FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Requirementsanalyse und Prototyp einer klinischen Plattform zur aktiven Patientenpartizipation

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Magister der Sozial- und Wirtschaftswissenschaften

im Rahmen des Studiums

Masterstudium Informatikmanagement

eingereicht von

Matthias Eder, BSc
Matrikelnummer 0300272

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer: Ao.Univ.-Prof. Mag.rer.nat. Dipl.-Ing.Dr.techn. Rudolf Freund

Wien, 22.10.2011

(Unterschrift Verfasser) (Unterschrift Betreuer)

Erklärung zur Verfassung der Arbeit

Matthias Eder
Erne-Seder-Gasse 6/3/12, 1030 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 22.10.2011

Zusammenfassung

Eine Vielzahl an universitären und wirtschaftlichen Organisationen entwickeln patientengeführte elektronische Gesundheitsakten (Personal Health Records - PHR) und stellen sie bereits teilweise der breiten Öffentlichkeit zur Verfügung. Neben der zentralen Speicherung und einer optimierten Arzt-Patienten Kommunikation ergeben sich durch den Einsatz von PHRs zusätzliche Möglichkeiten, die medizinische Behandlung zu verbessern. Eine davon sind elektronische Tagebücher, die den Patienten das Protokollieren ihres Gesundheitszustands abseits medizinischer Einrichtungen erlauben. Diese Arbeit umfasst die Analyse und Umsetzung einer PHR-Anwendung zur Definition und Konsumation generischer Patiententagebücher.

Aufbauend auf einer Analyse der etablierten Standards zur Speicherung und Kommunikation medizinischer Daten (openEHR, HL7 CDA, CCR) und aktueller Projekte, die diese Standards einsetzen, werden mögliche Ansätze der Systemarchitektur sowie des Datenmodells erarbeitet. Dabei liegt der Fokus auf frei definierbare, erweiterbare und wiederverwendbare Tagebuchstrukturen sowie auf einer möglichen Interoperabilität mit externen Systemen. Im UI-Design werden speziell die Aspekte des komplexen Tagebuchaufbaus behandelt und mögliche Benutzerschnittstellen aufgezeigt, die sowohl das Entwerfen von Tagebüchern als auch deren Verwendung im Kontext der Anwendungsarchitektur erlaubt.

Die prototypische Implementierung der Anwendung erfolgt als Web-basierte online Applikation im Sinne des Web 2.0 in ASP.NET MVC 3 unter Verwendung mehrerer Frameworks zur Unterstützung von Best Practices wie Dependency Injection und OR-Mapping. Mit dem Einsatz der Rendering-Engine Razor sowie externer Javascript Funktionsbibliotheken werden die Anforderungen an das User-Interface umgesetzt. Der Prototyp umfasst ein administratives Modul zum Bearbeiten und Zuweisen der Tagebücher und ein Patientenmodul, um diese regelmäßig auszufüllen.

Abstract

A number of academic and economic organizations started to develop patient-centric electronic health records (Personal Health Records - PHR) and partly made those available to the public. Besides central data storage and optimized physician-patient-communication, the use of PHRs provides additional options in order to enhance medical treatment. One of those options are electronic diaries, allowing patients to protocol their health status independently outside a medical facility. This thesis covers the analysis as well as the development of a PHR-application to define and use generic patient diaries.

Based on the analysis of well-known standards to store and communicate medical data (openEHR, HL7 CDA, CCR) and current projects using these standards, possible approaches of the system architecture and the data model were developed. This process focused on freely definable, expandable and reusable diary structures as well as interoperability with external systems. Furthermore, the UI-design deals with the aspects of the complex diary format and provides interfaces, allowing the design and the usage of the diaries in the context of the applications architecture.

The prototypical implementation was done as a web-based online application using ASP.NET MVC 3 and different frameworks to support best practices like dependency injection and OR-mapping. The usage of the rendering engine Razor and open-source Javascript libraries allowed the implementation of the user-interface requirements. The prototype was divided into an administrative module to edit and assign diaries and a patient module for periodic use.

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Allgemeines

Mit dem Einzug der elektronischen Datenverarbeitung im medizinischen Sektor in den letzten Jahrzehnten sind unzählige neue Möglichkeiten zur Verbesserung der medizinischen Versorgung entstanden. Von Arztpraxissystemen bis hin zu Krankenhausinformationssystemen (KIS) wurde vor allem im Verwaltungssektor viel investiert. Letztere sind aber auch zur Verbesserung des medizinischen Bereichs geeignet. Ein KIS ist ein Zusammenschluss mehrerer Komponenten wie z.B. Betriebs- und Managementsunterstützende Systeme[1]. Hierzu zählen die Patientdatenverwaltung, das Labor- und Radiologieinformationssystem, das Operationsdokumentation, ein Apothekensystem oder Materialwirtschafts- und Personalinformationssysteme. Zur Verbesserung des medizinischen Bereichs tragen vor allem die Elektronische Krankenakte (siehe Kapitel 2.1), die Integration von Literatur sowie Entscheidungsunterstützende Funktionen[2] bei.

Die Entwicklung in der Gesundheitsversorgung blieb aber nicht bei der lokalen Digitalisierung vorhandener Konzepte stehen. Vor allem die flächendeckende Verbreitung des Internets und der Einzug in unser tägliches Leben trug zu neuen Ideen und Konzepten bei. Annähernd fünf Prozent der Suchanfragen bei Google sind gesundheitsbezogen[3]. Eigene Foren oder Weblogs erlauben Patienten sowie medizinischem Fachpersonal ihre Anliegen

und Erfahrungen zu veröffentlichen. Fachartikel sind sofort und meistens kostenlos einsehbar. Diese und weitere Entwicklungen trugen zur Entstehung des Begriffs „eHealth" als Verwendung von Internet-Technologien im Gesundheitsbereich bei[4].

Mit der Evolution des Internets formte sich im Jahr 2004 die Terminologie „Web 2.0", die dadurch gekennzeichnet ist, dass aus dem unidirektionalen Massenmedium Internet ein Medium entstanden ist, das seinen Benutzern die aktive Partizipation, also das eigenständige Generieren von Inhalten ohne speziellem Know-How erlaubt. In diesem Zusammenhang versteht man mittlerweile den Einsatz von Web 2.0-Technologieen im medizinischen Bereich als „Health 2.0" oder „Medicine 2.0"[5].

1.2 Aufbau der Arbeit

Diese Arbeit gibt zuerst einen einleitenden Überblick über zwei Kernkonzepte von eHealth und Health 2.0, nämlich den Electronic Health Records (EHR) sowie den Personal Health Records (PHR) (siehe 2.1 und 2.2). In diesem Zusammenhang werden im Kapitel 2.5 anschließend aktuell etablierte Standards zur Kommunikation medizinischer Daten wie openEHR, Health Level 7 oder Continuity of Care Record analysiert und verglichen.

Der praktische Teil der Arbeit konzentriert sich auf die Analyse und Umsetzung einer Teilfunktionalität eines PHR, einem elektronischen Patiententagebuch als Internetportal unter Verwendung der oben erwähnten Standards. Kapitel 4 untersucht die verschiedenen Möglichkeiten für den Entwurf dieses Systems. Dabei werden folgende Schwerpunkte gesetzt:

- Verschiedene Ansätze zur Systemarchitektur, die sowohl medizinische Einrichtungen als auch Drittanbieter als Betreiber in Erwägung ziehen. Kapitel 4.2 beschreibt die verschiedene Plattformen auf denen das System aufgesetzt werden kann sowie die Unterschiede, die sich durch verschiedene Betreiber ergeben.
- Möglichkeiten, Patienten-spezifische aber auch Krankheits-spezifische online-Tagebücher zu erzeugen und bestimmten Patienten bereitzustel-

len. Im Abschnitt 4.2.2 werden mögliche Datenmodelle analysiert, die diese Anforderung unterstützen.

- Sowohl Produzenten als auch Konsumenten dieser Tagebücher müssen über eine geeignete Benutzerschnittstelle verfügen, die intuitiv alle Möglichkeiten bereitstellt. Verschiedene Ansätze die sowohl die Patientenseite als auch das User-Interface des medizinischen Personals betreffen, finden sich im Abschnitt 4.4.

- Schnittstellen zu anderen Systemen oder medizinischen Geräten die die Handhabung erleichtern und somit die Akzeptanz erhöhen. Kapitel 4.5 beschreibt die Implementierung von Webservices zur Verknüpfung verschiedener Systeme als auch die Implementierung von Hardware-Schnittstellen.

- Die gewählte Architektur beeinflusst in gewissem Maße auch den Datenschutz, der bei sensiblen personenbezogenen Daten eine große Rolle spielt. Im Abschnitt 4.3 werden dazu die rechtlichen Aspekte behandelt.

1.3 Motivation

Auch wenn es bereits verschiedene Ansätze und auch Produkte im Bereich der Personal Health Records (PHR) gibt (siehe Kapitel 2.6.2), erfuhren diese Systeme noch keine wirkliche Evolution. Trotz großer potentieller Vorteile, wie der Zugang zu den eigenen medizinischen Daten oder einer einfachen und zielgerichteten Kommunikation, fehlt es vielen Anwendungen nach wie vor an der notwendigen Akzeptanz der Benutzer. Der wichtigste Aspekt dabei ist der Zugriff auf die eigenen Gesundheitsdaten und damit der bessere Überblick über den medizinischen Zustand. Durch die integrierten Kommunikationsmechanismen kann das Arzt-Patienten-Verhältnis gestärkt werden und unkompliziert Termine vereinbart, Fragen gestellt und beantwortet oder auch Rezepte ausgestellt bzw. erneuert werden. Auf diese Weise kann auch die Überwachung eines medizinischen Zustands von einer episodischen hin zu einer kontinuierlichen Betreuung gebracht werden, was zu einer schnelleren Reaktionszeit auf bestimmte Situationen führt. [6]

Die Entwicklung von PHR-Systemen passiert zum Großteil in den USA. Auch wenn in Europa viele Regierungen in Richtung einer Einführung eines Electronic Health Records (EHR) gehen, wird das Ergebnis eher eine lebenslangen Patientenakte sein und den Fokus in erster Linie weniger auf die Interaktion und die Partizipation des Patienten legen.

Der in dieser Arbeit analysierte Entwurf und die Implementierung eines elektronischen Patiententagebuchs soll einen weiteren Ansatz zur Umsetzung eines Teils eines PHRs liefern. Die Architektur und das Datenmodell bzw. die Datenkommunikation sind von einigen großen Projekten bereits erfolgreich durchgeführt worden oder sogar als Standard verfügbar. Die Verwendung dieser bekannten Vorlagen in elektronischen Tagebüchern ist zum aktuellen Zeitpunkt eher oberflächlich. Entweder sind die Tagebücher als einfache und kontextlose Eigenschaft-Werte-Paare realisiert oder strikt auf eine oder mehrere vorgegebene Krankheiten bzw. Geräte limitiert[7]. An diesem Punkt liegt das Potenzial, durch die Verwendung eines etablierten Standards ein System zu schaffen, das für jede erdenkliche Situation ein Tagebuch erstellen kann und sich unkompliziert in bestehende Anwendungen integrieren lässt.

Das im Rahmen dieser Arbeit zu entwickelnde System verfolgt diesen Ansatz, indem für das Grundgerüst des Datenmodells einer der Standards herangezogen wird, der es erlaubt, Tagebücher sowie deren Daten vollkommen frei abzubilden. Darauf aufbauend soll eine Online-Applikation entworfen und implementiert werden, die die Modellierung und Verwendung der Tagebücher ermöglicht und durch den Einsatz von Best Practices bei der Entwicklung Schnittstellen zur Kommunikation mit anderen Systemen bereitstellt und Möglichkeiten für Erweiterungen bietet. Die genaue Spezifikation der Anforderungen findet sich im Kapitel 4.1.

Kapitel 2

Grundlagen der elektronischen medizinischen Datenhaltung

2.1 Electronic Health Records (EHR)

Die elektronische Krankenakte bzw. der Electronic Health Record ist seit vielen Jahren ein zentraler Punkt der Forschungen im Bereich der medizinischen Informatik[8]. Sie kann als lebenslange, digitale Aufzeichnung von Gesundheitsinformationen eines Individuums definiert werden, mit dem Ziel, die medizinische Versorgung, Ausbildung und Forschung zu verbessern[9].

2.1.1 Die konventionelle Krankenakte

Mit der methodischen Behandlung von Menschen entstand gleichzeitig die Dokumentation des medizinischen Zustandes sowie der angewendeten Verfahren. Mit der Weiterentwicklung der Medizin über Jahrhunderte, entwickelten sich ebenfalls die Methoden der medizinischen Dokumentation. Sie ermöglichten nicht nur die effektivere Behandlung von Patienten sondern führten zu einer immensen Wissensbasis für die medizinische Forschung[1].

