a basis (cf. Figure 3.2). This was necessary because the *Disease Ontology* of the *OBO Foundry* does not contain *ICD-10* codes. Since the *ICD-10 Ontology* does not contain any additional information, we tried with the help of *Wikipedia* [wik12b] to find a connection to the *Disease Ontology* (*OBO Foundry*). In a further step synonyms found in the *Diseases Database* have been added to the diseases.

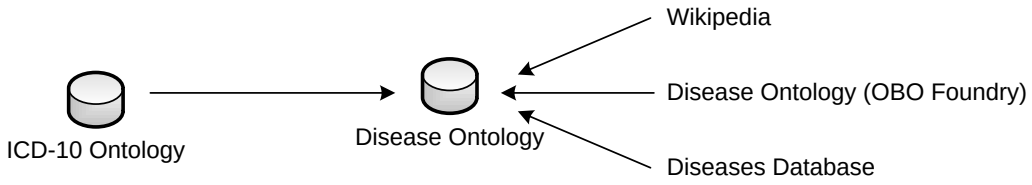

Figure 3.2: Disease Ontology Process

3.1.4 Architecture

Now having the ontologies ready, this section presents the architecture of the *Crawler* consisting of five main components.

CrawlerController	Controls the data collection and manipulation process
SourceManager	Collects and stores raw data from various sources
PhraseMatcher	Analyzes and matches raw data with phrases (symptoms found)
Evaluator	Evaluates and ranks the computed result
Exporter	Stores disease profiles in different formats

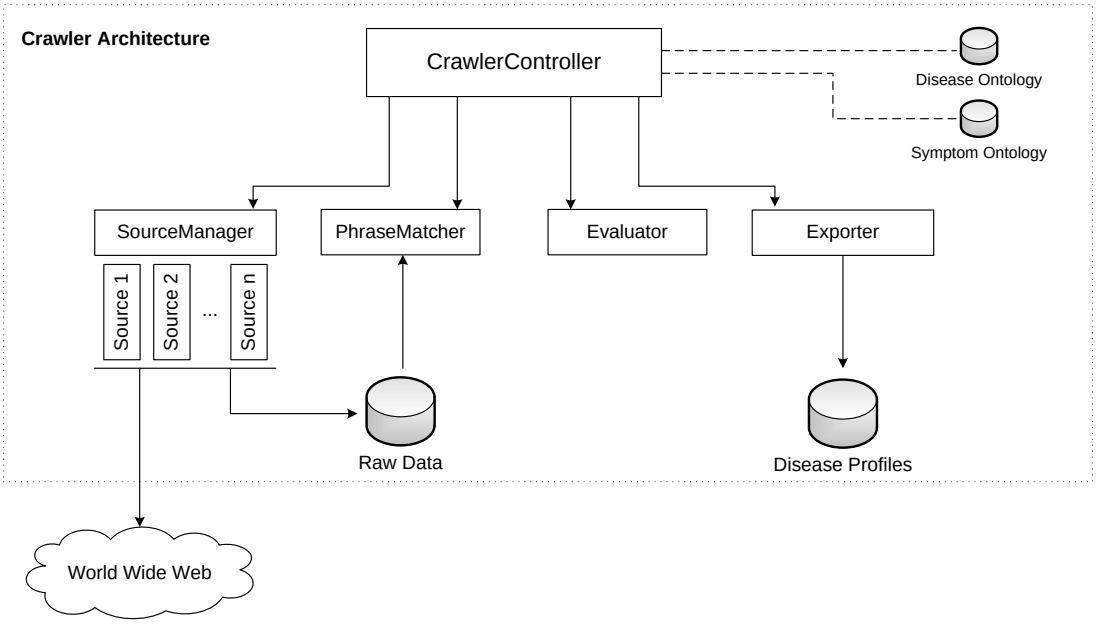

Figure 3.3: Crawler Architecture

CrawlerController

First the *CrawlerController* loads the *Disease Ontology* as well as the *Symptom Ontology* in *OWL* format. Then it creates the list of diseases which needs to be crawled. This happens by just searching individuals in the disease ontology annotated with a defined property.

SourceManager

The *SourceManager* retrieves the website for each disease from various sources (cf. Section 4.1). To locate a website (e.g. *Pneumonia* at *Healthline* [hea12]), a request to *Google* search is sent to retrieve the URI of the most probable website. In this example, *Google* returns `http://www.healthline.com/health/pneumonia`, which is an optimal result. The result is not always perfect, but by the mass of different sources, errors can be averaged out. Once the URI is known, the site can be downloaded via a *Hypertext Transfer Protocol* (HTTP) request and cached in the *Raw Data* store. Thus, the retrieving process is completed.

PhraseMatcher

It continues with the *PhraseMatcher* which analyzes and matches the raw data. For this task, the *PhraseMatcher* receives a list with all symptoms and appropriate synonyms built up from the *Symptom Ontology* (via *CrawlerController*). By analyzing each document from the *Raw Data* store and matching with the symptom list, a disease profile can be generated with all symptoms and a boolean value if it was found or not found in the document. The matcher also considers symptoms in plural (e.g. "*palpitation*" vs. "*palpitations*") by adding an 's' character.

Evaluator

The *Evaluator* aggregates and ranks the computed results from the various sources. Uncertain symptoms below a certain threshold are filtered out. Furthermore, different ratios are calculated against a gold standard if available. These gold standards have been compiled manually for certain diseases to proof the quality of the machine generated profiles.

Exporter

Finally, the *Exporter* assembles all results and generates various documents. These include a human-readable document with additional information (Figure 3.4) in a comma-separated values (CSV) format, as well as a machine readable format in *JavaScript Object Notation* (JSON) to be later used to build up the similarity matrix described in Section 4.2.4.

Pneumonia (unspecified)

ICD-10 J18.9

Rank	Symptom	Confidence	# Sources	True Positive
1.	Cough	91.53%	15	x
	Dyspnea	91.53%	15	x
3.	Fever	83.05%	14	x
4.	Fatigue	71.19%	12	x
5.	Chest pain	64.41%	11	x
6.	Sputum	62.72%	11	x
7.	Chills	55.94%	11	x
8.	Headache	47.45%	8	
9.	Hypertension	42.38%	8	
10.	Painful respiration	42.37%	7	
11.	Tachypnea	40.69%	7	x
12.	Vomiting	38.99%	7	x
	Loss of appetite	38.99%	7	x
14.	Weight loss	37.30%	7	
	Abdominal pain	37.30%	4	
16.	Confusion	32.21%	6	
17.	Cyanosis	32.20%	6	
18.	Nausea	28.83%	5	
	Hypotension	28.83%	5	
	Hyperhydrosis	28.83%	6	
21.	Diarrhea	27.11%	5	
22.	Edema	25.44%	5	
	Wheezing	25.44%	6	
	Seizure	25.44%	6	

Sensitivity (Recall)	90.91%	(10 of 11)
Specificity	94.98%	(265 of 279)
Precision	41.67%	(10 of 24)
Accuracy	94.83%	
Sources with hits	16	
False Negative	Tachycardia	

Figure 3.4: Generated disease profile for Pneumonia

The confidence value is calculated as the ratio between number of sources that contains the

specific symptom and the number of all sources. Symptoms with a confidence value less than a certain threshold are filtered out. Since not all sources are weighted in the same way, it may happen that a symptom has a higher confidence value and therefore a higher rank with fewer sources. The column *True Positive* indicates whether a symptom belongs to the manually compiled gold standard or not. The ratios *Sensitivity* (*Recall*), *Specifity*, *Precision*, and *Accuracy* are explained in Section 4.1. The symptom *Tachycardia* is false negative because related to the gold standard the *Crawler* has missed *Tachycardia*. The gold standard for *Pneumonia* has been manually compiled.

An example of the final JSON document can be seen in Listing 3.1 in descending order for the disease *Pneumonia*. The *id* stands for the symptom identifier (according to the *Symptom Ontology*. The *weight* (in percentage) expresses in how many sources the symptom occurs.

```
[
   {"id":1024,"weight":91.52686},
   {"id":1094,"weight":91.52686},
   {"id":1259,"weight":83.05372},
   {"id":1229,"weight":71.19132},
   {"id":1165,"weight":64.41281},
   {"id":1212,"weight":62.71818},
   {"id":1106,"weight":55.93967},
   {"id":1180,"weight":47.449585},
   {"id":1694,"weight":42.38265},
   {"id":1114,"weight":42.3657},
   {"id":1099,"weight":40.68802},
   {"id":1030,"weight":38.99339},
   {"id":1273,"weight":38.99339},
   {"id":1100,"weight":37.298763},
   {"id":1102,"weight":37.298763},
   {"id":1105,"weight":32.214878},
   {"id":1191,"weight":32.197933},
   {"id":1192,"weight":28.825623},
   {"id":1036,"weight":28.825623},
   {"id":1053,"weight":28.825623},
   {"id":1198,"weight":27.114048},
   {"id":1161,"weight":25.436367},
   {"id":1086,"weight":25.436367},
   {"id":1115,"weight":25.436367}
]
```

Listing 3.1: Generated disease profile of Pneumonia in JSON

3.2 Mobile Application

To get the right information at the right time and the right place in an easy and comprehensive manner is a difficult task for eHealth[5] application designers. The system should actively involve the users, while giving them full control over the information in their ways of understanding. Moreover, an easy navigation and orientation within the information is crucial, having

[5]Healthcare supported by electronic processes and communication

in mind the limitation of the accessing device (tablet device, laptop, netbook, etc.). The connectivity medium (e.g. wireless network) is also an important aspect in order to support different interaction methods and disruption of service. [GGS+10]

3.2.1 User Stories

The process of the design was as follows: We performed usability tests with medical doctors on a monthly basis each of them with the latest prototype. One user at a time is shown a prototype of the system, or some sketches of individual screens and asked to either figure out, what it is, or try to complete a typical task [BD09]. During the task completion, the participant is encouraged to think out loud as much as possible. This is known as the think-aloud protocol [Lew82]. Whenever the observer is not sure what they are thinking, the questions "What are you looking at?" or "What are you thinking?" should be asked. But without giving hints about what to do. [Kru05]

The test sessions were recorded with a screen recording software which logs all screen and audio activity. Afterwards the session was analyzed. For every session, the three most serious usability problems were determined and fixed for the upcoming session as suggested in [Kru09]. Also the new and most needed features for the next version were discussed and selected. We followed an agile software development method, which is based on an incremental and iterative development. The characteristic of it is that requirements and solutions evolve over time. Agile software development encourages rapid response to change in a flexible way. [B+12]

During the development time, many requirements have been collected and written down as *user stories*. "A user story is one or more sentences in the everyday or business language of the end user or user of a system that captures what a user does or needs to do as part of his or her job function" [wik12a]. User stories are suitable because stories are generally the natural way people process information. [Wei11]

After the user stories were formulated, mockups for the concept were created (partially including interaction steps). The mockups created in *Balsamiq Mockups*[6] look like sketches, so the stakeholders do not get distracted by little details and provide honest feedback.

Only after the designs were verified, the actual implementation was started. As the implementation was completed, there was again a review of the implemented user story with possible refinements.

The major requirements formulated as user stories are presented below.

Starting Application

User story The application begins by bringing up the dashboard.

This is the home page of the application where all interactions start. The page needs to be easy to find from everywhere in the application. [FS04]

Administrative Task 1

User story As a user, I want to create a new patient's records.

A new patient's record is typically created by entering personal data of the patient like name (or identifier), date of birth, and gender.

[6] `http://www.balsamiq.com`

Administrative Task 2

User story As a user, I want to load an already created patient's records.

A patient's record in the system needs to be retrievable, displayable, and editable.

History of the present illness (entry point)

User story As a user, I want to enter the first symptom.