Die medizinische Dokumentation zu einem Patienten wird heute als Krankenakte geführt und umfasst alle Dokumente die im Rahmen der medizinischen Versorgung eines Patienten entstehen. Teil dieser Krankenakte sind

unter anderem Anamnese- und Befunddokumentation, zusammenfassende Berichte sowie diverse Übersichten. Diese Teildokumentationen besitzen unterschiedliche Ziele und Eigenschaften. Im Detail beschreiben die Einträge der Krankenakte

- welche medizinische Aktion (Beobachtung, Medikation, Instruktion, etc.)
- zu welchem Zeitpunkt
- von wem
- aus welchem Grund
- mit welchem Ergebnis und Schlussfolgerungen
- an welchem Patienten

durchgeführt wurden. Dies geschieht hinsichtlich vergangener aber auch zukünftiger Ereignisse im Sinne von geplanten Untersuchungen und Behandlungen. [10]

Die konventionelle Krankenakte in Papierform entwickelte sich gemeinsam mit der medizinischen Dokumentation und galt bis zur Einführung von Krankenhausinformationssystemen als Standard. Sie wurde in nahezu allen medizinischen Einrichtungen in unterschiedlicher Komplexität geführt. Dieser Umstand führt jedoch auch dazu, dass die gesammelten Daten dieser Einrichtungen zum Großteil auf einen Ort beschränkt sind. Und selbst innerhalb einer medizinischen Einrichtung, die eine gewisse Größe übersteigt und mehrere unterschiedliche Leistungen anbietet, werden Krankenakten dezentral gewartet und mit komplexen Aktenplänen und Archivierungsverfahren verwaltet. Ein Beispiel aus der Praxis zeigt, dass bei verschiedenen Ambulanten Behandlungen eines Patienten, jedes mal ein neues Dokument angelegt und in die Patientenakte hinzugefügt wird (siehe Abbildung 2.1), was in diesem Fall zu neun verschiedenen Dokumenten führt. [1]

Folglich wird die papierene Akte in Umgebungen, in denen mehrere Leistungsträger bzw. Personen Dokumente beisteuern, über kurz oder lang nicht funktionieren bzw. zu erheblichem zusätzlichen Aufwand führen.

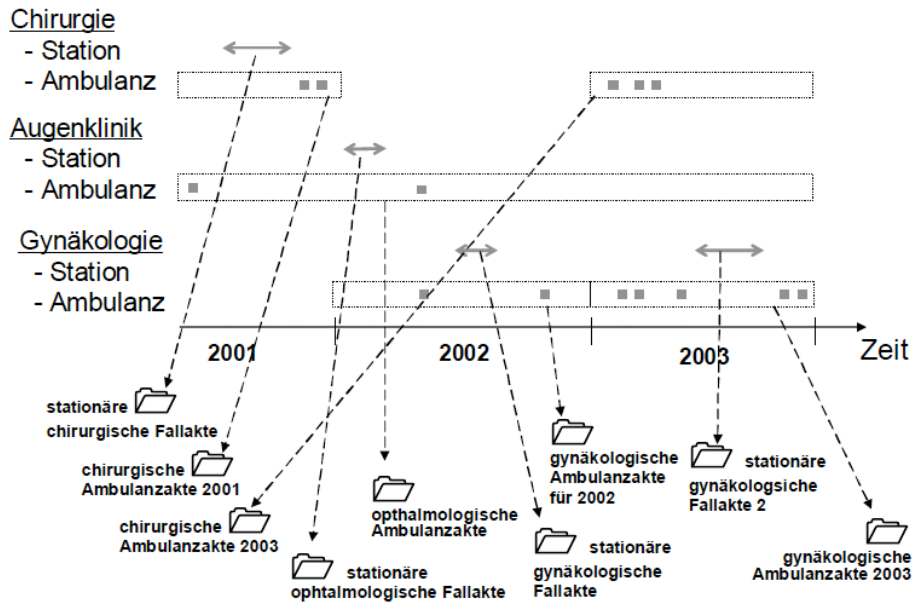


Abbildung 2.1: Neun unterschiedliche Akten einer Patientin in einem Krankenhaus [1].

2.1.2 Die elektronische Krankenakte

Mit der Einführung von medizinischen Informationssystemen hat auch die digitale Version der papierenen Krankenakte, die elektronische Akte, an Relevanz gewonnen. Sie basiert auf den gleichen Prinzipien ihres Vorgängers und ermöglicht dabei den zeitnahen Zugriff auf sämtliche Daten ohne die Einschränkung auf einen bestimmten Ort[11]. So soll den Ärzten und sonstigem betreuenden Personal des Gesundheitswesens der Zugriff auf Informationen ermöglicht werden, die bis dahin auf mehrere Orte verteilt waren, auf Papier oder digital existierten und als narrative, strukturierte, kodierte oder multimediale Einträge zu finden waren[12].

In der Fachliteratur existieren mehrere verschiedene Definitionen der elektronischen Krankenakte, die je nach Inhalt, Erzeuger, Herkunft oder Zugriffsmöglichkeiten verschiedene Begriffe einführen. Die Definition des „Office of Health and the Information Highway Canada" aus dem Jahr 2001 beschreibt aufeinander aufbauende Einheiten, deren letzte Stufe den EHR darstellt. Diese Einheiten sind:

Incident Record Daten, die erzeugt werden, wenn ein Individuum mit jemandem aus dem Gesundheitsbereich interagiert.

Patient Record Eine Sammlung von „Incident Records" zu einem bestimmten Patienten, die von einer Einrichtung des Gesundheitswesens geführt wird.

Health Record Alle „Patient Records" eines Patienten aus allen Einrichtungen, die solch einen „Patient Record" führen.

Werden diese Einheiten elektronisch geführt, ergeben sich somit der „Electronic Incident Record", der „Electronic Patient Record" sowie schlussendlich der „Electronic Health Record". [13]

Eine weitere Definition von Waegemann (1999), die stufenweise zum EHR führt, orientiert sich eher an der natürlichen Entwicklung des IT-Einsatzes in medizinischen Einrichtungen[1]. Wiederum bauen die Stufen aufeinander auf. Im Detail sind diese:

Automated Record In dieser ersten Stufe werden hauptsächlich Patienteninformationen wie Stamm- und Falldaten gespeichert, um die administrativen Einrichtungen zu unterstützen. Ergänzt wird der „Automated Record" durch Information zu den Ablageorten der papierbasierten Akte.

Computerized Medical Record System Bei solch einem System werden die Akten zwar teilweise bzw. vollständig elektronisch geführt, existieren jedoch nur begleitend zur herkömmlichen papierene Akte.

Electronic Medical Record System Ein EMR beschreibt in diesem Kontext ein Informationssystem, dass anstelle der papierenen Krankenakte eine elektronische medizinische Dokumentation ermöglicht.

Electronic Patient Record System Die vierte Stufe ergibt die Vernetzung mehrerer EMRs verschiedener medizinischer Einrichtungen zu einer gesammelten Krankenakte.

Electronic Health Record Der EHR bezeichnet in diesem Konzept die höchste Stufe, die erreicht wird, wenn eine vollständige und lebenslange medizinische Dokumentation von Geburt an ermöglicht wird. [14]

Eine essentielle Eigenschaft jeder elektronischen Krankenakte ist die Informationstransparenz, die Durch die strukturierte Erfassung des medizinischen Zustandes sowie der Behandlungsverfahren erreicht wird. Dadurch ergeben sich unterschiedlich visualisierte oder strukturierte Sichten auf die Daten, um der medizinischen Einrichtung einen raschen und effizienten Überblick über die Eigenschaften eines oder mehrerer Patienten zu verschaffen[15]. Unter anderem hilft dies bei der Wahl strategischer, medizinischer Entscheidungen. Ein Beispiel für eine speziell aufbereitete Sicht auf die elektronische Krankenakte ist das problemorientierte Krankenblatt [16]. Es organisiert die Einträge der Akte nicht wie üblich nach ihrer Art bzw. Herkunft oder ihrem Eingangsdatum, sondern nach dem spezifischen Problem des Patienten. Andere Darstellungen versuchen Informationszusammenhänge aus mehreren Patientenakten hervorzuheben. Wang et al. zeigen im Projekt „Lifelines2" ein Interface, mit dem zeitlich relevante Daten, wie eben jene aus Patientenakten, durch geeignete Filter und Anordnungen so auf Zeitleisten dargestellt werden können, dass etwaige Relationen zwischen dokumentierten Ereignissen aus diesen Akten erkannt werden können[17].

Ein weiterer Aspekt der elektronischen Patientenakte ist die Tatsache, dass Informationen unmittelbar und eventuell auch gleichzeitig an alle abrufenden Stellen weitergegeben werden. Was bei der konventionellen Krankenakte vom aktuellen Ort der Akte abhängig war, ist jetzt sofort verfügbar. Davon abgesehen, dass die verschiedenen Stellen nicht mehr auf die lokale Präsenz der benötigten Information warten müssen, erlaubt die Automatisierung zusätzliche unterstützende Funktionen wie Benachrichtigungs- und Erinnerungsfunktionen (siehe auch Kapitel 6.1.1) oder auch entscheidungsunterstützende Funktionen[18].

Die Digitalisierung der Informationen ermöglicht außerdem die Integration von klinischen Pfaden in die Umgebung der elektronischen Krankenakten. Diese Pfade beschreiben von der klinischen Einrichtung vordefinierte Handlungsabläufe bezogen auf bestimmte Probleme. Je nach Patient und Problem können diese klinischen Pfade individuell erweitert oder eingeschränkt werden und sollen die behandelnden Abteilungen bzw. Personen bei der Planung unterstützen [19]. Gerade die Kombination mit den oben erwähnten Benachrichtigungs- und Erinnerungsfunktionen stellt ein großes Potential zur verbesserten Behandlung von Patienten dar. Obwohl die klinischen Pfa-

de dieses Potential in Aussicht stellen, ist es auf Grund der Komplexität sehr schwierig, sich innerhalb einer medizinischen Einrichtung auf einheitliche Vorgehensweisen zu einigen.

Allgemein gesprochen soll die Einführung und der Einsatz eines EHR (1) die Situation eines Patienten auf Basis der Diagnosen und Therapien transparenter machen, (2) eine Problem- und Ergebnisorientierte Behandlung ermöglichen und (3) interdisziplinäres Arbeiten unterstützen und fördern. Laut Haas könnte diese Technologie einen ähnlichen Beitrag zum Gesundheitsbereich bringen wie die Medizin-Technik. Gleichzeitig warnt er jedoch auch vor der Realisierung des gläsernen Patienten und Arztes, was die Notwendigkeit nach einem besonders verantwortungsvollen Umgang mit dieser Technologie deutlich macht[1].

2.1.3 Datenschutz & Rechtliches

Laut § 4 Ziffer 2 des Datenschutzgesetzes 2000 (DSG 2000) zählen Daten über Gesundheit zu den „sensiblen Daten" oder „besonders schutzwürdigen Daten". Somit sind gesundheitsspezifische Daten einer Person gleich schützenswert wie ihre rassische und ethnische Herkunft, politische Meinung, Gewerkschaftszugehörigkeit, religiöse oder philosophische Überzeugung oder ihr Sexualleben. Sollten die oben genannten Daten in unbefugte Hände fallen, könnte das zu fatalen Folgen im beruflichen und sozialen Umfeld führen. Gerade im Kontext der vernetzten medizinischen Infrastruktur durch die Einführung von elektronischen Krankenakten ist der Schutz dieser Daten von höchster Priorität.

Im Zusammenhang mit der elektronischen Krankenakte ist der bestehende rechtliche Rahmen jedoch problematisch. Im § 9 des DSG 2000 heißt es nämlich:

> „Schutzwürdige Geheimhaltungsinteressen werden bei der Verwendung sensibler Daten ausschließlich dann nicht verletzt, wenn ... 12. die Daten zum Zweck der Gesundheitsvorsorge, der medizinischen Diagnostik, der Gesundheitsversorgung oder -behandlung oder für die Verwaltung von Gesundheitsdiensten erforderlich ist,

> und die Verwendung dieser Daten durch ärztliches Personal oder sonstige Personen erfolgt, die einer entsprechenden Geheimhaltungspflicht unterliegen ..."[1]

Die von IBM durchgeführte Machbarkeitsstudie zur Einführung einer elektronischen Gesundheitsakte im österreichischen Gesundheitswesen von 2006 ist der Meinung, dass sich daraus die Einführung nicht unmittelbar ableiten lässt, da die Gesundheitsakte eine andere Zielsetzung aufweist. Weiters werden laut § 9 des DSG 2000 die schutzwürdigen Geheimhaltungsinteressen dann nicht verletzt, wenn der Betroffene seine Zustimmung zur Verwendung der Daten ausdrücklich erteilt hat, wobei ein Widerruf jederzeit möglich ist[2] [20].

Laut § 4 Ziffer 14 ist eine Zustimmung

> „die gültige, insbesondere ohne Zwang abgegebene Willenserklärung des Betroffenen, dass er in Kenntnis der Sachlage für den konkreten Fall in die Verwendung seiner Daten einwilligt"[3]

Die Autoren der Machbarkeitsstudie merken unter anderem an, dass die Patienten meistens nicht alle beteiligten Personen kennen. Außerdem sei die Definition eines „konkreten Falles" unklar, da nicht geklärt ist, ob es sich dabei um einen einzelnen Wert oder die gesammelten Werte einer Krankheitsperiode handelt. Das Zustimmungsprinzip würde weiters einen undurchführbaren administrativen Aufwand bedeuten, da jeder Patient vor jeder Datenverwendung um seine Einwilligung gefragt werden müsste[20].

Die folglich notwendige Gesetzesänderung liegt zur Zeit als Gesundheitstelematikgesetz (GTelG 2011) zur Begutachtung vor[4]. Es soll regeln, wie die

[1] § 9 Z 12 DSG 2000
[2] § 9 Z 6 DSG 2000
[3] § 4 Z 14 DSG 2000
[4] Bundesgesetz, mit dem ein Gesundheitstelematikgesetz 2011 erlassen und das Allgemeine Sozialversicherungsgesetz, das Gewerbliche Sozialversicherungsgesetz, das Bauern-Sozialversicherungsgesetz, das Beamten-Kranken- und Unfallversicherungsgesetz, das Gentechnikgesetz, das Gesundheits- und Krankenpflegegesetz, das Hebammengesetz, das Medizinische Masseur- und Heilmasseurgesetz und das Strafgesetzbuch, geändert werden (Elektronische Gesundheitsakte-Gesetz – ELGA-G)

elektronische Gesundheitsakte (ELGA) im österreichischen Gesundheitswesen funktioniert und definiert das notwendige Umfeld. Im Konkreten werden folgende Punkte behandelt:

- Begriffsbestimmungen wie „Gesundheitsdiensteanbieter" (GDA) oder „ELGA-Teilnehmer"
- Datensicherheit
- Funktionsweise der ELGA
- Rechte der Patienten bzw. Pflichten der GDA
- Änderungen diverser Sozialversicherungsgesetze sowie des Gentechnikgesetzes und des Strafgesetzbuches

An dem Gesetzesentwurf sind bereits erste Kritiken aufgekommen. Unter anderem wird das „Opt-Out-System" kritisiert, das die Verwendung der sensiblen Daten ohne die vorherige Zustimmung der Betroffenen ermöglicht und für diese kaum verständlich sein wird, welche datenschutzrechtlichen Folgen eine derartige Verwendung mit sich bringt. Der österreichische Datenschutzrat merkt außerdem an, dass die Einschränkung, welche Institutionen als Gesundheitsdiensteanbieter gelten und somit Zugriff auf die Daten bekommen, ungenügend ist. So könnten nach dem Gesetzesentwurf Rechtsanwälte oder Versicherungen die regelmäßig mit Gesundheitsdaten arbeiten unter diese Definition fallen[21].

Das Medzinproduktegesetz (MPG) ist eine weitere wichtige gesetzliche Grundlage bei der Entwicklung und dem Einsatz von Software für den Gebrauch im medizinischen Sektor. War zu Beginn nicht eindeutig, ob eigenständige Software, also Software die nicht Teil eines deklarierten Medizinproduktes ist, von diesem Gesetz betroffen ist, wurde mit der Novelle des MPG als Umsetzung der Richtlinie 2007/47/EG in nationales Recht, der Begriff ausgedehnt[22]. Das Einbeziehen von Software und die damit einhergehenden Sicherheits- und Kontrollaspekte als wichtigen Schritt zeigt ein Artikel der Huffington Post von Februar 2010, in dem berichtet wird, dass die US Food and Drug Administration (FDA) Sicherheitsrichtlinien für elektronische Gesundheitssysteme in Erwägung zieht[23]. Grund dafür seien unter anderem sechs Tote

sowie 44 Verletzte infolge teils schwerwiegender Fehler in Krankenhausinformationssystemen. Dazu zählen das Vertauschen von Informationen, Fehlzuordnungen von Daten zu Patienten oder der Datenverlust.

Das nun eingeführte MPG und die Anwendung auf Software hat umfangreiche Konsequenzen für Hersteller, Betreiber und Anwender. Hersteller müssen die Entwickelte Software einem Konformitätsbewertungsverfahren unterziehen, in dem zu prüfen ist, ob das Medizinprodukt den grundlegenden Anforderungen nach §§ 8 und 9 MPG und den einschlägigen europäischen Vorschriften entspricht. Nach erfolgreicher Prüfung wird das Produkt mit einem CE-Kennzeichen versehen. Jede nachträgliche Änderung, wie Funktionserweiterungen durch Updates, müssen sich erneut dem Konformitätsbewertungsverfahren unterziehen. Betreiber sind darüber hinaus verpflichtet, die Software nur entsprechend der Zweckbestimmung einzusetzen. Das bedeutet zum Beispiel, dass mehrere Medizinprodukte nicht ohne weiteres miteinander kombiniert werden dürfen, solange dies nicht Teil der Zweckbestimmung ist. Die Schulung des anwendenden Personals, die Überprüfung der Funktionsfähigkeit und Sicherheit des Systems sowie Meldungen über aufgetretene Vorfälle fallen weiters unter die Pflichten des Betreibers. Neben dem Hersteller und dem Betreiber können auch die Anwender für eine Haftung in Betracht kommen, da sie durch ihr Handeln mit den Medizinprodukten für allfällige Schäden verantwortlich sein können. [24, 25]

2.2 Personal Health Records (PHR)

Die patientengeführte Dokumentation des medizinischen Zustandes wird als Personal Health Record (PHR) bezeichnet. Er wird im Gegensatz zu einem EHR vom Patienten selbst erzeugt und steht unter der Kontrolle des Patienten. Auch wenn Auszüge aus seinem EHR in den PHR einfließen können, ist es die Aufgabe des Patienten dies zu veranlassen. Weitere extern erzeugte Dokumente können ebenfalls als Teil des PHR den Zustand des Patienten protokollieren und können vom Patienten selbst oder auch von medizinischen Einrichtungen wie Krankenhäusern, Fachärzten oder Apotheken erzeugt werden[26].

Wie die elektronischen Krankenakten hat auch die patientengeführte Akte

ihren Ursprung in papierbasierten Aufzeichnungen, und auch heute pflegen viele ihre persönlichen medizinischen Aufzeichnungen in dieser Form [26]. Die steigende Popularität der PHRs durch Entwicklungen wie „Health 2.0" oder Social-Networking, entstehen aber immer mehr elektronische Systeme zur privaten medizinischen Datenhaltung. Auch der teilweise sehr intensive Einsatz von marktbestimmenden Konzernen wie Google oder Microsoft, mit Google Health bzw. Microsoft HealthVault, öffnet der breiten Masse den Zugang zu elektronischen PHRs[27, 28].

Die Möglichkeiten der verschiedenen Lösungen variieren stark nach ihren Einsatzbereichen und ihrem Aufbau. Im Allgemeinen können PHRs wie folgt unterschieden werden:

- **Standalone**: Ein Teil der angebotenen Lösungen sind abgeschlossene Systeme, die alle verfügbaren Daten verwalten um sie darzustellen.
- **Tethered**: Als zweite Lösungsarchitektur existieren Systeme, die auf bestehende EHRs zugreifen um ihre Daten zu holen. Diese Systeme sind abhängig von den Betreibern des benutzten EHRs und entweder direkt von diesen Betreibern zur Verfügung gestellt, oder in enger Kooperation mit diesen entwickelt.

Der Großteil der zur Zeit verfügbaren PHRs wird jedoch eine hybride Form der beiden Architekturansätze sein, bei der die Daten im System geschlossen vorhanden sind und zusätzlich Daten aus verbundenen EHRs geholt werden können[6].

Die Funktionen der heutigen PHRs umfassen neben der bereits erwähnten Aufzeichnung und Einsicht in die persönlichen medizinischen Daten

- eine sichere Kommunikation mit medizinischen Einrichtungen,
- die Integration von Wissensdatenbanken (z.B. MedlinePlus[5] oder Healthwise[6]),
- die Verlängerung von Rezepten oder

[5] http://www.nlm.nih.gov/medlineplus/connect/overview.html
[6] http://www.healthwise.org/

- Funktionen die auf spezielle Gesundheitsaspekte, wie z.B. chronische Krankheiten oder auf spezielle Zielgruppen wie Kinder oder Migranten zugeschnitten sind.

Auch die Benutzergruppen unterscheiden sich bei den einzelnen Projekten voneinander. Während es in vielen Projekten keine Einschränkungen für Benutzer gibt, wie zum Beispiel Google Health, sind es vor allem die mit bestehenden EHRs verbundenen Serviceanbieter, die Voraussetzungen, wie den Abschluss einer Versicherung oder die Mitgliedschaft in einer bestimmten Gruppe, zur Bedingung machen[26]. Ein Beispiel dafür ist UPMC HealhTrak (siehe auch Kapitel 2.6.3), dessen Betreiber als Versicherungsunternehmen und Krankenhausbetreiber ein PHR-System für seine Kunden zur Verfügung stellt[29].

2.3 Elektronische Patiententagebücher

Eine mögliche Funktion eines PHR-Systems können elektronische Tagebücher sein, die dem Patienten helfen sollen, seinen Gesundheitszustand zu protokollieren.

Bereits seit den 1940er-Jahren werden Patiententagebücher geführt, um durch die Partizipation des Patienten dessen Behandlung zu verbessern[30]. Durch das Aufkommen von eHealth ist auch ein starkes Interesse an einer digitalen Version solch eines Patiententagebuchs vorhanden. Sie erlauben es dem Patienten auf einfache Art und Weise seinen Zustand zu protokollieren und ermöglichen gleichzeitig der behandelnden Einrichtung eine verbesserte Diagnostik. Speziell bei chronischen Erkrankungen kann der Einsatz von Patiententagebüchern wesentlich zu einer optimierten Behandlung beitragen[31].

Kapitel 4 und 5 beschreiben den Entwurf sowie die Implementierung einer PHR-Anwendung für elektronische Patiententagebücher. Nach den in Kapitel 2.2 erläuterten Definitionen, handelt es sich dabei um ein hybrides System, mit der Möglichkeit sich über Kommunikationsschnittstellen mit einem EHR zu verbinden. Ob und wie der Zugriff für unterschiedliche Benutzergruppen eingeschränkt wird, ist in erster Linie vom Einsatzbereich und dem Serviceanbieter abhängig.

2.4 Sozioökonomische Aspekte

Der rasante Fortschritt im Gesundheitswesen der letzten Jahrzehnte hat nicht nur zu einer besseren Lebensqualität geführt, auch die durchschnittliche Lebenserwartung ist gleichzeitig deutlich angestiegen. Das resultiert unweigerlich in höhere Kosten für die medizinische Behandlung, um die Lebensqualität aufrecht zu erhalten[1]. Zusätzlich ist die Realisierung verschiedener EHR- bzw. PHR-Systeme und deren erfolgreicher Einsatz bereits bei einem sehr eingeschränkten Funktionsumfang enorm kostspielig. Allerdings bieten diese Systeme ein umfangreiches Einsparungspotential bei der Gesundheitsversorgung, das Staaten, Versicherungen, den medizinischen Einrichtungen und somit auch den Patienten zugute kommt[28].

Speziell PHRs fördern die Arzt-Patienten Kommunikation ungemein. Durch die im System definierten Kommunikationswege ist ein kontextbezogener und vor allem sicherer Informationsaustausch zwischen einem Patienten und seinem behandelnden Arzt möglich. Abgesehen von der einfacheren Kommunikation verändert die Verwendung von PHRs außerdem die Interaktion beider Rollen von einem episodischen hin zu einem kontinuierlichen Austausch[6].

Aber nicht nur die Arzt-Patienten Kommunikation kann durch den Einsatz solcher Anwendungen profitieren. Das Teilen von persönlichen medizinischen Information mit Familie und Freunden, anderen Patienten mit ähnlichen Eigenschaften oder fremden Ärzten über Social Networks macht sie alle zu „wellness co-producers"[32]. Solche Plattformen bieten emotionale Unterstützung, gegenseitiges Verständnis, das Teilen von Erfahrungen und Suchen von Ratschlägen und führen somit zu einer besseren Kontrolle der eigenen Gesundheit[33].

2.5 Standards für medizinische Daten

Durch den Erfolg von digitalen Informationssystemen ist eine Vielzahl an Projekten rund um den medizinischen Sektor und speziell zur Verwaltung von medizinisch relevanter Daten von Patienten entstanden (siehe Kapitel 2.6). Auch, wenn diese Systeme als isolierte Einheiten durchaus einen Mehrnutzen für ihre Benutzer darstellen, ergeben sich viele essentielle Vorteile

erst mit der Vernetzung. Zum Beispiel können Informationen ortsunabhängig zugänglich gemacht werden, was die Wartezeit verkürzt bis alle relevanten Informationen eines Patienten zusammengetragen sind und somit eventuell lebensbedrohende Situationen sowie unnötige mehrfach-Untersuchungen und Verschreibungen vermeidet. Interessante Möglichkeiten ergeben sich auch durch die Verarbeitung der umfassenden gesammelten Daten um neue medizinische Erkenntnisse zu erringen und das Auftreten von Krankheiten zu verhindern, zu erkennen und zu behandeln[8].

Um die Interoperabilität zwischen verschiedenen Systemen zu ermöglichen, müssen geeignete Standards gefunden und eingesetzt werden. Aus diversen Projekten (siehe Kapitel 2.6) und wissenschaftlichen sowie nationalen Initiativen sind demnach mehrere Standards hervorgegangen, die Formate zur Speicherung und/oder Übertragung definieren. Im Folgenden werden einige solcher Standards vorgestellt, die zum einen bereits weit verbreitet sind und zum anderen die Anforderungen (siehe Kapitel 4.1) des praktischen Teils dieser Arbeit erfüllen.

2.5.1 openEHR

Der openEHR Standard ist ein Ergebnis aus Forschungen die mit der Advanced Informatics in Medicine (AIM) Initiative der Europäischen Union begonnen haben und über die Entwicklung des Good European Health Records (GEHR) schlussendlich 1999 zur Gründung der openEHR Foundation führten[34]. Diese non-profit organization, die vom University College London (CHIME department)[7] und dem australischen Unternehmen Ocean Informatics[8] gegründet wurde, besitzt seit dem das geistige Eigentumsrecht am openEHR Standard[35].