We have investigated to design a screen to select the first symptom (cardinal symptom). The view consists of the most important body systems like the *gastrointestinal tract*, *heart and lung*, *musculoskeletal system*, *nervous system*, and the *urogenital tract* including corresponding sub-symptoms. The accordion menu is a common user interface design pattern when a user needs to navigate among main sections while still being able to quickly browse to sub-elements of another section. The mockup in Figure 3.5 illustrates the design. Figure 3.6 shows a screenshot of the actual implementation.

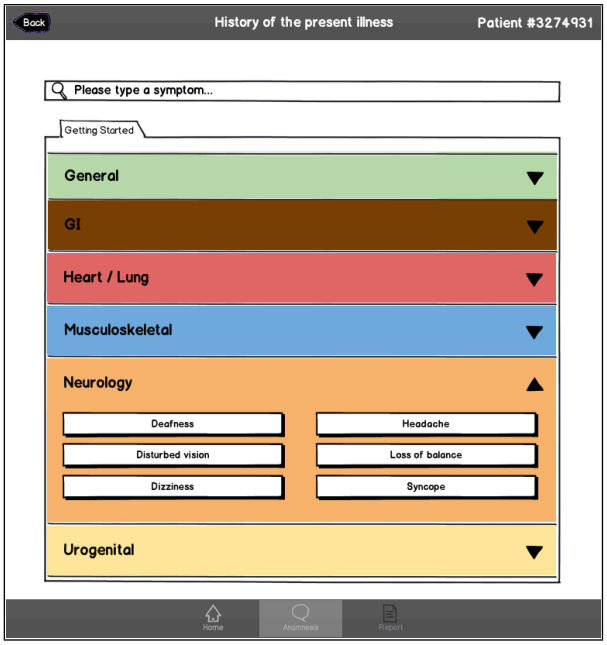

Figure 3.5: Concept of the accordion menu to select the cardinal symptom

History of the present illness (navigate in the graph)

User story As a user, I want to document relevant symptoms.

This part of the application is the main screen and therefore the most important one. All interface elements must be suitable to each other to ensure the best possible user experience. Figure 3.7 shows an early concept of the main screen. On top is the title bar with button to

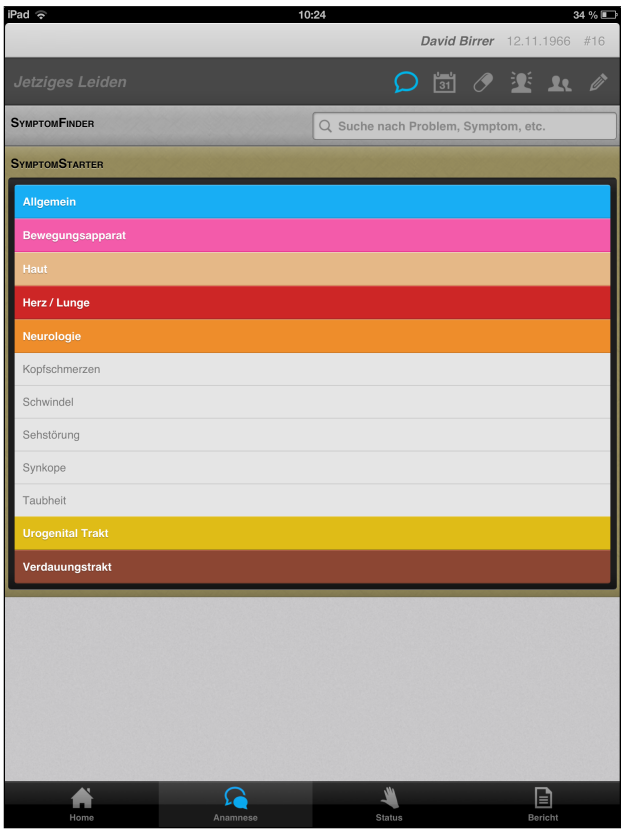

Figure 3.6: Screenshot of the accordion menu

navigate back to the previous page, the page title and information about the current patient (e.g. the name or the patient identifier of the hospital). In the middle section, there is first a search input to find symptoms and problems. If nothing has been searched, the *Symptom Recommender* shows related symptoms. Otherwise, the search results are presented in this view. In the box below all recorded symptoms are listed. In the lowest region is the main navigation to switch to the home screen for instance.

History of the present illness (add details)

User story As a user, I want to add details to a certain symptom.

Adding a symptom either existent or not existent is insufficient. Especially with cardinal symptoms, it is important for later diagnosis to document the precise characteristics of the symptom. For *chest pain* this may be the onset, progression, localisation and radiation, pain characteristics and intensity, provocation and relief, and history of *chest pain*. These details have the character of a static check list. Meaning that does not need a recommender system as a basis.

History of the present illness (group symptoms)

User story As a user, I want to combine symptoms to symptom groups.

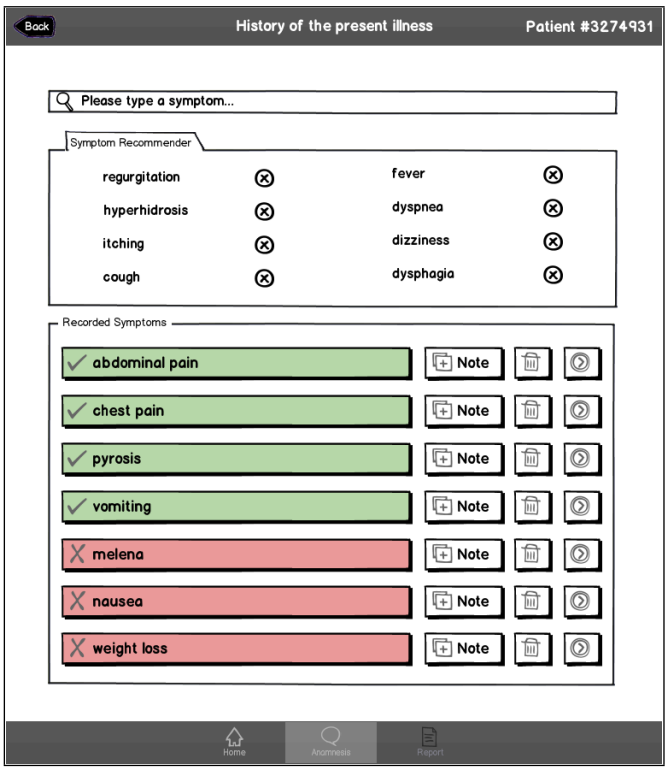

Figure 3.7: Early concept of the main screen

Symptoms can be combined to symptom groups with intuitive drag and drop gestures (see Figure 3.8). Symptoms can be dragged from the problem recommender and dragged either in an already existing group or in a new group. Recorded symptoms can either be existent (✓ in green) or not existent (× in red). The star icon indicates whether it is a cardinal symptom or not.

For comparison, a screenshot of the actual implementation is imaged in Figure 3.9.

History of the present illness (order symptom groups)

User story As a user, I want to arrange symptom groups in an order.

The reordering of symptom groups as well as symptoms within a group is an important task for doctors. With this work they get the symptoms in an appropriate sequence of priority.

3.2.2 Architecture

The *Mobile Application* in this work is built on *JavaScriptMVC*[7]. The framework focuses on the *Model-View-Controller* (MVC) architecture which is illustrated in Figure 3.10 [aja12]. In this concept, the representation of information is separated from the user's interaction with it. The main idea of MVC is code reusability and separation of concerns. [Hal12]

The *model* represents the domain-specific data and knowledge in an application. It publishes the current state to the subscribed controller. The model provides basic functionality to organize

[7] `http://javascriptmvc.com`

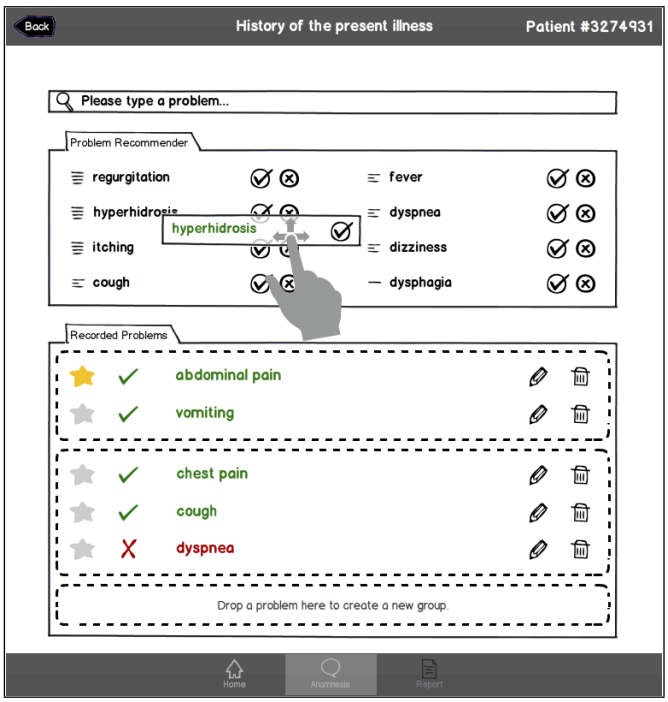

Figure 3.8: Late concept of the main screen

the data layer of the application.

The *view* is typically the user interface in the application. It knows about the existence of models to observe them, but does not directly communicate with them. *JavaScriptMVC* makes use of *Embedded JavaScript*[8] (EJS) templates to render data in *HyperText Markup Language* (HTML) and inject them into the *Document Object Model* (DOM).

The *controller* is able to change the model's state (e.g. property update). It can also send commands to update the view's presentation of the model (e.g. update the DOM). In more detail, the controller is a list of callback functions that get called when the appropriate event is triggered. [wik12c]

[8] `http://embeddedjs.com`

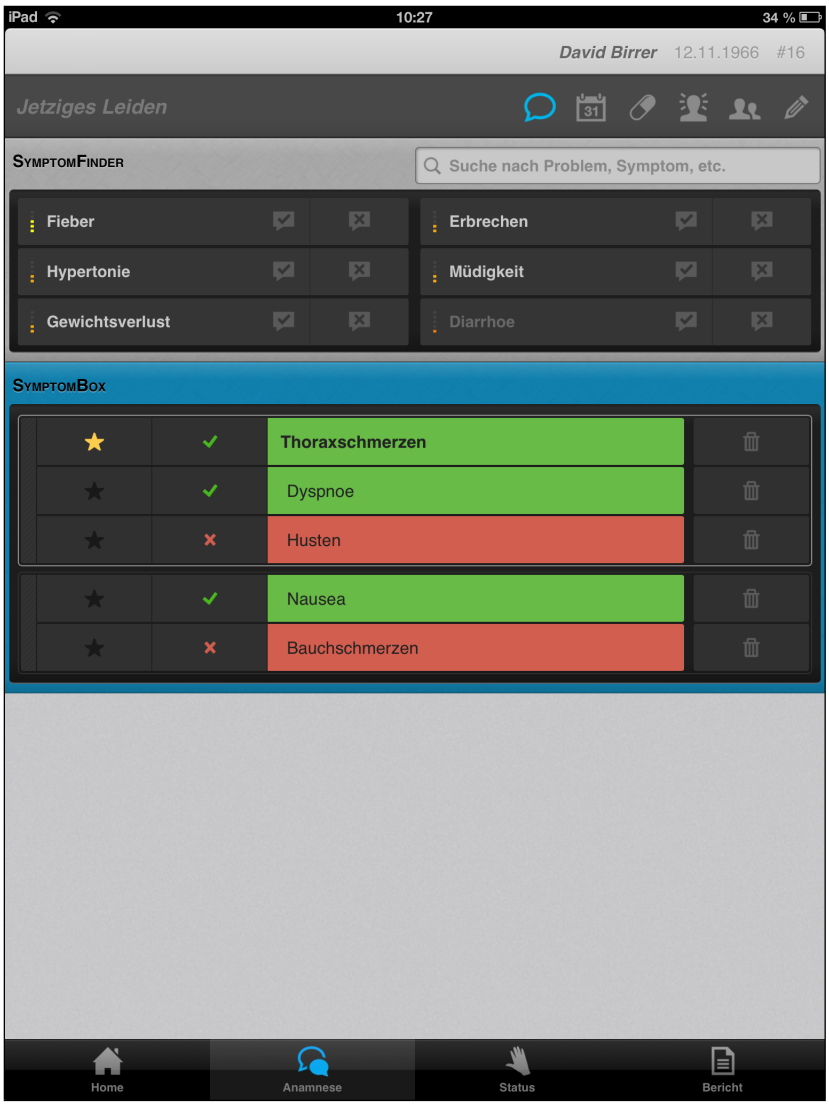

Figure 3.9: Screenshot of the main screen

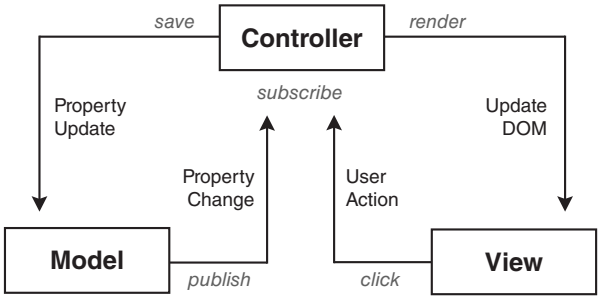

Figure 3.10: Mobile Application Architecture

Chapter 4

Implementation

This chapter presents the implementation of the components we have realized in this work. We start in Section 4.1 by introducing the *Crawler*, followed by the *Mobile Application* (Section 4.2) including the recommender system.