Dieser beschreibt die Haltung und Verwaltung medizinischer Daten in EHRs. Das Kernkonzept von openEHR ist die zweischichtige Datenmodellierung durch die Verwendung von Archetypen. Die erste Schicht, das Referenz Modell (Reference Model – RM), beschreibt die technische Umsetzung eines generischen Datenmodells. Das gesamte Expertenwissen der medizinischen

[7] http://www.chime.ucl.ac.uk/
[8] http://www.oceaninformatics.com/

Domäne wird in die zweite Schicht, dem Archetypen Modell (Archetype Model – AM) ausgelagert. Dieser Ansatz erlaubt es, EHR-Systeme ohne umfangreiches medizinisches Know-How zu entwickeln, während im medizinischen Sektor die notwendigen Definitionen des klinischen Datenmodells erarbeitet werden. Dadurch, dass also das medizinisches Wissen nicht Teil der Software ist, ergeben sich folgende Vorteile: [36][8]

- Die medizinischen Definitionen (Archetypen) können nach der Fertigstellung eines EHR- oder PHR-Systems verändert werden. Vor allem im sich ständig weiterentwickelnden Gesundheitsbereich ist das von großer Bedeutung.
- Die fertigen Archetypen können (und sollen) von mehreren Institutionen verwendet werden. Auf diese Weise wird auch die so wichtige Interoperabilität zwischen den diversen EHR- und PHR-Systemen geschaffen.
- Die Entwickler von EHR- und PHR-Systemen benötigen nur geringes medizinisches Fachwissen und können sich auf die technische Umsetzung konzentrieren.

Ein Archetyp enthält Definitionen für alle Informationen, die aus medizinischer Sicht relevant sein können[37]. Dies sind vor allem die einzelnen Datenfelder mit Beschreibung, Datentyp, Kardinalität, Einheiten und deren Einschränkungen. Datenfelder können innerhalb des Archetyps zu Clustern zusammengefügt werden, um so ein Verschachteln zu ermöglichen. Zusätzlich kann angegeben werden, zu welchen Zeitpunkten oder Zeitintervallen ein weiteres Dokument eines Archetyps erstellt werden kann. Das folgende simple Beispiel soll die Eigenschaften verdeutlichen[9]:

Der Archetyp „Body-Mass-Index" ist eine *Beobachtung (Observation)* mit einem Datenfeld *„body mass index"* vom Datentyp *Menge (Quantity)*. Das Datenfeld wird in der *Einheit kg/m²* angegeben, ist *verpflichtend* und erlaubt einen *Wert zwischen 2 und 1000*. Mehrere Beobachtungen können zu

[9] Archetyp-Definition verfügbar auf der Webseite der openEHR Foundation: `http://openehr.org/svn/knowledge/archetypes/dev/html/en/openEHR-EHR-OBSERVATION.body_mass_index.v1.html`

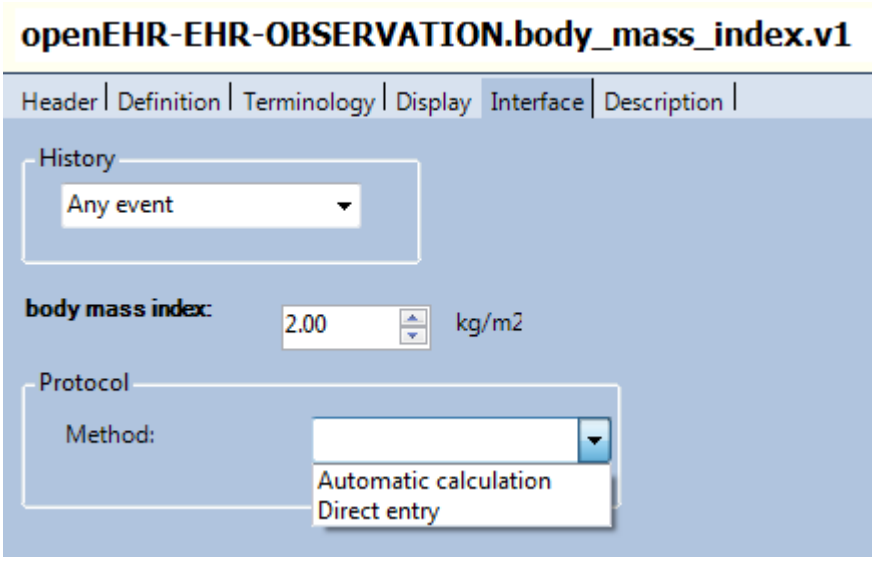


Abbildung 2.2: Generiertes Interface des Body-Mass-Index Archetyps durch den Archetype Editor von Ocean Informatics.

verschiedenen *frei wählbaren Zeitpunkten* aufgezeichnet werden. Zusätzlich kann ein Datenfeld *„Method"* einmal für alle erfolgten Beobachtungen verwendet werden. Das Feld ist vom Datentyp *„Vorgegebener Text" (Coded text)* und erlaubt die Werte *„Automatic calculation"* oder *„Direct entry"*. Dieses letzte Feld wird als Protokoll bezeichnet. Abbildung 2.2 zeigt das durch den Archetype Editor von Ocean Informatics erzeugte Interface des Body-Mass-Index Archetyps.

Archetypen können auf viele verschiedene Arten beschrieben werden. Um die Interoperabilität zu gewährleisten ist im Zuge der Entwicklung von openEHR die Archtype Definition Language (ADL) entstanden. Die ADL bietet eine formale, abstrakte Syntax zur Beschreibung von Bedingungen eines Objektes aus einer Domäne. Im Anhang A findet sich die ADL Syntax zum obigen Beispiel des Body-Mass-Index. Ein ADL-Dokument kann durch die schlichte Struktur auch mit XML repräsentiert werden[10].

Eine weitere essenzielle Eigenschaft von openEHR sind Templates. Ein Template ist ein Zusammenschluss mehrerer Archetypen zu einer Vorlage für Bildschirmmasken. Sie erlaubt es von den gewählten Archetypen einzelne

[10] XML-Schemas sind unter http://www.openehr.org/releases/1.0.2/its/XML-schema/ zu finden.

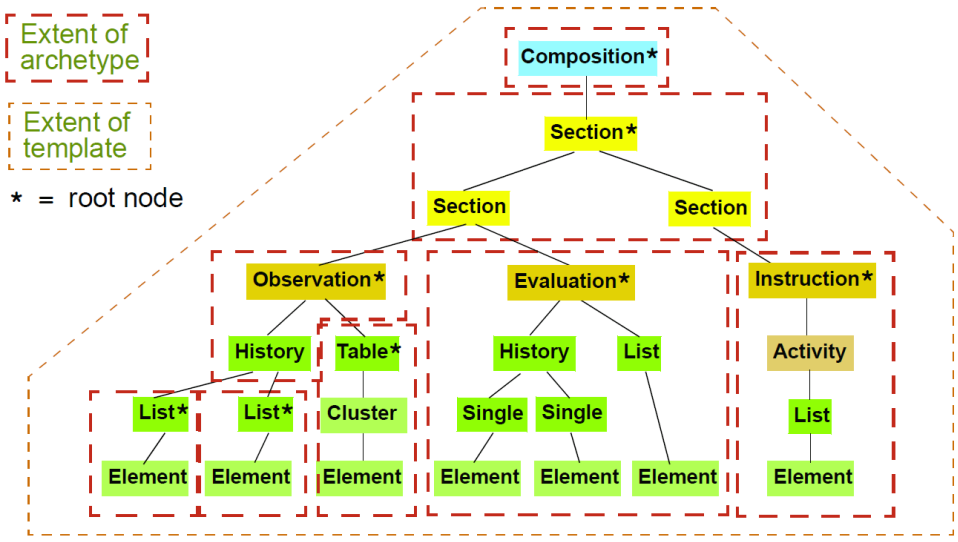


Abbildung 2.3: Organisation von Archetypen in ein Template[36].

Datenfelder oder Einschränkungen zu entfernen, die in dem vorliegenden Kontext nicht benötigt werden, oder auch neue hinzuzufügen. Außerdem werden in diesen Templates die Sprache sowie Terminologie, aus den in den Archetypen vorhandenen ausgewählt. Genau wie die Archetypen selbst, werden auch Templates mit der ADL beschrieben[36]. Abbildung 2.3 zeigt einen Zusammenschluss mehrerer Archetypen in einem Template.

2.5.2 Health Level 7 - Clinical Document Architecture (HL7 CDA)

Genau wie die openEHR Foundation ist Health Level 7 International (HL7) eine non-profit organization, die sich mit Standards zur medizinischen Datenhaltung und -übertragung beschäftigt. Im Zuge ihrer Arbeit hat HL7 eine Reihe von Standards verabschiedet, die teilweise auch ISO zertifiziert wurden, wie zum Beispiel:

- Reference Information Model (RIM)
- Clinical Document Architecture (CDA)
- Clinical Context Object Workgroup (CCOW)

- Messaging Standards (V2.x, V3)

Für diese Arbeit ist vor allem die Clinical Document Architecture (CDA) interessant, die die Strukturierung, den Inhalt und den Austausch medizinischer Dokumente definiert[8]. Nach der Fertigstellung des Release 1 im Jahr 2000 wurde das Release 2, als Teil des HL7 V3 Standards, 2005 zum ANSI Standard erklärt. Für alle CDA-Dokumente gelten sechs Eigenschaften:

- Persistenz – Die Dokumente sind von dauerhafter Existenz.
- Verantwortlichkeit für die Verwaltung des Dokuments – Eine Organisation zeichnet verantwortlich für die Verwaltung eines Dokuments.
- Signaturfähigkeit – Ein Dokument besitzt Informationen, die signiert werden können.
- Kontext – Ein Dokument wird in einen bestimmten Kontext gesetzt. Zum Beispiel umfasst ein Entlassungsbrief alle Daten der vorangegangenen Behandlungsepisode.
- Vollständigkeit – Teilinformationen eines klinischen Dokuments können nicht ohne Bezug auf das Dokument verwendet werden.
- Lesbarkeit – Die klinischen Informationen müssen in lesbarer Form Teil des Dokuments sein.[38]

Die CDA-Dokumente sind XML-basiert und stellen Tags zur Verfügung, die die Semantik von Personen- und Dokumenteigenschaften (z.B. `<provider>`, `<patient>`, `<authenticator>`, usw.) bereitstellen und für die Abbildung von Dokumentstrukturen und -hierarchien (z.B. `<section>`, `<table>`, `<paragraph>`, usw.) genutzt werden können. Abbildung 2.4 zeigt eine vereinfachte Darstellung des CDA Modells mit der Unterteilung in Header und Body sowie der obersten Elemente.

HL7 Standards, und insbesondere die CDA (Release 2) werden international für zahlreiche Projekte herangezogen. Auch in Österreich wird die CDA als Grundlage für die Datenhaltung der geplanten elektronischen Gesundheitsakte (ELGA) verwendet[39]. Für den praktischen Teil diese Arbeit ist die CDA aufgrund der enormen Komplexität im Vergleich zu anderen Standards weniger interessant.

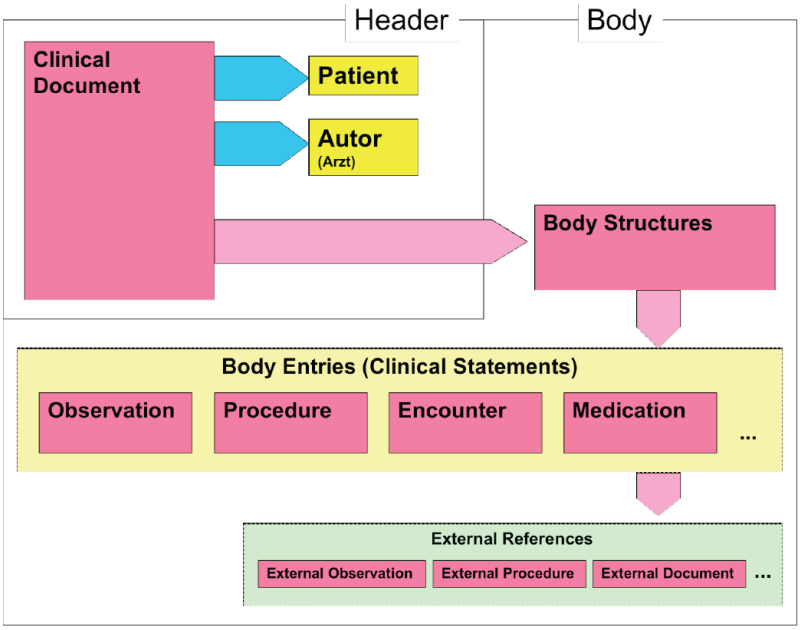


Abbildung 2.4: Vereinfachte Darstellung des CDA Modells[39].

2.5.3 Continuity of Care Record

Der von ASTM International und einigen Softwareanbietern aus dem Gesundheitsbereich entwickelte Standard Continuity of Care Record (CCR), stellt im Gegensatz zu den oben erwähnten Standards keine Definition von Dokumenten dar. Mit ihm werden Momentaufnahmen der wichtigsten administrativen, demographischen und klinischen Daten eines Patienten erzeugt, die für die Übertragung zwischen medizinischen Leistungsträgern verwendet werden können. Somit ist dieser Standard auch nicht zur dauerhaften Speicherung von Gesundheitsdaten im Zuge eines EHR geeignet[40]. Dennoch wird er von mehreren Anbietern von EHR- bzw. PHR-Systemen zur Kommunikation mit anderen Systemen zumindest teilweise unterstützt, darunter Google Health und Microsoft HealthVault[41]. Abbildung 2.5 zeigt die Elemente eines CCR Objekts, die von der Google Health API unterstützt werden[42].

Auf Basis der HL7 CDA gibt es eine Implementierung des Konzepts des Continuity of Care Records. Das Ergebnis ist der Continuity of Care Document Standard. Er ist wie die CDA Dokumenten-basiert und soll die Vorteile beider Seiten vereinen[43].

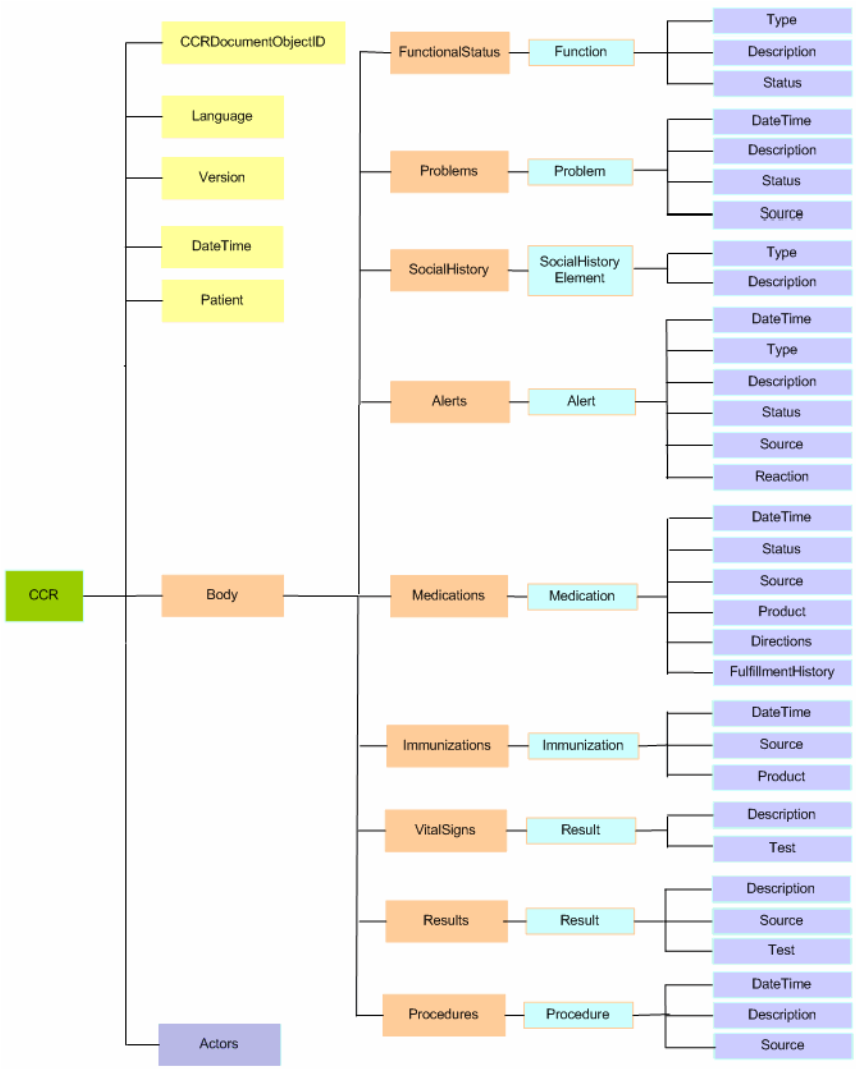


Abbildung 2.5: Dokumentstruktur des durch die Google Health API unterstützen Ausschnitts aus dem CCR Standard[42].

2.6 Aktuelle Projekte

In Deutschland gab es im Jahr 2005 ungefähr zehn umfassende Krankenhausinformationssysteme, 50 Speziallösungen für verschiedene Fachabteilungen in Krankenhäusern, 200 Arztpraxissysteme sowie weitere Lösungen für Gesundheitsämter, arbeitsmedizinische Dienste, Krankenkassen, Rehabilitationskliniken etc.[1]. Der Einzug von eHealth in unserer Gesellschaft ist also bereits voll im Gange. Was mit verschiedenen Forschungsprojekten (z.B. Advanced Informatics in Medicine[34]) begonnen hat, führte über die Entwicklung von Standards (siehe Kapitel 2.5) zur Implementierung experimenteller und konkreter Software- und Hardwareprojekte. Zentrale EHR-Systeme als lebensbegleitende Gesundheitsakte werden von vielen Staaten angestrengt oder sind bereits in Entwicklung. In diesem Kapitel werden dazu einige nationale und internationale Projekte vorgestellt. Außerdem gibt es mehrere PHR-Systeme, die Benutzern zur Verfügung gestellt werden. Speziell im Kontext des praktischen Teils dieser Arbeit werden auch Projekte erwähnt, die sich mit elektronischen Tagebüchern beschäftigen.

2.6.1 EHR-Systeme

In Österreich wurde für die Einführung eines EHR im September 2006 die Arbeitsgemeinschaft Elektronische Gesundheitsakte (Arge ELGA) gegründet die 2010 in die ELGA GmbH übergeführt wurde. Die Institution hat nach durchgeführter Machbarkeitsstudie in Zusammenarbeit mit IBM[20] einen Implementierungsleitfaden inklusive der gewählten Standards ausgegeben[39] und arbeitet aktuell an der Umsetzung, die laut Gesundheitsminister Stöger mit 2015 soweit fortgeschritten sein wird, dass alle Befunde gespeichert sein könnten[11].

Wie im Kapitel 2.1.3 erläutert, würde durch die geplante Gesetzesänderung und durch die Einführung des Gesundheitstelematikgesetzes keine ausdrückliche Zustimmung des Bürgers benötigt um ihn in das EHR aufzunehmen. Ein „Opt-Out-System", bei dem ein Bürger den Ausschluss aus der Datenbank beantragen kann ist vorgesehen.

[11] Meldung vom 17. Februar 2011 der Arge ELGA auf `www.arge-elga.at`.

Auf internationaler Ebene wird zum Beispiel in den USA der EHR im Zuge des „Nationwide Health Information Network" (NHIN) realisiert. Die Entwicklung des NHIN liegt in der Verantwortung des „Department of Health and Human Services" (HHS), das vom „Office of the National Coordinator for Health Information Technology" (ONCHIT) koordiniert wird und in Kooperation von privaten und staatlichen Unternehmen durchgeführt wird. Die Realisierung der „interoperable health information technology infrastructure" ist bis 2014 geplant[44].

2.6.2 PHR-Systeme

Mit dem Beginn des Projektes 1998 gilt Indivo als eines der ersten großen PHR-Projekte [45]. Die als Open-Source Projekt entwickelte Plattform ist als Drei-Schichten-Modell konzipiert. Die mittlere Schicht, der Indivo Server, stellt eine API zur Verfügung, die die diversen Client-Applikationen (Präsentationsschicht) verwenden können, um Daten von der verschlüsselten Datenschicht zu holen, oder in diese zu schreiben.

Die API ist auf einem zweiteiligen Sicherheitskonzept aufgebaut. Der erste Teil ist ein einfaches Rollenbasiertes Sicherheitskonzept, das für die Zugreifenden Institutionen gilt. Im zweiten Teil kann ein Patient zusätzlich bestimmen, welche dieser Institutionen auf welchen Teil seiner Informationen Zugriff erhalten. Nur wenn beide Teile einen Zugriff erlauben, ist das Auslesen oder Ändern der Daten möglich. Abbildung 2.6 zeigt die Architektur von Indivo.

Mittlerweile ist aus dem Indivo Projekt das System Indivo X im Bostoner Children's Hospital Informatics Program (CHIP) resultiert [46], das eine weiterentwickelte Open-Source Plattform darstellt. Die Abbildung 2.7 zeigt das Indivo X Frontend mit der Darstellung eines Labortests.

Den gleichen architektonischen Ansatz verfolgen auch weitere Produzenten von PHR-Systemen, wie zum Beispiel Microsoft mit HealthVault oder Dossia. Beide haben zumindest einen Teil der Indivo-Plattform in ihren Source-Code integriert [7]. Auch Google baut auf diesem Ansatz auf und ist mit dem Portal Google Health auf dem Markt stark vertreten. Durch die Integration von

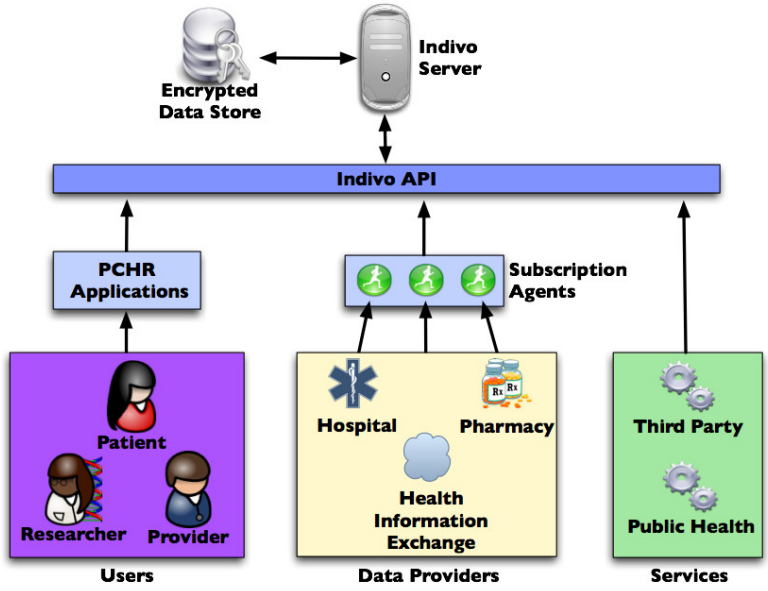


Abbildung 2.6: Plattformarchitektur des Indivo-Projekts.[45]

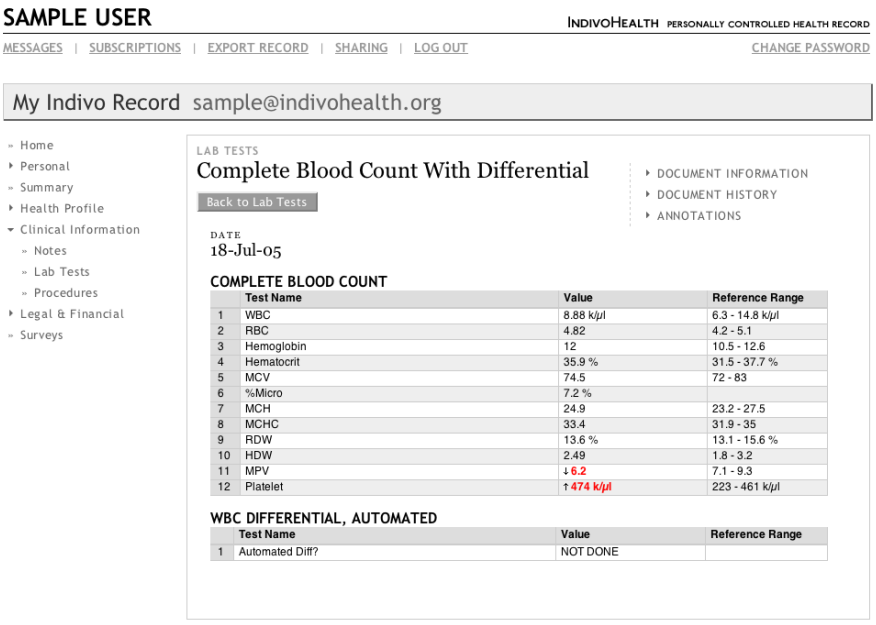


Abbildung 2.7: Frontend von Indivo X mit einer Darstellung eines Labortests. [45]

möglichst vielen medizinischen Einrichtungen in die einzelnen Portale wird versucht, dem Benutzer die Möglichkeit zu geben seine Gesundheitsdaten ohne viel Aufwand an einer Stelle zu verwalten[47].

2.6.3 Elektronische Tagebücher

Es existiert bereits eine Reihe an elektronischen Patiententagebüchern, die Daten lokal oder zentral im Netz sammeln. Ein überwiegender Teil davon ist auf eine einzige Krankheit fokussiert:

- Mit **DIABTel** entwickelten und testeten Gómez et al. ein System zur Überwachung und telemedizinischen Behandlung von Diabetespatienten. Die Implementierung besteht aus zwei Client-Anwendungen, der Medical Workstation (MW) und der Patients Unit (PU). Die MW läuft auf PCs in der klinischen Einrichtung und wird dem medizinischen Personal bereitgestellt. Die PU läuft auf einem Palmtop, den der Patient mit nach Hause bekommt. Über diese Client-Anwendung kann er jeder Zeit die Daten, die die Krankheit betreffen, wie z.B. Blutzuckerwerte, Insulininjektionen oder Diätänderungen protokollieren und bei hergestellter Internetverbindung an den zugehörigen Server schicken.