4.1 Crawler

In this work, the *Crawler* considers the sources among others for analysis as listed in Table 4.1. Through a flexible way new sources in text format can easily be added as long it is possible to locate the file by disorder.

Not every source is of equal quality. That is why we have introduced source scores with a simple bonus malus system. For every correct symptom according to the gold standard diagnosis, the algorithm rewards the source with two points. On the other hand, an incorrect symptom means one penalty point. This yields the following ranking when all recorded chest pain gold standards are considered as listed in Table 4.2. In our application, the *Diseases Database* performs best. On the other hand, *Cleveland Clinic* is in the lowest ranking and has been missed out.

The discretized ranking then flows into the weighting for the confidence value.

For the automatic symptom selection the following assumption is made: The more often a particular symptom occurs in various sources, the more relevant is this symptom for the diagnosis.

Figure 4.1 shows the result of the generated symptom list for stroke. Compared to the manually compiled gold standard (true positive), the *Crawler* has reached a sensitivity of 100% (10 out of 10 symptoms). The precision is 11.91%, 10 out of 84 symptoms are correct. The corresponding formulas are listed below [CL09].

Name	**Description**
Diseases Database	Cross-referenced medical dictionary of diseases, medications, symptoms, signs and investigations. [dis12]
Healthline	Doctor-reviewed information about diseases, medical symptoms, and treatments. [hea12]
Mayo Clinic	Guides on diseases and conditions from experts. [may12]
MedicineNet	Articles by doctors on medical diseases and conditions. [med12a]
Medscape	Clinical reference features medical articles. [med12b]
WebMD	Source for health and medical news and information. [web12]
Wikipedia	Wikipedia, the free online encyclopedia. [wik12b]

Table 4.1: Source list

Rank	Source	Score	Absolute Score	Discretized Score
1.	Diseases Database	54	567	6
2.	Better Medicine	8	521	5
3.	Health Central	6	519	5
4.	Family Practice Notebook	-10	503	5
5.	Better Health Channel	-11	502	5
6.	Mayo Clinic	-34	479	5
7.	WebMD	-38	475	5
8.	Virtual Medical Centre	-49	464	5
9.	Medscape	-174	339	3
10.	MedicineNet	-197	316	3
11.	Healthline	-220	293	3
12.	Wikipedia	-254	259	3
13.	Merck Manual	-289	224	2
14.	The Free Dictionary	-384	129	1
15.	Cleveland Clinic	-513	0	0

Table 4.2: Source weights

Sensitivity formula (recall)

$$sensitivity\ (recall) = \frac{\#\ true\ positives}{\#\ true\ positives\ +\ \#\ false\ negatives}$$

Specificity formula

$$specificity = \frac{\#\ true\ negatives}{\#\ true\ negatives\ +\ \#\ false\ positives}$$

Precision formula

$$precision = \frac{\#\ true\ positives}{\#\ true\ positives\ +\ \#\ false\ positives}$$

Accuracy formula

$$accuracy = \frac{\#\ true\ positives\ +\#\ true\ negatives}{\#\ true\ positives\ +\ \#\ false\ positives +\ \#\ false\ negatives\ +\ \#\ true\ negatives}$$

The precision can be increased by introducing an appropriate threshold to filter out unconfident symptoms (symptoms with less appearance). As a threshold we have used a confidence value of at least 25.00% that a specific symptom is listed positive. Figure 4.2 shows the *receiver operating characteristic* (ROC) curve which illustrates the performance of the binary classifier system as the threshold is varied.

The box plot in Figure 4.3 illustrates the achieved accuracy of selected diseases with an existent gold standard. The sample size of diseases presenting with *abdominal pain* is 29 and 41 diseases

presenting with *chest pain* respectively. The median values are 96.06% for *abdominal* and 96.42% for *chest pain*.

The two arrows in Figure 4.1 indicate starting points for manual editing. *Headache* as an example (false positive) could have too many synonyms. Therefore, it is a prominent candidate to approve the synonym set in the symptom ontology. On the other hand *Urinary incontinence* (false negative) may have a too narrow synonym set. This symptom is a candidate to widen the synonym set by adding more suitable synonyms.

Table 4.3 shows the comparison between disease profiles of [Pro10] and machine generated profiles of the *Crawler* of selected diseases. The bullet indicates that a symptom belongs to the corresponding disease.

Causes	**Source**	Chest pain	Cough	Cyanosis	Diaphoresis	Dizziness	Dyspnea	Fever	Hemoptysis	Murmur	Nausea and vomiting	Tachycardia	Tachypnea	Wheezing
Angina pectoris	[Pro10]	•			•	•	•			•	•	•		
	Crawler	•			•		•				•	•		
Asthma	[Pro10]	•	•	•	•		•					•	•	•
	Crawler	•	•	•	•		•	•			•	•	•	•
Lung cancer	[Pro10]	•					•	•	•					•
	Crawler	•	•				•	•	•					•
Myocardial infarction	[Pro10]	•			•		•	•		•	•			
	Crawler	•	•		•	•	•	•		•	•	•		
Pneumonia	[Pro10]	•	•	•	•		•	•				•	•	
	Crawler	•	•	•	•		•	•			•		•	•
Pneumo-thorax	[Pro10]	•	•	•			•						•	
	Crawler	•	•	•			•					•	•	
Pulmonary embolism	[Pro10]	•	•	•	•		•	•	•			•	•	•
	Crawler	•	•	•	•		•	•	•			•	•	•

Table 4.3: Performance of selected causes with chest pain

4.2 Mobile Application

The web-based *Mobile Application* is optimized to work on mobile touch devices with a state-of-the-art web browser. The application gives the ability to users of using the system 24/7 from everywhere. A connection to the Internet is not needed per se. The application works as an offline application to document the patient's disorders. Only for generating the report out of the recorded data a connection to the Internet is required.

4.2.1 Native Development vs. Web Application

A mobile application is software that performs a specific task, such as the management of a calendar, and that is optimized for mobile devices like smartphones and tablet computers.

In mobile application development, two main technologies are distinguished: *Native development* and *Web application*.

We are using *Web application* technology because it has the following benefits over *native development*. Web applications have only one common code base across all platforms and independent release cycles without delay until the release has been approved through a third party application store. In general, mobile Web applications cannot access all of the device's features (yet), but they are not required in this project.

A native developed application is specifically designed to run on a device's operating system and machine firmware. It typically needs to be adapted for different devices. In a Web application, or browser application, some or all parts of the software are downloaded from the Web each time it is run. It can usually be accessed from all Web-capable mobile devices. [Sta10]

A native application developed for the *Apple iPhone* nees to run on its proprietary *iOS* platform, or on *Android* for many other devices, and so forth. The software is installed directly onto the device and users typically acquire these applications through an online store or marketplace. A Web application, however, is generally coded in a browser-rendered language (HTML5, *Cascading Style Sheets* Level 3 (CSS3), combined with JavaScript). The software is accessed through the mobile device's web browser and it does not need to be installed on the device. [Fir10]

There are frameworks and tools available to help in developing application for deployment on multiple mobile platforms (e.g. *Titanium*[1] or *PhoneGap*[2]) and web browsers (e.g. *jQuery Mobile*[3] or *Sencha Touch*[4]).

Various categories in which the two technologies differ fundamentally are listed below.

User Experience

With the development of increasingly faster devices, the difference in the user experience between native and mobile Web applications will be negligibly small. [Hal12]

Capability

Native applications have the capability to interface with the device's native functions, information and hardware (camera, accelerometer, etc.).

Mobile web applications have only restricted access to the device's native functions and information (orientation, geolocation, media library, etc.).

Monetization

For monetization, developers have the ability to charge a download price for native applications and the application store typically handles the payment process (in exchange for a percentage of sales).

Mobile Web applications need to set up their own subscription-based system or monetize through site advertisement.

Versioning

The versioning is a further distinction. While some users may choose to ignore an update, this results in different users running different versions of the native application.

In mobile Web applications, all users are on the same version.

[1] `http://www.appcelerator.com`
[2] `http://phonegap.com`
[3] `http://jquerymobile.com`
[4] `http://www.sencha.com/products/touch`

For completeness, the strengths of native applications are typically faster performance, support of application stores and market places to help user find native applications and provided tools by device manufactures to speed up development. The weaknesses of native applications are more expensive development and maintenance costs when supporting multiple mobile devices and possible delay of the launch or pushing out updates through an application store.

4.2.2 Datastore

The *Mobile Application* of this work makes use of *Web SQL Database*. This client-side database integrated in the web browser allows storing data that can be queried using a variant of *Structured Query Language* (SQL). [Hal12] The specification by the W3C is no longer in active maintenance and the *Web Applications Working Group* does not intend to maintain it further. [w3c12b]

This project makes use of this approach because there is no cross-browser solution as an alternative available yet.

The *Web Applications Working Group* of the W3C has continued to work on two other storage-related specifications: *Web Storage* and *Indexed Database Application Programming Interface* (API). *Web Storage* defines an API for persistent data storage of key-value pair data in Web clients. On the other hand *Indexed Database* defines an API for database of records holding simple values and hierarchical objects. "Each record consists of a key and some value. Moreover, the database maintains indexes over records it stores. An application developer directly uses an API to locate records either by their key or by using an index. A query language can be layered on this API. An indexed database can be implemented using a persistent B-tree data structure" [w3c12a].

4.2.3 Features

With regard to the challenges described in Section 3.2 we created a flexible, fluid layout in CSS3 to ensure the best possible user experience independent of the screen size or the orientation of the device.

In addition, we consequently minimized the number of clicks (taps) to get the right information in various usability test iterations. Users in the healthcare sector have limited knowledge of computer systems. Thus, we adopted screens that are easy to understand and do no need much effort on accessing information. We used comprehensible icons as buttons to navigate to the different screens. All these attempts resulted in serious time reduction to access a service and the requested information.

We have implemented the following features.

Symptom
: Symptoms can be distinguished between guiding symptoms and normal symptoms which can either be existent or not existent.

Details of symptom
: Symptom details can be recorded, including onset, progression, quality, etc. as discussed in user story 3.2.1.