  Ein besonderes Merkmal dieses Systems ist das „Telecare service", das es dem behandelnden Arzt erlaubt aufgrund der erhaltenen Daten eines Patienten, diesen über Änderungen in seiner Behandlung zu informieren. Die Kommunikation läuft über die integrierte Kommunikationsmöglichkeit via Textnachrichten. Als Ergebnis einer Studie des Systems in einem realen Arbeitsumfeld mit 10 Patienten über 6 Monate, sahen die Autoren die qualitative und quantitative Verbesserung der von den Patienten gesammelten Informationen, die dem Arzt eine Entscheidungshilfe geben sowie eine bessere Arzt-Patienten-Kommunikation. [48]

- Das University of Pittsburgh Medical Center (UPMC)[12] stellt mit dem System **UPMC HealthTrak** ihren Kunden ein PHR-System zur Ver-

[12] Zum UPMC gehören 20 Krankenhäuser sowie 400 Ambulanzen und Arztpraxen mit insgesamt 2700 angestellten Ärzten und eine Krankenversicherung mit 1,4 Millionen Kunden[29].

fügung, das ein Tagebuch-Modul speziell für Diabetes Patienten beinhaltet. Im Vergleich zum oben beschriebenen DIABTel, ist UPMC HealthTrak vollständig Web-basiert. So können Patienten ohne spezielle Hardware und nur mit einem einfachen Internetzugang von Zuhause aus Werte wie Body-Mass-Index und Blutzuckerwert angeben. Zusätzlich werden die Laborwerte, die beim Arzt aufgenommen wurden grafisch auf dieser Plattform aufbereitet. Neben der Tagebuch-Funktion können Patienten außerdem von Zuhause aus Termine mit den medizinischen Einrichtungen vereinbaren. Ein weiterer Teil bietet den Patienten die Möglichkeit, medizinische Fragen zu stellen, die von Ärzten oder anderem Personal von UPMC beantwortet werden. Bei einer Umfrage unter UPMC Kunden wurde aber die Tagebuch-Funktionalität als besonders hilfreich hervorgehoben. [49]

- Der **Med-eMonitor** der Firma InforMedix ist in erster Linie ein Gerät, das Patienten bei der Einnahme von Medikamenten unterstützen soll. Das Gerät besteht aus einem Display sowie fünf Kammern, die die wichtigsten Medikamente des Patienten enthalten. Durch eine integrierte Kommunikationsschnittstelle kann es über das Internet konfiguriert werden. Zu bestimmten Zeitpunkten erinnert das Gerät durch Signaltöne und entsprechendem Inhalt auf dem Display den Anwender an die Einnahme von Medikamenten. Die fünf Medikamentenkammern werden dabei sensorisch überwacht und die Entnahme protokolliert. Zusätzlich kann der Patient über seinen Zustand oder andere medizinisch relevante Informationen befragt werden. Die Antworten sind jedoch auf wenige Möglichkeiten begrenzt, da nur einige Tasten zur Steuerung und keine vollständige Tastatur vorhanden sind.

  Ein Beispiel für den Einsatz des Med-eMonitors nicht nur als Medikationshilfe, sondern auch als Tagebuch zeigen Ruskin et al. mit einer Studie zur Anwendung des Geräts bei Schizophrenie-Patienten. [50]

Wie der Med-eMonitor setzten auch andere Implementierungen auf den Einsatz spezieller Hardware, wie zum Beispiel angepasste Handhelds. Siegmund zeigt mit Haemoassist 2009 ein Beispiel dafür[51]. Hämophilie ist eine Blutungserkrankung, bei der schwerwiegende und lebensbedrohliche Blutungen

Abbildung 2.8: Zusammenfassende Darstellung der Daten aus mehreren elektronischen Tagebüchern in Google Health[52].

auftreten können. Um dies zu verhindern, kann eine Substitution des fehlenden Gerinnungsfaktors intravenös verabreicht werden. Das kann sowohl umgehend im Bedarfsfall oder auch prophylaktisch erfolgen. Der Einsatz des Gerinnungsfaktors unterliegt der Dokumentationspflicht und wurde bisher vorzugsweise in Papierform aufgezeichnet. Mit einer Software für Handhelds ersetzt Haemoassist diese Papierform und erlaubt den Patienten den Einsatz des Mittels komfortabel zu dokumentieren. Der Vorteil von diesem und anderen vergleichbaren Systemen liegt klar in der mobilen Anwendbarkeit.

Eine dritte Gruppe an elektronischen Patiententagebüchern stellen die großen Gesundheitsportale wie Google Health und Microsoft HealthVault dar[47]. Das Tagebuch ist dabei allerdings nur eine von vielen Funktionen die diese PHR-Systeme anbieten. Siehe auch Kapitel 2.6.2. Die Abbildungen 2.8 und 2.9 zeigen jeweils die Aufbereitete Darstellung eines oder mehrerer medizinischer Tagebücher in Google Health bzw. Microsoft HealthVault.

Was diese Systeme gemeinsam haben, ist die Einschränkung des Angebots an verschiedenen Tagebücher auf genau eines oder auf mehrere Vorlagen, die die „üblichen" Krankheiten abdecken. Keines der oben genannten Systeme erlaubt es, Tagebücher frei als strukturierte Sammlung von Fragen zum

Abbildung 2.9: Graphisch aufbereitete Ansicht der Daten eines elektronischen Tagebuchs zur Protokollierung des Körpergewichts in Microsoft HealthVault.

medizinischen Zustand zu definieren. Weiters werden alle Benutzer der Tagebücher gleich behandelt. Auf spezielle Umstände eines einzelnen Patienten kann keine Rücksicht genommen werden. Die folgenden Kapitel Beschreiben einen Entwurf eines Systems für elektronische Tagebücher, der versucht genau diese Einschränkungen aufzuheben.

Kapitel 3

Methoden

Basis dieser Arbeit war eine Recherche im Umfeld von EHR und PHR relevanten Studien und wissenschaftlichen Arbeiten, deren Ergebnisse in den Kapiteln 1 und 2 detailliert aufgeführt worden sind. Da das Ergebnis dieser Arbeit als Teilbereich eines PHR verstanden werden kann, waren speziell Arbeiten mit einem Fokus auf PHR sowie konkreten Implementierungen von besonderem Interesse. Für die schlussendliche prototypische Implementierung des Entwurfs war eine weitere Recherche im Technischen Umfeld notwendig, um die Ergebnisse des Entwurfs umsetzen zu können.

Aufgrund dieser Recherchen wurden die in Kapitel 4.1 beschriebenen Anforderungen erarbeitet. Diese Anforderungen decken die wichtigsten Grundvoraussetzungen an eine verteilte Anwendung ab, nämlich Architektur, Datenmodell, User-Interface und Datenschutz. Aus den erkannten Einschränkungen bestehender Systeme leiteten sich speziell für das Datenmodell sowie für die Benutzerschnittstelle wichtige Anforderungen ab, die diese Einschränkungen umgehen sollen.

Für den Entwurf wurden großenteils Best Practices herangezogen um für eine hohe Qualität der zu entwickelnden Software zu sorgen. Der besonders wichtige Teil der Benutzerschnittstellen wurde unter starker Verwendung von Mock-ups erarbeitet (siehe 4.4). Diese rudimentären und funktionslosen Attrappen zeigen ohne großen Aufwand und in frühen Phasen der Entwicklung das Zusammenspiel der funktionellen Komponenten mit der Benutzeroberfläche. Dieser Entwurf konnte anschließend in einen funktionalen Prototypen

umgesetzt werden, der die wichtigen Aspekte dieser Arbeit verdeutlicht. Dabei kamen unter anderem auch Technologien, wie Dependency Injection (5.2) und die Rendering Engine Razor (5.4) zum Einsatz, die dafür sorgen sollen, dass die einzelnen Module des Systems ohne großen Aufwand erweitert bzw. in anderen Anwendungen wiederverwendet werden können.

Zum Abschluss wurde ein Benutzerhandbuch sowie ein anwendungsinternes Hilfesystem entworfen, das den Benutzern einen möglichst einfachen und schnellen Einstieg erlaubt. Programmabbildungen mit gekennzeichneten Bereichen und eine anschließende Erklärung stellen dabei die Grundstruktur des Benutzerhandbuchs.

Kapitel 4

Entwurf

Dieser Abschnitt behandelt denkbare Ansätze, eine in Kapitel 1 beschriebene Plattform zu implementieren, die es Patienten ermöglicht selbstständig und unabhängig von klinischen Einrichtungen medizinisch relevante Daten über ihren Zustand zu protokollieren und diese bei Bedarf den zuständigen behandelnden Ärzten und Betreuern zukommen zu lassen. Dabei werden ausgehend von den im Folgenden beschriebenen Anforderungen die Architektur der Applikation, das Datenmodell sowie die Benutzerschnittstelle und Schnittstellen zu externen Systemen erarbeitet.

4.1 Anforderungen

Im Folgenden werden die grundsätzlichen Anforderungen an das System beschrieben. Diese Anforderungen ergeben sich aus der Motivation sowie den Zielen für dieses Projekt, ein öffentlich zugängliches elektronisches Patiententagebuch bereitzustellen, bei dem die Tagebücher zu jeder Zeit den entsprechenden Umständen individuell angepasst werden können. Aufgrund dieser Anforderungen wurden die verschiedenen Ansätze erarbeitet.

Das System erlaubt es elektronische Tagebücher zu erstellen, diese anzuzeigen und auszufüllen. Dazu kann ein administrativer Benutzer ein Tagebuch erstellen und dieses einem Patienten zuweisen. Außerdem können andere

Benutzer die Tagebücher entweder direkt verwenden oder durch weitere Adaption auf andere Fälle anpassen. Auf diese Weise werden auf der einen Seite Tagebücher unterstützt, die auf eine bestimmte Krankheit zugeschnitten sind und auf der anderen Seite können die Tagebücher speziell auf die Anforderungen eines einzelnen Patienten abgestimmt werden. Ein Patient kann sich am System anmelden und erhält die ihm zugewiesenen Tagebücher als Formulare angezeigt. Wichtig dabei ist es, mehrere Tagebücher zu einem Patienten zuweisen zu können. Ein Patient kann zu jedem beliebigen Tagebuch Daten eintragen und diese sichern. Eine Verpflichtung zur Vollständigkeit eines gesamten Tagebuchs besteht nicht, jedoch werden Tagebucheinträge, in denen verpflichtende Felder nicht ausgefüllt sind als unvollständig markiert. Bereits eingegebene Daten sind durch den Patienten jederzeit wieder abrufbar.

Die Daten werden in einer Art und Weise gespeichert, die es erlaubt, sich mit anderen Systemen in zumindest einem etablierten Standard zur Kommunikation medizinischer Daten zu verbinden. Zusätzlich ist es möglich die Wiederholrate der einzelnen Tagebücher detailliert zu bestimmen. Werden Tagebucheinträge nicht an dem dafür vorgesehenen Zeitpunkt gemacht, wird der Patient darüber aufmerksam gemacht.

Die Benutzerschnittstelle ermöglicht die vollständige Definition von Tagebüchern und deren Wiederholraten als auch die Zuweisung zu Patienten. Eine Vorschaufunktion ermöglicht es dem Ersteller eines Tagebuchs dieses während des Designvorganges zu betrachten. Auf der Patientenseite wird das aktuelle Tagebuch mit Verweisen auf vergangene sowie folgende angezeigt. Eine Möglichkeit zur übersichtlichen Aufbereitung der eingegebenen Daten in Form von Diagrammen hilft bei der Evaluierung des medizinischen Zustandes des Patienten.

4.2 Architektur

Vor allem die Architektur des Systems prägt die Stabilität, die Erweiterbarkeit als auch die Verwendbarkeit und somit die Akzeptanz des Systems bei Benutzern und administrativem Personal. Dieses Kapitel unterteilt die Architektur in zwei Hauptpunkte: Applikationsarchitektur (4.2.1) und Datenmodell (4.2.2).

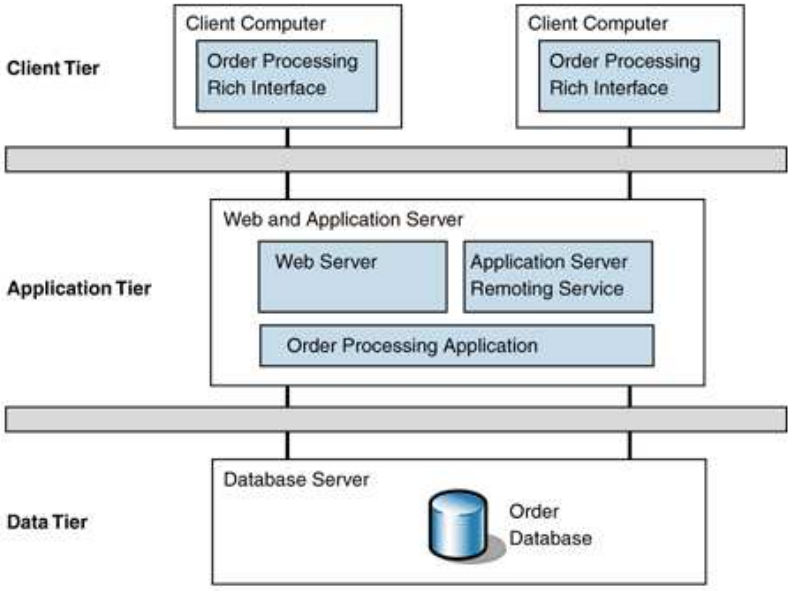


Abbildung 4.1: Aufbau einer Drei-Schichtenarchitektur (three-tier architecture). Quelle: Microsoft Patterns & Practices, Deployment Patterns, Three-Tiered Distribution (http://msdn.microsoft.com/en-us/library/ff647546.aspx)

4.2.1 Architektur der Applikation

Die prinzipielle Architektur der Applikation folgt, wie in heutigen Implementierungen von verteilten Anwendungen mit zentraler Datenquelle üblich, einer Drei-Schichtenarchitektur (three-tier architecture)[53]. In diesem Szenario stellt die Datenbank als Datenquelle die Datenschicht (Data Tier) und die Anwendung des Patienten, die Daten einzusehen bzw. einzugeben die Präsentationsschicht (Client Tier) dar. Zwischen diesen beiden Schichten arbeitet die Anwendungsschicht (Application Tier), die die Geschäftslogik repräsentiert und die Daten der jeweiligen Schicht für die Speicherung bzw. Darstellung verarbeitet. Die Abbildung 4.1 zeigt eine allgemeine Anordnung der Schichten einer Drei-Schichtenarchitektur.

Ein besonderes Augenmerk bei der Entwicklung des Systems soll auf den modularen Aufbau gelegt werden. Die Applikation wird auf Grund der sehr unterschiedlichen Arbeitsabläufe der Benutzerrollen „Patient" und „medizinisches Personal" in jeweils ein Modul aufgeteilt. Dies betrifft vor allem die

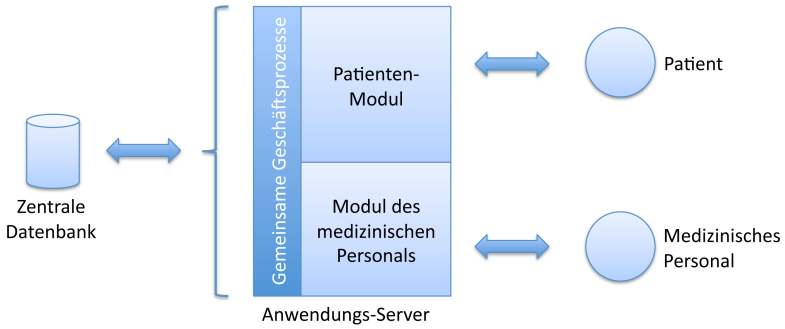


Abbildung 4.2: Modularer Aufbau der Applikation.

Präsentationsschicht aber auch große Teile der Geschäftslogik und somit der Anwendungsschicht. Das Datenmodell sowie sämtliche Mechanismen für den Zugriff darauf (z.B. OR-Mapping) werden als weiters Modul verstanden. Abbildung 4.2 zeigt den minimal erforderlichen modularen Aufbau des Systems.

Im Folgenden werden verschiedene Ansätze verfolgt, die vor allem die Implementierung der einzelnen Module stark beeinflussen. Die Ansätze sind unabhängig vom gewählten Datenmodell sowie des Serviceanbieters, und basieren auf der Annahme, dass der Patient die Anwendung von einem gewöhnlichen PC mit Internetverbindung nutzen kann. Ähnliches gilt für das medizinische Personal mit dem Unterschied, dass der freie Zugang von jedem beliebigen Ort keine Voraussetzung ist.

4.2.1.1 Vollständig Web-basiert

Um den bestehenden Anforderungen gerecht zu werden, ist eine nahezu unumgängliche Konsequenz die Implementierung des Patienten-Moduls als Web-Plattform, die mit gängigen Browsern verwendet werden kann. Soll auch der Zugriff über mobile Geräte mit limitierter Displaygröße realisiert werden, sind die mobilen Browser als weitere Client-Applikationen einzubeziehen[1].

Das Modul für das medizinische Personal ist ebenfalls als Web-basierte Lösung möglich. Der Unterschied in der Komplexität der Aufgaben im Vergleich

[1] Es existieren bereits mehrere geeignete Frameworks und Entwicklungshilfen um solche Anwendungen für Smartphones zugänglich zu machen u.a. Sencha Touch, jQuery Mobile und Mobile Boilerplate.

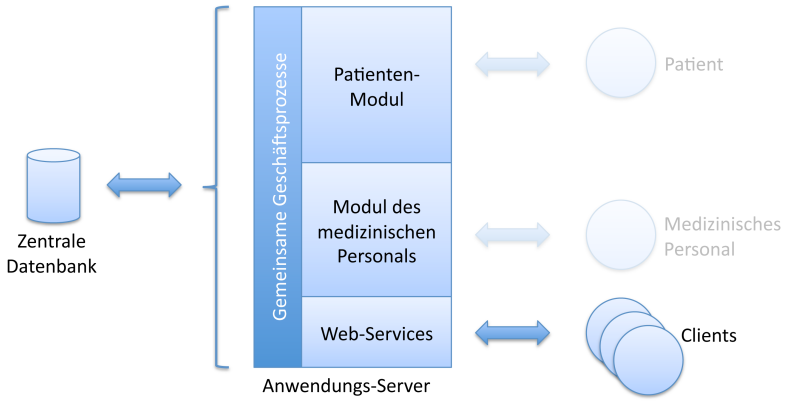


Abbildung 4.3: Aufbau der Applikation mit Webservices.

zum Patientenportal resultiert aber in einer deutlich besseren Verwendbarkeit durch Browser mit vollständiger Javascript-Unterstützung.

4.2.1.2 Integration von Webservices

Durch die Implementierung von Webservices um auf die bereits aufbereiteten Daten zuzugreifen, ergeben sich weitere Möglichkeiten, verschiedene Client-Typen in die Anwendung aufzunehmen. Eigene Desktop-Versionen für Patienten und Mediziner (siehe unten), die wesentlich komplexere Benutzerschnittstellen erlauben und somit die Qualität der Anwendung deutlich steigern können, sind somit einfacher realisierbar. Die alleinige Bereitstellung von Webservices für Patienten ist allerdings nicht im Sinne der Anforderungen an einen freien Zugangspunkt, da dem Patienten Hürden wie Inkompatibilitäten mit seinem Betriebssystem und hoher Download- und Installationsaufwand in den Weg gelegt würden.

Mobile Geräte wären eine weitere geeignete Klasse von Clients. Durch die Implementierung von Apps in den unterschiedlichen mobilen Betriebssystemen, kann eine schnelle Verfügbarkeit des Systems für die Benutzer ermöglicht werden. Auch hier ist jedoch die weiter oben erwähnte Web-Anwendung unerlässlich aus den selben Gründen wie die der Desktop-Applikationen.

Abbildung 4.3 zeigt die Struktur des Systems, wenn Webservices bereitgestellt werden, die von verschiedenen Clients verwendet werden.

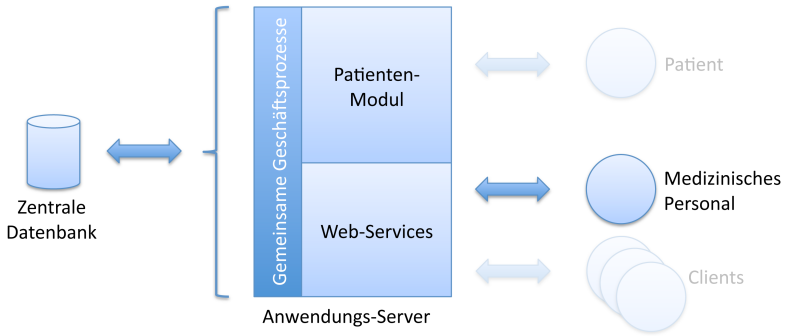


Abbildung 4.4: Aufbau der Applikation bei Einsatz einer Desktop-Applikation für das medizinische Personal.

Ein Webservices Modul ist mit der mittleren Schicht der Indivo Architektur für PHR-Systeme (siehe Kapitel 2.6.2) vergleichbar. Die Implementierung würde aber ebenfalls die beim Indivo Projekt getätigten Überlegungen zum Datenschutz mit sich bringen.

4.2.1.3 Administrationsmodul als Desktop-Applikation

Der Einsatz von Webservices erleichtert die Verwendung einer Desktop-Anwendung anstelle der Web-basierten Benutzeroberfläche für das medizinische Personal. Je nach Art des Betreibers dieses Dienstes können, im Gegensatz zu den Patienten, dem medizinischen Personal gewisse Einschränkungen auferlegt werden, wie z.B. die Verwendung eines bestimmten Betriebssystems oder sogar die Verwendung ausschließlich am Arbeitsplatz. In der Abbildung 4.4 wird dieser Architekturansatz verdeutlicht.

Die Vorteile liegen klar bei der Möglichkeit, wesentlich komplexere Benutzeroberflächen bereitzustellen, die vor allem das Erstellen von Tagebuchvorlagen erheblich komfortabler gestalten. Außerdem würde bei der Verwendung des Systems das Arbeiten mit Entwürfen von Tagebüchern auch offline helfen.

4.2.2 Datenmodell

Eine zentrale Rolle im System spielt das Datenmodell. Abgesehen von der Speicherung der Daten sowie der Handhabung durch das OR-Mapping[2], bestimmt das Datenmodell auch die Funktionsvielfalt. Wird beim Entwurf des Modells alleinig davon ausgegangen, welche Funktionen die Anwendung unterstützen muss, treten beim Entwickeln von Schnittstellen zu anderen Systemen schnell unüberwindbare Hindernisse auf. Deshalb folgt der Entwurf sowie die spätere Implementierung einem Teil des openEHR Standards zur Spezifikation medizinischer Dokumente (siehe Kapitel 2.5.1).

Den größten Vorteil von openEHR im Zusammenhang mit der Entwicklung dieses Projekts ist die Abstraktion des medizinischen Know-Hows in Archetypen. Genau diese können verwendet werden, um die einzelnen Tagebücher zu repräsentieren. Die einzelnen Angaben eines Benutzers zu seinem gesundheitlichen Zustand entsprechen im Sinne von openEHR einer Beobachtung. Die Verwendung der openEHR Spezifikation wurde deshalb auch auf den Entry-Typ Observation beschränkt. Dieser Typ dient als Basisspezifikation aller mit openEHR aufgezeichneter Beobachtungen (Anhang A zeigt einen Observation-Entry in einem ADL-Dokument).

Anhang B zeigt das implementierte Datenmodell. Im Folgenden werden die Kernaspekte des Entwurfs sowie die Relation mit dem openEHR Datenmodell dargestellt.

4.2.2.1 Spezifikation der Tagebücher

Wie bereits erwähnt, wird die Definition eines Tagebuchs ähnlich einem Archetyp dargestellt. Abbildung 4.5 zeigt dieses Modell. Die einzelnen Felder eines Tagebuchs werden als *Fields* repräsentiert. Zusätzlich zu Titel und Beschreibung wird hier auch die Reihenfolge innerhalb der Feldersammlung angegeben. Über die FieldTypes werden die Datentypen der Felder definiert.

[2] OR-Mapping bezeichnet die automatisierte bidirektionale Zuweisung von Daten aus Objekten einer Objekt-orientierten Programmiersprache zu den Datensätzen relationaler Datenbanken.

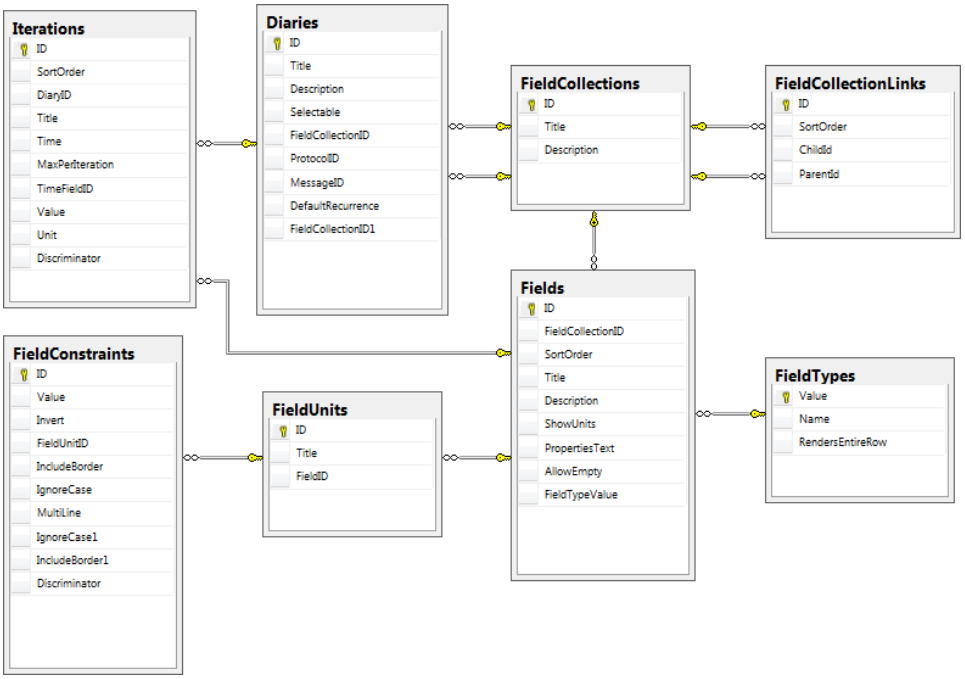


Abbildung 4.5: Datenmodell der Spezifikation der Tagebücher.

Einem Feld können beliebig viele Einheiten (FieldUnits) zugewiesen werden. Neben dem Text ist die Einheit aber auch wichtig, um Einschränkungen oder FieldConstraints angeben zu können. Nehmen wir als Beispiel das Feld Körpergröße vom Typ „Numerisch". Hängt man als Einheiten „cm" und „inch" an, dann müsste man für die Einheit „cm" eine Einschränkung von $0 \leq Wert \leq 300$ und für „inch" die Begrenzung $0 \leq Wert \leq 120$ definieren.

Die verschiedenen Felder können durch FieldCollections zu Sammlungen zusammengefasst und ein Titel sowie eine Beschreibung vergeben werden. Diese Sammlungen repräsentieren wiederverwendbare Objekte, die in mehreren Tagebüchern dargestellt werden können. Zusätzlich können die Sammlungen verschachtelt werden. So kann zum Beispiel eine FieldCollection „Blutdruck" mit Feldern für den systolischen und den diastolischen Wert im Tagebuch „Blutdruck" als auch als Zusatzinformation im Tagebuch „Adipositas" eingebettet werden. Um die FieldCollections für den Patienten tatsächlich nutzbar zu machen, werden sie in Diaries eingesetzt. Sollen mehrere FieldCollections als ein Tagebuch fungieren, muss vorher eine neue FieldCollection als Eltern-FieldCollection erzeugt werden.