Search
: Additionally to the suggested symptoms by the recommender system, the user can also search for a specific symptom. The search has autocomplete support: the software tries to predict a word that the user wants to type in without the user actually typing it in completely [HB11]. Moreover the search is fault-tolerant to a certain degree (one character may

be mistyped or missing completely) and allows synonyms (Epistaxis vs. Nosebleed).

Additional medical information
Record personal history, medications, allergies of the patient and family history, each consisting of a customized form. In medications, there is an input field for day plan and the medication itself. With a click in the day plan a list opens with commonly used entries like "0-0-1", "1-0-0", "1-0-1" and "if necessary". It works similar to the autocomplete functionality to reduce the user's text entry effort, and to reduce errors [HB11]. This is also implemented for the record of the information mentioned before with a useful vocabulary list. As an example, the allergy vocabulary list begins with *acrylamide*, *allergen*, *allergy*, ends with *walnut* and includes 218 items in total.

Notes
It is always possible that a particular symptom is not present in the system. Nevertheless, the doctor must be able to document this problem. In a separate section, the user can write notes in free text.

4.2.4 Recommender System

The recommender system has the task, starting from a symptom to propose other related symptoms. Therefore, the system accesses the data that have been generated by the *Crawler*. Out of all crawled diseases profiles, the *Crawler* generates a matrix in which every symptom gets a similarity to all other symptoms. As a simplification, the values get normalized between 0 and 1.

A simple example of how the algorithm works is illustrated below.

Initial point

An extract of the similarity matrix is shown in Table 4.4. The matrix is symmetric. A value of 1.00 means strongest similarity and a value of 0.00 means weakest similarity. Therefore *abdominal pain* has to itself the highest possible similarity of 1.00. *Nausea* and *vomiting* are strongly related to each other and have very similar values.

	abdominal pain	**chest pain**	**cough**	**dyspnea**	**nausea**	**vomiting**	**weight loss**
abdominal pain	1.00	0.13	0.22	0.23	0.77	0.77	0.58
chest pain	0.13	1.00	0.59	0.64	0.31	0.27	0.35
cough	0.22	0.59	1.00	0.71	0.40	0.41	0.48
dyspnea	0.23	0.64	0.71	1.00	0.44	0.39	0.46
nausea	0.77	0.31	0.40	0.44	1.00	1.00	0.70
vomiting	0.77	0.27	0.41	0.39	1.00	1.00	0.73
weight loss	0.58	0.35	0.48	0.46	0.70	0.73	1.00

Table 4.4: Similarity matrix

Step 1: Adding Chest Pain

In the first step, we are adding *chest pain*. Based on this symptom, the algorithm starts with recommendations as shown in Table 4.5. With only one symptom, the summation is trivial. It is striking that *chest pain* with the highest value was filtered out. Already recorded symptoms does not need to be suggested once again.

	abdominal pain	**chest pain**	**cough**	**dyspnea**	**nausea**	**vomiting**	**weight loss**
chest pain	0.13	1.00	0.59	0.64	0.31	0.27	0.35
SUM	0.13	1.00	0.59	0.64	0.31	0.27	0.35

Table 4.5: Recommender System Step 1: Adding chest pain

This leads to the recommendations as listed in Table 4.6.

1.	**dyspnea**	0.64
2.	**cough**	0.59
3.	**weight loss**	0.35
4.	**nausea**	0.31
5.	**vomiting**	0.27
6.	**abdominal pain**	0.13

Table 4.6: Recommender System Step 1: Recommendations

Step 2: Adding Dyspnea

In the second step, we add the top recommendation of the system: *dyspnea*. The summation with the two symptoms is shown in Table 4.7.

	abdominal pain	**chest pain**	**cough**	**dyspnea**	**nausea**	**vomiting**	**weight loss**
chest pain	0.13	1.00	0.59	0.64	0.31	0.27	0.35
dyspnea	0.23	0.64	0.71	1.00	0.44	0.39	0.46
SUM	0.36	1.64	1.30	1.64	0.75	0.66	0.81

Table 4.7: Recommender System Step 2: Adding dyspnea

By ordering the summations we get the new recommandations as shown in Table 4.8.

1.	**cough**	1.30
2.	**weight loss**	0.81
3.	**nausea**	0.75
4.	**vomiting**	0.66
5.	**abdominal pain**	0.36

Table 4.8: Recommender System Step 2: Recommendations

Step 3: Adding Nausea

The third step is very similar to the previous steps. Table 4.9 shows the according calculations when adding the symptom *nausea*.

	abdominal pain	**chest pain**	**cough**	**dyspnea**	**nausea**	**vomiting**	**weight loss**
chest pain	0.13	1.00	0.59	0.64	0.31	0.27	0.35
dyspnea	0.23	0.64	0.71	1.00	0.44	0.39	0.46
nausea	0.77	0.31	0.40	0.44	1.00	1.00	0.70
SUM	1.13	1.95	1.70	2.08	1.75	1.66	1.51

Table 4.9: Recommender System Step 3: Adding nausea

The recommendations for *chest pain*, *dyspnea*, and *nausea* are listed in Table 4.10.

1.	**cough**	1.70
2.	**vomiting**	1.66
3.	**weight loss**	1.51
4.	**abdominal pain**	1.13

Table 4.10: Recommender System Step 3: Recommendations

Listing 4.1 outlines the basic algorithm implemented in JavaScript described above. *patientNode.idNode* corresponds to the symptom identifier.

```
1 var that = this;
2 var idNodeMap = {};
3
4 // Loop through patient nodes
5 _.each(symptomPatientNodeList, function(patientNode) {
6    if(that.similarityMap[patientNode.idNode] != null &&
7       that.similarityMap != undefined) {
```

```
8
9         // Loop through similarities of patientNode
10        _.each(that.similarityMap[patientNode.idNode],
11            function(similaritySum, idNode) {
12
13            // Initialize entry in map if needed
14            if(idNodeMap[idNode] == undefined
15               idNodeMap[idNode] == null) {
16
17               idNodeMap[idNode] = 0;
18            }
19
20            if(patientNode.isEnabled()) {
21               // Sum up if enabled
22               idNodeMap[idNode] += similaritySum;
23            }
24        });
25    }
26 });
```

Listing 4.1: Recommender algorithm in JavaScript

Further improvements

The values with which the order is determined is displayed visually in the graphical user interface. This shows the user how certain the system is with the recommendation of the symptom. The grading is as follows: 5 green bars represent very high confidence, 3 yellow bars stand for medium confidence and only 1 orange bar expresses doubtfulness in the recommendation.

In steps 1 to 3 above the base algorithm of the recommender system was described. This method does not yet account for negative symptoms (e.g. the patient is not suffering from *fever*). In an extended version, we considered these cases also. The approach is practically the same, instead of adding the value, it gets subtracted. Thus, the effect is not too heavy, the subtrahend is weighted (20% of the original value).

In the system, we distinguish between cardinal symptoms and associated symptoms. To improve the suggestions, the system should weigh cardinal symptoms higher than associated symptoms. In our approach, the values of the cardinal symptoms are multiplied with 4. This factor turned out to be adequate.

To make the suggestions of the recommender system even more intelligent, it takes the patient's gender into account. If the patient is female, all male specific symptoms like *erectile dysfunction* are filtered out from the recommendation list and vice versa.

Stroke

ICD-10 I64

Rank	Symptom	Confidence	# Sources	True Positive	Comment
1.	Hypertension	87.50%	7	x	
	Weakness	87.50%	7	x	
	Bleeding	87.50%	7		Candidate for synonym *removal*.
	Headache	87.50%	7		Candidate for synonym *removal*.
5.	Vomiting	75.00%	6	x	
	Confusion	75.00%	6	x	
	Paresthesia	75.00%	6		Candidate for synonym *removal*.
8.	Dizziness	62.50%	5	x	
	Edema	62.50%	5		Candidate for synonym *removal*.
10.	Hematochezia	50.00%	4		
	Syncope	50.00%	4		
	Loss of consciousness	50.00%	4	x	Candidate for synonym *widening*.
	Melena	50.00%	4		
	Hemorrhage into skin	50.00%	4		
	Odynophagia	50.00%	4		
	Dysarthria	50.00%	4	x	Candidate for synonym *widening*.
	Aphasia	50.00%	4	x	Candidate for synonym *widening*.
	Dysphagia	50.00%	4	x	Candidate for synonym *widening*.
19.	Wound	37.50%	3		
	Seizure	37.50%	3		
	Diarrhea	37.50%	3		
	Abdominal pain	37.50%	3		
	Cough	37.50%	3		
	...				
34.	Pallor	25.00%	2		
	Deafness	25.00%	2		
	Urinary incontinence	25.00%	2	x	Candidate for synonym *widening*.
	Hypotension	25.00%	2		
	...				

Sensitivity (Recall)	100.00%	(10 of 10)
Specificity	72.99%	(200 of 274)
Precision	11.91%	(10 of 84)
Accuracy	73.94%	

Sources with hits 9

Figure 4.1: Stroke with candidates for enrichment

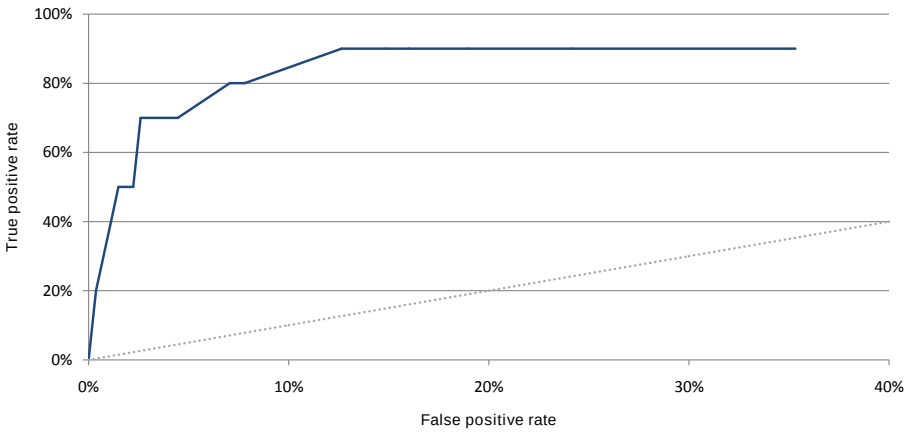

Figure 4.2: ROC curve for Stroke

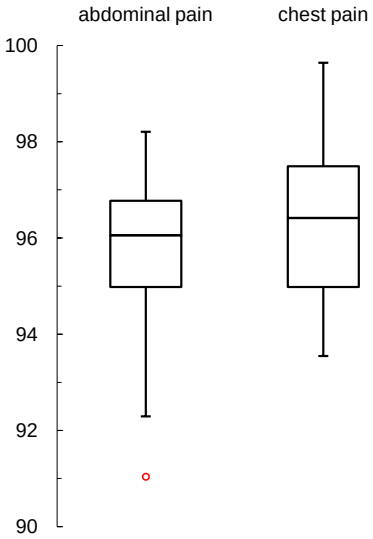

Figure 4.3: Box plot of the achieved accuracy (in percentage)

Chapter 5

Evaluation

Our vision is to design an intuitive and easily usable user interface that allows medical doctors to make the full patient documentation during the consultation. To evaluate whether our system meets this intention, we created and conducted a user study with 10 participants. This chapter includes the *Study Settings* describing the experiment, followed by the design and *Results* of the survey. Finally, possible *Threats to Validity* are presented.

5.1 Study settings

We performed an evaluation of our application to find answers to the following key questions:

- Can the application be used beneficially to assist medical doctors during patient history taking?
- Is the user interface intuitive and easy to use?
- Are users constrained by the system or are they free to act?
- Do users prefer to use the system during the consultation or afterwards?
- Which device do users prefer to utilizing our system?
- Are the users generally satisfied with the prototype?

The experiment was set up as follows: On the operational plan of 10 junior doctors working at the *UniversitätsSpital Zürich* a test session of 30 minutes was reserved for each participant. After a brief explanation about the course of the test session, the participants were given a 5 minutes introduction to the tool on an *Apple iPad* (3rd generation). Thereafter, the attendees had to interview a briefed patient and document his complaint with our system. The case further described in the next section lasted about fifteen minutes. At the end, the participants were asked to complete a questionnaire on the usefulness and usability of the prototype. We opted for a paper-and-pencil survey that is quick and easy to complete and asked the participants to fill out the survey immediately after the experiment. This method has been proved in thousands of studies. [Ozo08]

For this study, the system had integrated an algorithm to generate a medical report in prose out of the recorded symptoms and problems. The participants were able to read the formulated report at the end of their session. The algorithm mentioned before is not in the scope of this thesis. Nevertheless, it was put in action to better illustrate the usefulness of the system in its entirety during this study with the medical doctors.

Due to technical difficulties with connectivity and access to the *Virtual Private Network* (VPN) in the hospital we had to perform the last four experiments on a laptop. The advantage is that the software runs smoother on a laptop (especially the CSS3 transitions for effects) than on an *iPad*. Tests on newer devices with a faster central processing unit (CPU) have shown that this problem

occurs much less. The disadvantage is that the feeling is different on a conventional laptop with mouse than on a touch device. During the conversation, the doctor focuses more on the keyboard than on the patient and the documentation process is less casual (cf. paper pad and pen) than on a mobile device.

5.1.1 Case

The selected case for this experiment is heavily based on case number 6 "*46-Year-Old Man with Chest Pain*" in [LBSAS09]. This book is designed for medical students to prepare for exams and contains cases of patients presenting certain symptoms. The participants of this study were given the information as follows.

Doorway Information

Opening Scenario:
David Birrer, a 46-year-old male, comes to the emergency room suffering from *chest pain*.

Examinee Tasks:
1. Prepare for the ward round (cf. Figure 5.1)
2. Take a focused history and document the patient's disorders (cf. Figure 5.2)

Sie sind Arzt für Innere Medizin am UniversitätsSpital Zürich und Ihr nächster Patient heisst **David Birrer**, geboren am **12.11.1966**.

Er kommt mit **Thoraxschmerzen** auf den Notfall. Bereiten Sie sich für die Visite vor.

Figure 5.1: Task 1

Führen Sie beim Patienten David Birrer eine **Anamnese** durch und **dokumentieren** Sie seine Beschwerden.

Figure 5.2: Task 2

Profile of the patient's disorders

The briefed patient was acting as follows.

History of presenting complaint
Table 5.1 shows the behaviour of the patient on selected questions in the case.

Question	Patient Response
Chief complaint	*Chest pain*
Onset	Three hours ago
Precipitating events	Nothing
Progression	Constant severity
Severity on a scale	7/10
Radiation	To my neck and left arm
Quality	Pressure
Alleviating / exacerbating factors	Nothing
Shortness of breath	Yes
Nausea / vomiting	I feel nauseated, but I did not vomit
Sweating	Yes
Cough	No
Wheezing	No
Abdominal pain	No
Diarrhea	No
Constipation	No
Previous episodes of similar pain	No

Table 5.1: Overview history of presenting complaint

Personal history
The patient states to have had his *appendix* removed in 1996.

Medications
The patient says to take the medicine *Paracetamol 500 mg Grünenthal* for *headaches*.

Allergies
The patient suffers from *hay fever* in the summer.

Family history
The patient states that his maternal grandmother suffered a *heart attack* at the age of 66 years.

5.1.2 Incentives

As an appreciation, Ozok suggests to compensate the test participants for their time [Ozo08]. Krug lists in [Kru09] various incentives for compensation to offer the people including payment in kind (e.g. merchandising articles), gift certificates or cash.

Our test participants were given a small package with goodies as shown in Figure 5.3 after completing the questionnaire. It includes a ball pen shaped as a syringe, a set with sticky notes looking like band-aids, and a Swiss chocolate.

Figure 5.3: Incentive package

5.2 Results

After completing the task with the case, the participants are asked to fill in the questionnaire. The questionnaire consists of six parts as follows: Personal information, perceived usefulness, perceived ease of use, field of application, satisfaction and open-end questions.

The questions of the individual parts are based largely on proven questionnaires by different authors. The sources are mentioned in the relevant sections. Only the choices have sometimes been standardized in order not to unnecessarily confuse the user.

In total 10 people have been invited to the experiment. In one case the experiment could only start with a delay of 15 minutes. Instead of the planned 30 minutes, the session took only half as long and the experiment with the case could not be undertaken. Thus, there are 9 valid questionnaires in total that can be examined.

As an overview the results of the questionnaire are visualized in Figure 5.4 with mean and median. The alternately order of positive and negative statements from statement C1 to C10 is eye-catching. In the next section the results are discussed in more detail.

A. Personal information

A1 Gender of the test participants

The study was run with 4 males and 5 females.

A2 Age of the test participants

The age of the participants was between 27 and 42 years with a median of 30.

A3 Function in the hospital

The test participants all work as junior doctors at the *UniversitätsSpital Zürich* in Switzerland.

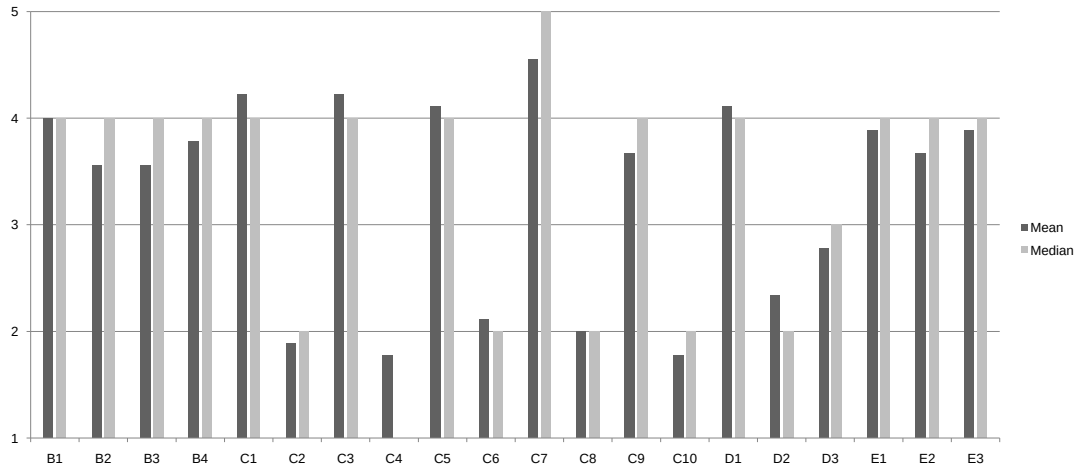

Figure 5.4: Results overview. The x-axis stands for the identifiers of the statements. The y-axis shows the likert scale from 1 (strongly disagree) to 5 (strongly agree).

B. Perceived usefulness

The statements in section B are based on the work of [Dav89] and focus on the perceived usefulness. Since we entirely used the questionnaire *System Usability Scale* of [Bro96] in section C, we have omitted overlapping statements. The outcome of all statements is shown in Table 5.2.

The first statement (B1) was taken from [Dav89]. Statement B2 is geared to "I felt comfortable using this system." from [Lew95]. The next statement (B3) checks if the user feels restricted by the system or if the system let them freedoms at work. Statement B4, the last of this section, shows if the user finds the system useful in general. It is also used in [Dav89] as number 6 in section *perceived usefulness*.

#	Statement	Mean	Median	STDV
1	Using the system in my job would enable me to accomplish tasks more quickly. [Dav89]	4.00	4.00	0.71
2	Using the system in my job would make my tasks more comfortable.	3.56	4.00	0.53
3	The system gives me freedoms to accomplish my tasks.	3.56	4.00	0.88
4	I would find the system useful in my job. [Dav89]	3.78	4.00	0.83

Table 5.2: Results for statement in B. (1 = *Strongly disagree*; 5 = *Strongly agree*)

C. Perceived ease of use

To measure the ease of use, we have used the System Usability Scale (SUS) introduced by John Brooke in 1986. Originally it was a quick and dirty approach, but over the years "it has become an industry standard with references in over 600 publications" [Sau12].

SUS is based on a *Likert scale* with forced-choice questions. The statement is made and the respondent then indicates the degree of agreement or disagreement respectively with the statement on a 5 point scale. The scale reaches from *Strongly disagree* (1) to *Strongly agree* (5).

The advantage of the SUS is its score. The 10 statements are used to measure the overall usability of the system being studied. The SUS score has a range from 0 to 100 and can be compared with the scores of other systems. [Bro96]

Since the statements of SUS are standardized, but officially just available in English, we have investigated in a German translation as accurately as possible. Our solution is based on the proposals of the initiative of *Crowdsourcing the translation of SUS* [Rei12]. The German translation of the 10 statements can be found in the appendix in Section 7.4.

C1 - C10

The 10 items (cf. Table 5.3) should not be considered individually for themselves. The calculated overall SUS score discussed in the next section is meaningful.

#	Statment	Mean	Median	STDV
1	I think that I would like to use this system frequently. [Bro96]	4.22	4.00	0.67
2	I found the system unnecessarily complex. [Bro96]	1.89	2.00	0.60
3	I thought the system was easy to use. [Bro96]	4.22	4.00	0.44
4	I think that I would need the support of a technical person to be able to use this system. [Bro96]	1.78	1.00	1.09
5	I found the various functions in this system were well integrated. [Bro96]	4.11	4.00	0.78
6	I thought there was too much inconsistency in this system. [Bro96]	2.11	2.00	0.60
7	I would imagine that most people would learn to use this system very quickly. [Bro96]	4.56	5.00	0.73
8	I found the system very cumbersome to use. [Bro96]	2.00	2.00	0.71
9	I felt very confident using the system. [Bro96]	3.67	4.00	0.50
10	I needed to learn a lot of things before I could get going with this system. [Bro96]	1.78	2.00	0.97

Table 5.3: Results for statements in C (1 = *Strongly disagree*; 5 = *Strongly agree*)

SUS Scores

The SUS score is calculated as follows: "For positively-worded items (1, 3, 5, 7 and 9), the score contribution is the scale position minus 1. For negatively-worded items (2, 4, 6, 8 and 10), it is 5 minus the scale position" [LS09]. The sum of the score contribution is multiplied by 2.5. This results in a total score between 0 and 100. Please note that although the SUS score can range from 0 to 100, it is not a percentage.

Sauro has reviewed existing research on SUS while analyzing 500 different evaluations containing data of around 5000 users. Over these 500 studies it results in an average SUS score of 68. A SUS score with less than 68 points would be considered below average and any higher score would be considered above average. [Sau12]

Our application reached a SUS score of 77.5 (median). The individual SUS scores are between 62.5 and 90 points. All results are summarized in Table 5.4. Figure 5.5 shows the histogram with the distribution of the SUS scores and the box plot in Figure 5.6 further illustrates the SUS scores. Please note that the median is exactly on the lower quartile (Q1).

	Mean	Median	STDV
SUS Score	78.06	77.50	7.68

Table 5.4: SUS Scores (from 0 to 100)

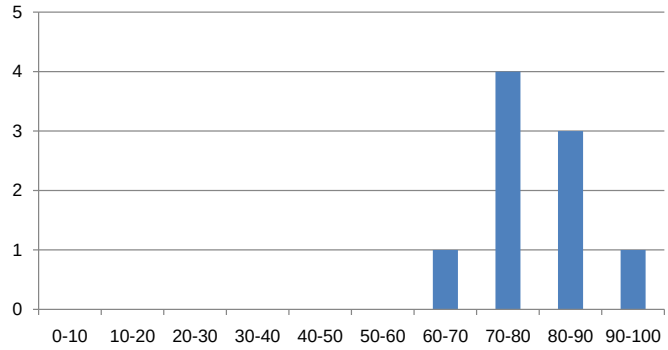

Figure 5.5: Histogram of SUS scores. The x-axis shows the SUS scores in steps of 10 and the y-axis the occurrence of each score.

D. Field of application

The statements in this section relate to the field of application of the system. This involves the question when the system is used and on what medium. The results are listed in Table 5.5.

#	Statement	Mean	Median	STDV
1	I think that I would use this system *during* the doctor‘s consultation.	4.11	4.00	0.60
2	I think that I would use this system *after* the doctor‘s consultation.	2.33	2.00	1.12
3	I think that I would prefer to use the system on a common computer with mouse.	2.78	3.00	1.20

Table 5.5: Results for statements in D. (1 = *Strongly disagree*; 5 = *Strongly agree*)

D1 Time (I) *I think that I would use this system during the doctor‘s consultation.*
Statement D1 refers that the system is used during the doctor‘s consultation. This means the doctor documents on the fly and live at the bedside of the patient respectively. All 9 participants agree on this statement at a mean of 4.11.
Conclusion: The 100 percentage agreement shows that the system is designed to operate during the doctor‘s consultation.

D2 Time (II) *I think that I would use this system after the doctor‘s consultation.*
The next statement (D2) is in contrast to the previous statement. Here is the idea that the doctors as in the past make their notes with notepad and pen. In a free moment they document the complaints with our system.
Conclusion: People more likely reject this statement (2.33). But they do not share the same

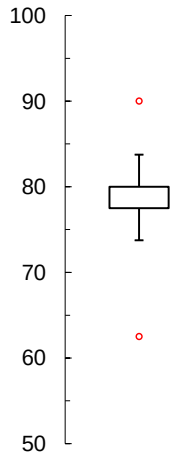

Figure 5.6: Box plot of SUS scores

opinion (standard deviation is 1.12). One possible explanation is that people prefer a post-processing until they have become accustomed to the system and feel confident to use it in front of the patient. The biggest time saving can be achieved when doctors use the system live during the consultation.

D3 Device *I think that I would prefer to use the system on a common computer with mouse.*
Statement D3 is targeted on the device. The system is designed for mobile devices with touch screen capabilities, but can also be used on a conventional computer with mouse. We wanted to find out which platform the test participants prefer. The median is at 3.00 ("I neither agree nor disagree") with a standard deviation of 1.20. Figure 5.7 illustrates this variance.

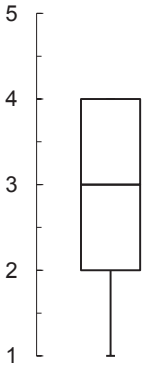

Figure 5.7: Box plot of D3 (1 = *Strongly disagree*; 5 = *Strongly agree*)

Conclusion: The participants have different opinions. We performed the test run on an *Apple iPad* and a conventional laptop. These varying devices may have influenced the test results. Another explanation is the fact that in our experiment a computer was available in the consulting room. Thus, the medical doctors are already accustomed to work on a

computer with keyboard and mouse. Only two participants have indicated that they own a tablet computer. It is possible that the others feel too insecure to work with such a device.

E. Satisfaction

This section focuses on the satisfaction in general. Two of the three statements relate to the findings of [Lun01] in the category *Satisfaction*. Results can be found in Table 5.6.

#	Statement	Mean	Median	STDV
1	I find the graphical user interface appealing.	3.89	4.00	0.93
2	I am satisfied with the system. [Lun01]	3.67	4.00	0.71
3	I would recommend the system to a friend. [Lun01]	3.89	4.00	0.33

Table 5.6: Results for statements in E. (1 = *Strongly disagree*; 5 = *Strongly agree*)

E1 Graphical User Interface *I find the graphical user interface appealing.*
Since we have paid particular attention to the graphical user interface, we wanted to test if the design is appealing. Many computer systems in the medical field have an outdated appearance. We would like to stand out with an attractive design. 7 of 9 participants have indicated that they find the graphical user interface appealing.
Conclusion: Generally, the design has been well received and the effort has paid off.

E2 Satisfaction *I am satisfied with the system.*
Statement E2 corresponds to the item 26 in [Lun01]. The calculated mean is 3.67 and the median is 4.00 ("Agree").
Conclusion: Most of the participants state that they are satisfied with the system.

E3 Recommend to a friend *I would recommend the system to a friend.*
This statement intends to find out if people recommend the system to a friend as used in [Lun01]. Recommending a product to a friend is very personal and can be called word-of-mouth (marketing). It is believed that this kind of marketing is more credible than advertisement, because of the personal nature of the communications between individuals. The listener assumes that the communicator is honest and does not have an ulterior motive. [GCD03]
8 out of 9 participants would recommend the system to a friend.
Conclusion: Sauro notes that the point where users are more likely to be recommending the product to a friend is a SUS score above 80.30 [Sau12]. In this work, we have reached a SUS score of 78.06 (mean) and 77.50 (median) respectively which is close to Sauro's cut-off.

F. Open-end questions

In the last section of the questionnaire the subjects were asked to answer three questions in free text. The initial question states important items that should have to be met before the subjects will use the system. The other questions address positive and negative aspects of the system.

The suggestion to implement an expansion of the systemic history was mentioned far more frequently than any other idea for improvement. The doctor should have the opportunity to capture symptoms by organ system.

At the positive aspects the neat graphical user interface was often remarked. One of the biggest

challenges seems to be the typing on the touch device which should be kept as minimal as possible.

The detailed outcome is listed below.

What would have to be met before I use the system?

- *"Expansion of the systemic history"*
- *"Recording the vital parameters (blood pressure, pulse, ...)"*
- *"Minimization of typing in the 1-finger system (on touch device)"*
- *"90% of the possible answers need to be clickable"*
- *"Recording the diagnosis"*
- *"Recording also normal findings"*
- *"Integration of the compendium (medication database) with adverse effects"*

Positive aspects

- *"Quick and simple"*
- *"Well arranged and easy handling"*
- *"Good design"*
- *"In chest pain wide range of items to describe the characteristics of the symptom precisely"*
- *"2 in 1: conducting the history of the present illness and writing the documentation in one system"*
- *"Quality assurance: Forgetting no major questions"*
- *"Automated wording of the medical report"*

Negative aspects

- *"Typing on a touch device"*
- *"Less eye contact with the patient, because you focus on the tablet computer → seems very impolite"*
- *"Error-proneness of wireless networks"*
- *"The system provides questions that mislead me from my own concept"*
- *"Import from other systems is not supported (e.g. medication list)"*

Conclusion: Right now many subjects are using a computer during the consultation. With sufficient practice eye contact with the patient should become easier on a mobile device than working on a computer with keyboard.
The quality of the recommended symptoms has not been criticized in any form. Thus, we argue that our recommender system approach is well elaborated.

5.3 Threats to Validity

Possible reasons that inferences or conclusions of this study might be wrong with plausible explanations are discussed as follows. [wik12d]

Conclusion Validity

Threats to *Conclusion Validity* may derive from a too small number of participants in our research. The *sample* includes 10 people, one questionnaire had to be cancelled. The small sample size leads to decreased precision in statistical power. The result indicates at most

a trend. According to [Sau12] SUS (Section C) can be used on very small sample sizes (as few as two users) while still delivering reliable results. But "small sample sizes generate imprecise estimates of the unknown user-population SUS score" [Sau12].

Internal Validity

Internal Validity deals with uncontrolled factors that may have influenced our outcomes regarding the experimental group.

Our experimental group is very homogeneous. All participants work as junior doctors in the same hospital. The opinion of senior physicians and general practitioners would be interesting.

Some candidates seemed generally *stressed* by their clinical work (not by the experiment), and others a little distracted. A possible reason for this could be the weekday of the experiment. Half of the experimental group had their schedule on Monday morning.

Construct Validity

Construct Validity is the extent of what has to be measured to what was actually measured. Regarding this the *task* with the patient example from the textbook was possibly too simple. The support of the system especially during the documentation process could be insignificant. If a patient is examined with a foreseeable disease, then the documentation is not very complicated either. The doctor may be faster with a paper pad and pen than with the system which generates the medical report. The time saving would have been more impressive in a more complicated case.

It is possible that parts of the questionnaire we used to measure usability is not an adequate vehicle for our approach. We believe that this threat is relatively low, given that SUS was used effectively in thousands of usability assessments over various application domains and that numerous researchers attest an excellent reliability to the questionnaire.

Is it possible to estimate the real value of the system in such a *short time*? The test participants spent less than 20 minutes with the prototype. They may form their opinions based on the actual prototype and perhaps less on the concepts in general. This assumption is reinforced because the prototype was often compared during the conversation with the current patient management software *KISIM*[1] that is used at the *UniversitätsSpital Zürich*. In addition, a few participants complained of inappropriate words in the autocomplete search. This deficiency would be easy to fix by manual examination of the vocabulary.

External Validity

External validity is the extent of generalized inferences transferred to other situations. Regarding this matter, our subjects may not have been a *representative sample* of medical doctors in general. However, we only invited junior doctors that have completed their studies and have at least one year of working experience.

[1] `http://www.cistec.ch`

Chapter 6

Summary and Conclusions

This chapter summarizes the insights how tablet computers can be beneficially used for patient documentation work in daily medicine. Additionally, suggestions on how to improve our application are presented.

6.1 Conclusions

During this work, we first developed two medical ontologies for symptoms and diseases to meet our requirements for the *Crawler*. Among others, classes need synonyms to find disease and symptom names in various text sources.

Then we implemented the *Crawler* to generate disease profiles including probable symptoms out of many sources available on the Internet. The data is used to calculate similarities between symptoms to train a recommender system suggesting related symptoms the doctor may ask next during the consultation.

The *Mobile Application* provides a simple to use graphical user interface to cover various aspects of the patient documentation like current complaint, personal history, medications, allergies, and family history. The typing effort has been dramatically reduced by recommending related symptoms and autocomplete functionality with helpful and context sensitive suggestions.

The evaluation of the prototype shows that the implemented system has the potential to make the documentation work of medical doctors more efficient. Through an easy-to-use system, the doctor is able to make the documentation directly during the consultation and thus saves time. There is less overhead because the doctor does not have to take notes during the consultation and formulate the documentation afterwards. Most of the participants of the survey state that they are generally satisfied with the system.

With the possibility of guidance to ensure that less is forgotten during patient history taking, consultations can be more target-oriented. This capability is not exhausted in this work and has the potential to make medicine safer and more efficient.

The prototype developed in this work demonstrates the practical benefits in clinical practice.

6.2 Future Work

This section highlights different starting points to further improve our system.

Crawler

In this work, the *Crawler* is restricted to the most important diseases presenting with either *abdominal* or *chest pain*. The *Crawler* could expand its operations to other fields in medical science and generate data with diseases beyond *abdominal* and *chest pain*. The currently used source set can be extended with any source available in text format as long as the document can be retrieved by disease. Carefully selected sources can further increase the quality of the generated disease profiles.
A further improvement of the *Crawler* would be the distinction between "cough" and "no cough". This is a hard semantic problem since the negation can be nested into each other. However, predefined sentence structures would be feasible.

Mobile Application Features

There are plenty of features to improve the developed mobile prototype in this work. As mentioned several times in this study, the system needs to give the possibility to record the systemic history. The doctor would be able to systematically ask for symptoms by organ system. Another similar requirement is the possibility to document the status with findings of examinations.
Comparable to personal history, noxious agents like smoking or drinking alcohol need to be recorded in a proper way.
The implemented autocomplete functionality in the prototype should be enhanced to reduce tedious typing effort on the touch device even more.
An interesting feature would be the taking of pictures directly from the application. The photographical documentation of skin diseases could be especially interesting.
Not only the documenting process can be improved, but also the view of the recorded information in form of a summary has potential to quickly get an overview of a patient's complaints.

Data processing

In this work, we focus on the input of patient data. Once the data is entered completely, it can be processed for further use. One possibility is the generation of a report in flowing text. The writing of a medical report is tedious work and computer support would be a relief.
Another application could be the suggestion of possible diagnoses that match the entered symptoms.

Further studies

Our sample for the experiment was relatively small. If the prototype is more mature, a study with a larger sample and over a longer period of time will be appropriate. In a real setting new challenges come to light.

Chapter 7

Appendix

In this appendix we present resources used in this work in Section 7.1, provide a manual for using the software components (Section 7.2), and list the contents of the CD-ROM in Section 7.3. Section 7.4 shows the questionnaire used for the study, the results of the evaluation are listed in Section 7.5, and an example excerpt of a medical record is illustrated in Section 7.6.

7.1 Resources

The solution of this work has been implemented using the software libraries listed below.

7.1.1 Libraries

Crawler

google-gson - `http://code.google.com/p/google-gson`
Gson is a Java library that can be used to convert Java Objects into their JSON representation and vice-versa.

jsoup - `http://jsoup.org`
jsoup (Java HTML Parser) is a Java library for working with real-world HTML which provides an API for extracting and manipulating data.

The OWL API - `http://owlapi.sourceforge.net`
The OWL API is a Java API and reference implementation for creating, manipulating and serialising OWL Ontologies.

Mobile Application

FastClick - `https://github.com/ftlabs/fastclick`
FastClick is a JavaScript library for eliminating the 300ms delay between a physical tap and the firing of a click event on mobile browsers.

Hammer.js - `http://eightmedia.github.com/hammer.js`
Hammer.js is a JavaScript library for multi-touch gestures.

JavaScriptMVC - `http://javascriptmvc.com`
JavaScriptMVC (JMVC) is a JavaScript framework that builds maintainable, error-free, and lightweight applications as quick as possible.

jQuery - `http://jquery.com`
jQuery is a JavaScript library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for web development.

jQuery UI - `http://jqueryui.com`
jQuery UI is a curated set of user interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript library.

Moment.js - `http://momentjs.com`
Moment.js is a JavaScript date library for parsing, validating, manipulating, and formatting dates.

Underscore.js - `http://underscorejs.org`
Underscore is a JavaScript library that provides functional programming support.

7.2 Installation Guidelines

This section provides a step by step guide to install and run the software.

7.2.1 Crawler

1. Build the maven[1] project (`Crawler/pom.xml`)
2. Adjust the diseases considered by the *Crawler* in the *Load Manager* (`Crawler/main/java/core/LoadManager.java`)
3. Adjust the sources considered by the *Crawler* in the *Source Manager* (`Crawler/main/java/core/SourceManager.java`)
4. Run `Crawler/main/java/Application.java`
5. Hit *y* to crawl new data.
6. Hit *y* to aggregate data.
7. Find the generated documents in the `Crawler/output/` directory.

7.2.2 Mobile Application

1. Upload the *Mobile Application* files in the desired location on your web server.
2. Run the *Mobile Application* script by accessing `index.htm` in a web browser.

Note: The *Mobile Application* must be run on a web server. The script cannot be running via `file://` due to a cross-domain *XMLHttpRequest* restriction in web browsers. The application is optimized for *Google Chrome*[2] (version 23) and runs in modern WebKit-based web browsers like *Apple's Safari Mobile*.

7.3 Contents of the CD-ROM

The following files are included on the CD-ROM:

Zusfsg.pdf
German version of the abstract of this thesis.

[1] `http://maven.apache.org`
[2] `http://www.google.com/chrome`

Abstract.pdf
English version of the abstract of this thesis.

Masterarbeit.pdf
Copy of this thesis.

Crawler.zip
The *Crawler* application described in this thesis.

Mobile_Application.zip
The *Mobile Application* described in this thesis.

7.4 Questionnaire

This section includes the cover sheet with instructions and the whole questionnaire of the usability test in German.

Umfrage zur Benutzung des Systems

Liebe/r Teilnehmer/in,

der Fragebogen ist Teil unserer wissenschaftlichen Untersuchung im Rahmen einer Masterarbeit am *Institut für Informatik* der *Universität Zürich* in Zusammenarbeit mit dem *UniversitätsSpital Zürich*.

Das Ziel dieser Befragung ist die Messung des Nutzens durch das Systems bei der Patientendokumentation. Ausserdem soll die Untersuchung zeigen, wie Benutzer mit dem System in einer klinischen Situation zurechtkommen. Ihre Antworten sind wichtig für uns, um von medizinischen Fachpersonen ein umfassendes Verständnis zur Nützlichkeit und Benutzerfreundlichkeit des Systems zu bekommen.

Die Beantwortung des Fragebogens dauert ca. 7 Minuten. Achten Sie bitte darauf, die Antworten ehrlich und spontan zu wählen. Die Befragung ist selbstverständlich vertraulich, freiwillig und anonym.

Wenn Sie Interesse an den Ergebnissen unserer Untersuchung haben, können Sie Ihre Kontaktdaten in das untenstehende Feld eintragen.

Vielen Dank für Ihre wertvolle Unterstützung.

❑ Ich möchte über die Ergebnisse informiert werden.

E-Mail-Adresse _____

Umfrage zur Benutzung des Systems

A. Persönliche Angaben

1. Ihr Geschlecht?

◯ männlich ◯ weiblich

2. Wie alt sind Sie?

_____ Jahre

3. Sie sind im Spital zur Zeit tätig als:

◯ Unterassistent/in ◯ Leitende/r Arzt/Ärztin

◯ Assistenzarzt/-ärztin ◯ Chefarzt/-ärztin

◯ Oberarzt/-ärztin ◯ _____

4. Besitzen Sie ein Smartphone mit Touchscreen? (*iPhone*, *Android Smartphone*, etc.)

◯ ja ◯ nein ◯ keine Angabe

5. Besitzen Sie einen Tablet-Computer mit Touchscreen? (*iPad*, *Android Tablet*, etc.)

◯ ja ◯ nein ◯ keine Angabe

B. Nützlichkeit

1. Mit dem System kann ich Aufgaben bei meiner Arbeit schneller erledigen.

◯	◯	◯	◯	◯
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

2. Mit dem System wird meine Arbeit angenehmer.

◯	◯	◯	◯	◯
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

3. Das System lässt mir Freiheiten bei der Benutzung.

◯	◯	◯	◯	◯
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

4. Ich finde das System bei meiner Arbeit nützlich.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

C. Benutzerfreundlichkeit

1. Ich denke, dass ich das System häufig verwenden würde.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

2. Ich empfinde das System als unnötig kompliziert.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

3. Ich finde das System einfach zu benutzen.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

4. Ich denke, dass ich die Unterstützung einer fachkundigen Person benötige, um das System verwenden zu können.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

5. Die verschiedenen Funktionen erscheinen mir gut in das System integriert.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

6. Für mich wirkt das System zu inkonsistent.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

7. Ich kann mir vorstellen, dass die meisten Leute die Benutzung dieses Systems schnell erlernen können.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

8. Ich empfinde die Verwendung des Systems als sehr umständlich.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

9. Ich fühle mich bei der Benutzung des Systems sehr sicher.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

10. Ich muss viel lernen, bevor ich mit der Verwendung des Systems anfangen kann.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

D. Einsatzbereich

1. Ich würde das System *während* der Patientenkonsultation einsetzen.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

2. Ich würde das System *nach* der Patientenkonsultation einsetzen.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

3. Ich bevorzuge die Benutzung des Systems auf einem herkömmlichen Computer mit Maus.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

E. Allgemeine Zufriedenheit

1. Ich finde die grafische Benutzeroberfläche ansprechend.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

2. Ich bin mit dem System zufrieden.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

3. Ich würde das System einem Freund weiter empfehlen.

○	○	○	○	○
stimme überhaupt nicht zu	stimme eher nicht zu	neutral	stimme eher zu	stimme voll zu

F. Offene Fragen

1. Was müsste erfüllt sein, damit ich das System benutze?

_____

_____

_____

_____

_____

_____

2. Nennen Sie bitte die *positivsten* Aspekte.

_____

_____

_____

_____

_____

_____

3. Nennen Sie bitte die *negativsten* Aspekte.

_____

_____

_____

_____

_____

_____

Vielen Dank für Ihre Unterstützung!

7.5 Results of the Evaluation

The table on page 59 summarizes the results of the evaluation.

7.6 Example excerpt of a medical record

```
# Demographics
First Name: Ellen
Last Name: Ross
Gender: Female
Language Spoken: English
Birthday: March 7, 1960

# Allergies
Allergy Name: Penicillin
Reaction: Hives
Severity: Moderate to severe

# Immunizations
Date: May 2001
Immunization Name: Influenza virus vaccine, IM
Type: Intramuscular injection
Dose Quantity (value / unit): 50 / mcg
Education/Instructions: Possible flu-like symptoms for three days

# Medication
Date: December 10, 2003
Type: Tablet
Name of Medication: Indomethacin
Instructions: 50mg bid with food
Dose Quantity (value / unit): 50 / mg
Rate Quantity (value / unit): 2 / day
Name of Prescriber: Ashby Medical Center

# Problem List
Observation: Ankle Sprain
Status: Active
Date: March 28, 2005
Comments: Slipped on ice and fell.

# Procedures
Procedure: Laparoscopic Cholecystectomy
Date: September 28, 2002
Provider: Dr. Bala Venktaraman
Location: Ashby Medical Center
```

Listing 7.1: Example excerpt of a medical record [blu12]

	Personal information					Perceived usefulness				Perceived ease of use (SUS)										Field of application			Satisfaction			
#	A1	A2	A3	A4	A5	B1	B2	B3	B4	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	D1	D2	D3	E1	E2	E3	SUS
1	m	30	AA	1	0	4	3	4	4	5	1	4	1	5	1	3	1	4	1	5	2	1	4	4	4	90.00
2	m	31	AA	1	0	4	4	3	5	4	2	4	1	4	2	5	1	4	4	4	2	4	5	4	4	77.50
3	w	27	AA	1	0	5	4	4	5	5	1	5	2	5	3	5	2	4	2	3	5	4	5	4	4	85.00
4	w	30	AA	1	0	3	3	4	3	4	2	4	1	3	2	4	2	4	1	4	2	3	4	4	4	77.50
5	w	42	AA	0	1	5	4	5	4	4	2	5	1	3	2	5	2	4	2	4	2	3	4	4	4	80.00
6	m	29	AA	1	0	4	4	3	3	5	2	4	3	5	2	5	3	3	1	5	2	3	4	4	4	77.50
7	w	34	AA	1	0	3	3	4	3	3	2	4	4	4	2	4	3	3	2	4	1	4	4	3	4	62.50
8	m	32	AA	1	1	4	3	3	3	4	3	4	2	4	2	5	2	3	2	4	3	1	2	2	3	72.50
9	w	30	AA	1	0	4	4	2	4	4	2	4	1	4	3	5	2	4	1	4	2	2	3	4	4	80.00
	Average	31.67		0.89	0.22	4.00	3.56	3.56	3.78	4.22	1.89	4.22	1.78	4.11	2.11	4.56	2.00	3.67	1.78	4.11	2.33	2.78	3.89	3.67	3.89	78.06
	Median	30.00		1.00	0.00	4.00	4.00	4.00	4.00	4.00	2.00	4.00	1.00	4.00	2.00	5.00	2.00	4.00	2.00	4.00	2.00	3.00	4.00	4.00	4.00	77.50
	Lowest Value	27.00		0.00	0.00	3.00	3.00	2.00	3.00	3.00	1.00	4.00	1.00	3.00	1.00	3.00	1.00	3.00	1.00	3.00	1.00	1.00	2.00	2.00	3.00	62.50
	Highest Value	42.00		1.00	1.00	5.00	4.00	5.00	5.00	5.00	3.00	5.00	4.00	5.00	3.00	5.00	3.00	4.00	4.00	5.00	5.00	4.00	5.00	4.00	4.00	90.00
	Standard Deviation	4.33		0.33	0.44	0.71	0.53	0.88	0.83	0.67	0.60	0.44	1.09	0.78	0.60	0.73	0.71	0.50	0.97	0.60	1.12	1.20	0.93	0.71	0.33	7.68

Bibliography

[aja12] AJAX Magazine: JavaScriptMVC Framework. http://ajax.phpmagazine.net/2009/06/javascriptmvc_framework.html, last checked: 20.12.2012. Cited on page 20.

[B+12] Kent Beck et al. Manifesto for Agile Software Development. http://agilemanifesto.org, last checked: 03.12.2012. Cited on page 17.

[BD09] Niels Ole Bernsen and Laila Dybkjær. *Multimodal Usability*. Human-Computer Interaction Series. Springer, London, 2009. Cited on pages 7 and 17.

[blu12] Health Design Challenge: d+collab // THE PATIENT RECORD. http://blue-button.github.com/challenge, last checked: 05.12.2012. Cited on pages x and 58.

[Bro96] John Brooke. Usability Evaluation in Industry. chapter SUS: a "quick and dirty" usability scale. Taylor & Francis Group, London, 1996. Cited on pages 39 and 40.

[CFFH05] Hsinchun Chen, Sherrilynne S. Fuller, Carol Friedman, and William Hersh, editors. *Medical Informatics: Knowledge Management and Data Mining in Biomedicine*. Springer, New York, 1st edition, 2005. Cited on pages 6 and 9.

[CL09] Jake Y. Chen and Stefano Lonardi, editors. *Biological Data Mining (Chapman & Hall/CRC Data Mining and Knowledge Discovery Series)*. Chapman and Hall/CRC, Boca Raton, USA, 1st edition, 2009. Cited on page 23.

[Dav89] Fred D. Davis. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Q.*, 13(3):319–340, September 1989. Cited on page 39.

[dia12] DiagnosisPro - The Ultimate Differential Diagnosis Reminder Tool. http://www.diagnosispro.com, last checked: 05.11.2012. Cited on page 12.

[dic12] dict.md - medical dictionary. http://www.dict.md, last checked: 05.11.2012. Cited on page 12.

[dis12] Diseases Database - Medical lists and links Diseases Database. http://www.diseasesdatabase.com, last checked: 05.11.2012. Cited on pages 12 and 23.

[Fad09] Dmitry Fadeyev. The Smashing Book. chapter User Interface in Modern Web Applications. Smashing Media GmbH, Germany, 2009. Cited on pages 7 and 8.

[FBK+08] Anthony S. Fauci, Eugene Braunwald, Dennis L. Kasper, Stephen L. Hauser, Dan L. Longo, J. Larry Jameson, and Joseph Loscalzo. *Harrison's Principles of Internal Medicine*. McGraw-Hill Professional, New York, 17th edition, 2008. Cited on page 9.

[Fen07] David D. Feng, editor. *Biomedical Information Technology*. Academic Press, Burlington, MA, 1st edition, 2007. Cited on page 3.

[Fir10] Maximiliano Firtman. *Programming the Mobile Web*. O'Reilly Media, Inc., Sebastopol, USA, 1st edition, 2010. Cited on page 26.

[FS04] Susan Fowler and Victor Stanwick. *Web Application Design Handbook: Best Practices for Web-Based Software*. Morgan Kaufmann, San Francisco, USA, 2004. Cited on page 17.

[G+11] Lukas Golder et al. DRG: Befürchtungen einer zunehmenden Bürokratisierung der Medizin. Auszug aus der Forschungsarbeit von gfs.bern, 2011. Cited on pages iii, v, and 1.

[GCD03] Rajdeep Grewal, Thomas W. Cline, and Anthony Davies. Early-entrant advantage, word-of-mouth communication, brand similarity, and the consumer decision-making process. *Journal of Consumer Psychology*, 13(3):187–197, 2003. Cited on page 43.

[GGS+10] Dimosthenis Georgiadis, Panagiotis Germanakos, George Samaras, Constantinos Mourlas, and Eleni Christodoulou. An Intelligent Web-Based Healthcare System: The Case of DYMOS. In *Web-Based Applications in Healthcare and Biomedicine*, pages 19–46. 2010. Cited on page 17.

[Hal12] Wesley Hales. *HTML5 and JavaScript Web Apps*. O'Reilly Media, Inc., Sebastopol, USA, 1st edition, 2012. Cited on pages 20, 26, and 27.

[HB11] Steven Hoober and Eric Berkman. *Designing Mobile Interfaces*. O'Reilly Media, Inc., Sebastopol, USA, 1st edition, 2011. Cited on pages 27 and 28.

[Hea99] Marti A. Hearst. Untangling text data mining. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, pages 3–10, Stroudsburg, PA, USA, 1999. Association for Computational Linguistics. Cited on page 9.

[hea12] Healthline - Medical Information & Trusted Health Advice. `http://www.healthline.com`, last checked: 05.11.2012. Cited on pages 14 and 23.

[Joh10] Jeff Johnson. *Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Rules*. Morgan Kaufmann. Elsevier Science, USA, 2010. Cited on page 7.

[Kra12] Nicole Kraettli. Ein goldener Chip und viele rote Köpfe. *Beobachter*, (24):24–25, November 2012. Cited on page 4.

[Kru05] Steve Krug. *Don't Make Me Think: A Common Sense Approach to the Web*. New Riders Publishing, Thousand Oaks, CA, USA, 2nd edition, 2005. Cited on page 17.

[Kru09] Steve Krug. *Rocket Surgery Made Easy: The Do-It-Yourself Guide to Finding and Fixing Usability Problems*. New Riders Press, Berkeley, CA, 1st edition, 2009. Cited on pages 17 and 37.

[LBSAS09] Tao Le, Vikas Bhushan, Mae Sheikh-Ali, and Fadi Abu Shahin. *First Aid for the USMLE Step 2 CS*. McGraw-Hill Medical, New York, 3rd edition, 2009. Cited on page 36.

[Lew82] Clayton H. Lewis. Using the "Thinking Aloud" Method In Cognitive Interface Design. Technical Report RC-9265, IBM, 1982. Cited on page 17.

[Lew95] James R. Lewis. IBM computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *Int. J. Hum.-Comput. Interact.*, 7(1):57–78, January 1995. Cited on page 39.

[LS09] James R. Lewis and Jeff Sauro. The Factor Structure of the System Usability Scale. In *Proceedings of the 1st International Conference on Human Centered Design: Held as Part of HCI International 2009*, HCD 09, pages 94–103, Berlin, Heidelberg, 2009. Springer-Verlag. Cited on page 40.

[Lun01] Arnold M. Lund. Measuring Usability with the USE Questionnaire. *STC Usability SIG Newsletter*, 8(2), 2001. Cited on page 43.

[LWTC07] Murray Longmore, Ian Wilkinson, Tom Turmezei, and Chee Kay Cheung. *Oxford Handbook of Clinical Medicine*. Oxford University Press, New York, 7th edition, 2007. Cited on page 9.

[may12] Mayo Clinic. `http://www.mayoclinic.com`, last checked: 05.11.2012. Cited on page 23.

[med12a] MedicineNet - Health and Medical Information Produced by Doctors. `http://www.medicinenet.com`, last checked: 05.11.2012. Cited on page 23.

[med12b] Medscape: Medical News, Full-text Journal Articles & More. `http://www.medscape.com`, last checked: 05.11.2012. Cited on page 23.

[Ozo08] A. Ant Ozok. The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications. Human Factors & Ergonomics, chapter Survey Design and Implementation in HCI. Taylor & Francis Group, London, 2008. Cited on pages 35 and 37.

[PCL+12] Bhakti K. Patel, Christopher G. Chapman, Nancy Luo, James N. Woodruff, and Vineet M. Arora. Impact of mobile tablet computers on internal medicine resident efficiency. *Archives of Internal Medicine*, 172(5):436–438, 2012. Cited on page 1.

[Pro10] *Professional Guide to Signs and Symptoms*. Lippincott Williams & Wilkins, sixth edition, 2010. Cited on pages 10 and 25.

[Rei12] Wolfgang Reinhardt. Crowdsourcing the translation of SUS. `http://isitjustme.de/2012/01/crowdsourcing-the-translation-of-sus`, last checked: 28.11.2012. Cited on page 40.

[Sau12] Jeff Sauro. Measuring Usability With The System Usability Scale (SUS). `http://www.measuringusability.com/sus.php`, last checked: 28.11.2012. Cited on pages 39, 40, 43, and 45.

[SET09] Toby Segaran, Colin Evans, and Jamie Taylor. *Programming the Semantic Web*. O'Reilly Media, Inc., Sebastopol, USA, 1st edition, 2009. Cited on pages 4 and 6.

[Sta10] Jonathan Stark. *Building iPhone Apps with HTML, CSS, and JavaScript - Making App Store Apps Without Objective-C or Cocoa*. O'Reilly Media, Inc., 2010. Cited on page 26.

[SW09] Frank Sullivan and Jeremy C. Wyatt. *ABC of Health Informatics*. ABC Series. Wiley, Massachusetts, USA, 2009. Cited on page 3.

[sym12] Symptom Ontology Wiki. `http://symptomontologywiki.igs.umaryland.edu/wiki/index.php/Main_Page`, last checked: 19.12.2012. Cited on page 12.

[Tay06] Paul Taylor. *From Patient Data to Medical Knowledge: The Principles and Practice of Health Informatics*. BMJ Books, London, 1st edition, 2006. Cited on page 3.

[Tid05] Jenifer Tidwell. *Designing Interfaces: Patterns for Effective Interaction Design*. O'Reilly Media, Inc., Sebastopol, USA, 1st edition, 2005. Cited on page 7.

[w3c12a] Indexed Database API. `http://www.w3.org/TR/IndexedDB`, last checked: 05.12.2012. Cited on page 27.

[w3c12b] Web SQL Database. `http://www.w3.org/TR/webdatabase`, last checked: 05.12.2012. Cited on page 27.

[web12] WebMD - Better information. Better health. `http://www.webmd.com`, last checked: 05.11.2012. Cited on page 23.

[Wei11] Susan M. Weinschenk. *100 Things Every Designer Needs to Know About People*. New Riders, Berkeley, USA, 2011. Cited on page 17.

[wik12a] User story - Wikipedia, the free encyclopedia. `http://en.wikipedia.org/wiki/User_stories`, last checked: 03.12.2012. Cited on page 17.

[wik12b] Wikipedia, the free encyclopedia. `http://www.wikipedia.org`, last checked: 05.11.2012. Cited on pages 13 and 23.

[wik12c] JavaScriptMVC - Wikipedia, the free encyclopedia. `http://en.wikipedia.org/wiki/JavaScriptMVC`, last checked: 12.12.2012. Cited on page 21.

[wik12d] Validity (statistics) - Wikipedia, the free encyclopedia. `http://en.wikipedia.org/wiki/Validity_(statistics)`, last checked: 20.12.2012. Cited on page 44.

[WRDG10] Michael Würsch, Gerald Reif, Serge Demeyer, and Harald C. Gall. Fostering synergies - how semantic web technology could influence software repositories. In *2nd International Workshop on Search-driven Development: Users, Infrastructure, Tools and Evaluation*, pages 45–48, May 2010. Cited on page 6.

[Wür12] Michael Würsch. *A Query Framework for Software Evolution Data*. PhD thesis, 2012. Cited on page 5.