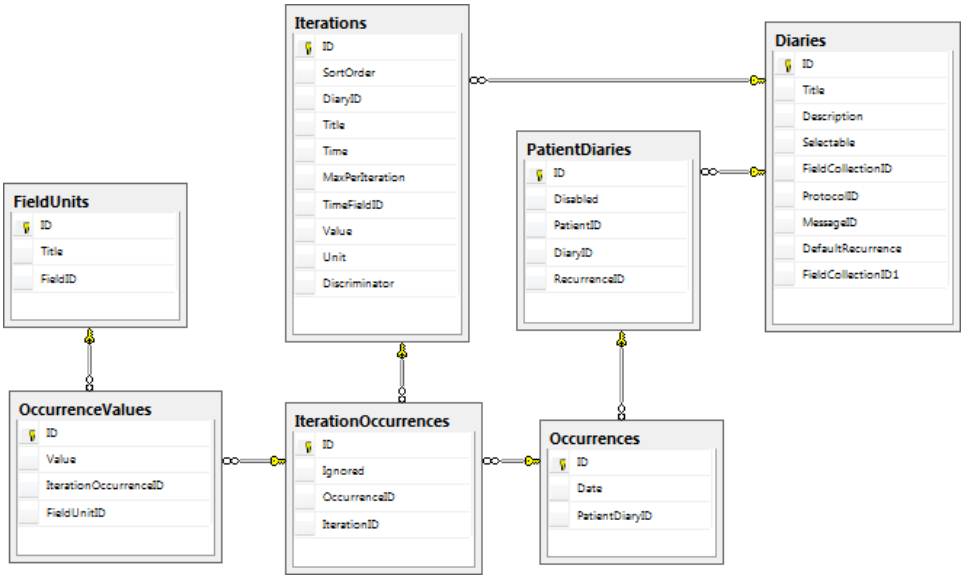


Abbildung 4.6: Datenmodell der Eingaben zu Tagebüchern.

Das mehrmalige Vorkommen eines Tagebuchs an einem Tag wird mit Iterationen geregelt. Sie entsprechen in etwa den Events von openEHR, werden also unterschieden in unspezifizierte Iterationen, Iterationen zu einer bestimmten Tageszeit und Intervalliterationen. Zusätzlich werden freie Iterationen verwendet, um beliebig viele Angaben eines Tagebuchs pro Tag zu ermöglichen und Protokolliterationen helfen bei der Identifikation von Eingaben in den Protokollteil des Tagebuchs.

4.2.2.2 Vorkommen und Eingaben zu Tagebüchern

Die einzelnen Tage, an denen ein Tagebuch auszufüllen ist, werden als Occurrence gespeichert und einem Patienten sowie einem Tagebuch zugeordnet (PatientDiaries). Die konkrete Angabe eines Teils des Tagebuchs durch den Patienten wird als OccurrenceValue gespeichert, der durch die Verweise auf die gewählte Einheit des Feldes, und somit das Feld selbst, die zugehörige Iteration sowie das Datum (Occurrence) eindeutig identifizieren kann. Abbildung 4.6 zeigt die Abhängigkeiten der einzelnen Objekte.

4.3 Datenschutz

Das Datenschutzmodell dieses Entwurfs lehnt sich an dem Modell an, dass bereits im Indivo-Projekt zum Einsatz kam und von vielen großen PHR-Projekten adaptiert wurde[45, 7]. Wie bereits in Kapitel 2.6.2 erklärt, besteht das Konzept aus zwei Teilen. Der Erste ist rollenbasiert und erlaubt bestimmten Institutionen gewisse Aktionen im System. Im zweiten Teil kann der einzelne Patient bestimmen, welche Institutionen auf seinen Account Zugriff haben. Nur wenn beide Teile den Zugriff gewähren, kann dieser auch tatsächlich erfolgen. Um die Komplexität der Anwendung zu reduzieren wird der erste Teil auf eine Rolle reduziert, die zur Ausführung jeglicher Aktion ermächtigt ist. Jeder Benutzer, der in das administrative Modul einsteigt, erhält auf dessen Funktionen vollen Zugriff. Es genügt also, dass ein Patient eine einmalige Zugriffserlaubnis für einen administrativen Benutzer erteilt, damit diesem vollen Zugriff auf die Patientendaten gewährt wird. Um einem Arzt nicht die Möglichkeit zu geben, Patientendaten wie Benutzernamen oder E-Mail Adressen aus dem System zu extrahieren, existiert im System keine Auflistung aller Patienten oder eine Suchfunktion.

Der Ablauf um Zugriff zu gewähren soll vollständig in der Kontrolle des Patienten bleiben. Abbildung 4.7 zeigt das denkbare Szenario auf dem dieser Ablauf beruht.

Zuweisungen von Tagebüchern laufen auf die gleiche Weise ab. Der Patient stimmt der Tagebuchzuweisung sowie dem Zugriff durch einen Arzt in einem zu.

Der Nachteil dieses Datenschutzkonzeptes ist der generell gewährt Zugriff. Hat ein Patient mehrere Ärzte, die ihm jeweils ein Tagebuch zuweisen, so muss er allen Zugriff auf alle Daten geben. Dieser Entwurf ist auf jeden Fall um die Möglichkeit zu erweitern, einem Arzt nur Zugriff auf die Daten eines bestimmten Tagebuchs zu geben. Außerdem ist es sinnvoll zumindest zwei administrative Rollen bereitzustellen:

Editor Erstellt und modifiziert FieldCollections und Tagebücher, erhält jedoch nie Zugriff zu Benutzerdaten.

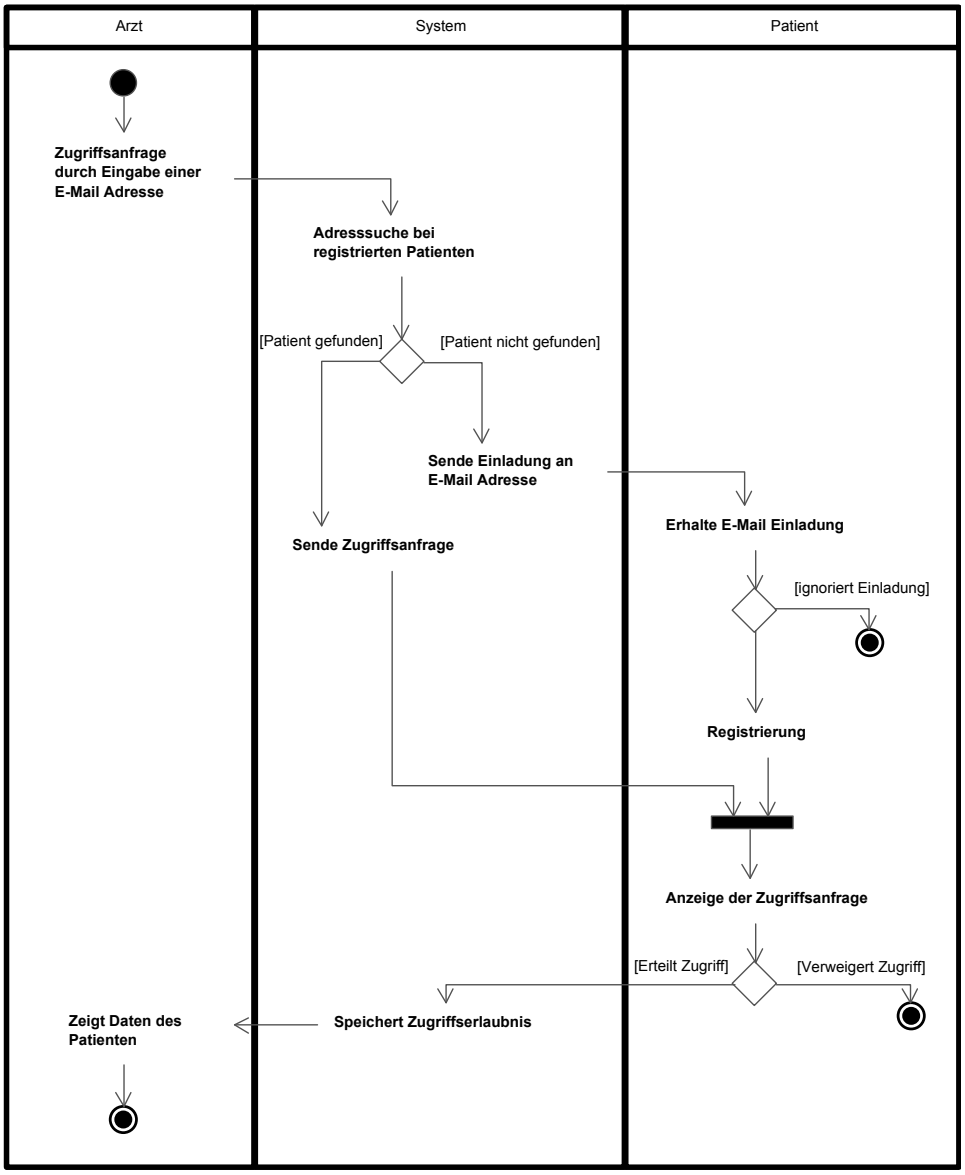


Abbildung 4.7: Aktivitätsdiagramm einer Zugriffsanfrage.

MedicalUser Erhält zusätzlich zu den Rechten des Editors die Möglichkeit Patienten ein Tagebuch zuzuweisen und Zugriffsrechte auf deren Daten zu beantragen.

Ein zusätzlicher Aspekt dieses Systems, bei dem mehrere Individuen Zugriff auf die sensiblen Daten einer Person haben können, ist genau zu wissen, wer wann auf welche Daten zugreift. Auch für den Eigentümer des Kontos kann das unter Umständen von Interesse sein. Ein Zugriffs-Log könnte somit das Vertrauen des Benutzers in das System erhöhen.

4.4 User-Interface

Neben einer stabilen Systemarchitektur ist ein gut organisiertes und intuitiv nutzbares User-Interface für die Akzeptanz der Anwendung durch den Benutzer enorm wichtig. Durch die Wahl der Systemarchitektur, genauer gesagt durch die Wahl der Anwendungsplattform, wird auch ein Teil der Benutzerschnittstelle bestimmt. Während eine Web-basierte Anwendung großen gestalterischen Freiraum bietet, wird man sich in einer Desktop-Applikation eher an vorgegebene User-Interface Standards halten. Umgekehrt ist die Darstellung komplexer Steuerelemente im Browser aufwendiger realisierbar. Dieser Umstand wird jedoch durch den breiten Einsatz von Javascript relativiert. Die resultierenden Rich Internet Applications (RIA) sind beinahe so umfangreich wie herkömmliche Anwendungen[54].

4.4.1 Patienten-Interface

Die Benutzeroberfläche des Patientenmoduls muss einige Anforderungen erfüllen, um die reibungslose Nutzbarkeit zu gewährleisten. Dafür dürfen auch keine Voraussetzungen an das technische Know-How des Benutzers gemacht werden. Es soll somit übersichtlich sein und sich auf das Wesentliche konzentrieren.

Wie in Kapitel 4.2.1 analysiert wurde, ist aufgrund der Anforderung an einen Zugang durch die Patienten von jedem Ort aus, die Entwicklung des Systems als Web-Anwendung zwingend erforderlich.

4.4.1.1 Tagebücher

Der Teil, auf den der Fokus des User-Interfaces naturgemäß gelegt wird, ist die Darstellung der Tagebücher sowie die Interaktion mit diesen.

Tagebücher sind bereits seit langem gängige Hilfsmittel um die Behandlung durch die Mithilfe des Patienten zu verbessern[30]. Um den Umstieg auf ein elektronisches Tagebuch zu erleichtern, wird beim Entwurf der Benutzerschnittstelle eine Metapher herangezogen. Diese Metapher soll ein papierenes Tagebuch darstellen, in dem jede Seite ein Formular mit allen Angaben zu einem bestimmten Tag enthält. Blättert man auf die nächste Seite, sieht man das Formular des Tages an dem die nächsten Informationen eingetragen werden können. Blättert man zurück sieht man die Formulare vergangener Tage, die wenn nötig auch neu ausgefüllt werden können. Durch diese globale Metapher verstehen Benutzer schnell den essentiellen Teil der Anwendung[55].

Analog zu dieser Metapher zeigt Abbildung 4.8 eine schematische Darstellung des User-Interface Konzepts. Im Zentrum der Ansicht werden alle Tagebücher inklusive der eventuell bereits vorhandenen Daten des gewählten Tages angezeigt. Links ist ein Verweise auf das vorherige Datum platziert, an dem zumindest ein Tagebuch vorhanden ist. Auf der rechten Seite der Ansicht steht ein Verweis auf das folgende Datum mit mindestens einem Tagebuch.

Bei der Darstellung der Tagebücher eines Tages ist auf eine schlichte Darstellung Wert zu legen. Um erneut die Metapher des papierenen Tagebuchs zu verwenden, sollen die verschiedenen Informationsanfragen gesammelt auf einer Seite nur durch verschiedene Abschnittsüberschriften getrennt angezeigt werden. Die Umsetzung dieses Prinzips wird in Abbildung 4.9 verdeutlicht. Das Tagebuch ist als abgeschlossene Einheit zu erkennen. Dagegen sind die Unterabschnitte (FieldCollections, gekennzeichnet durch einen strichlierten Rahmen) nur durch ihre Überschriften sichtbar. Dies soll einerseits zu Einfachheit der Darstellung beitragen und andererseits vermeiden, dass durch exzessives Verschachteln unübersichtliche Ansichten erzeugt werden.

Im Falle von Iterationen (siehe Abschnitt 4.2.2.1) folgt dieser Entwurf wiederum der oben erwähnten Metapher. Demzufolge werden die Iterationen wie eigene Tagebücher mit dem gleichen Titel und Inhalt angezeigt. Lediglich eine

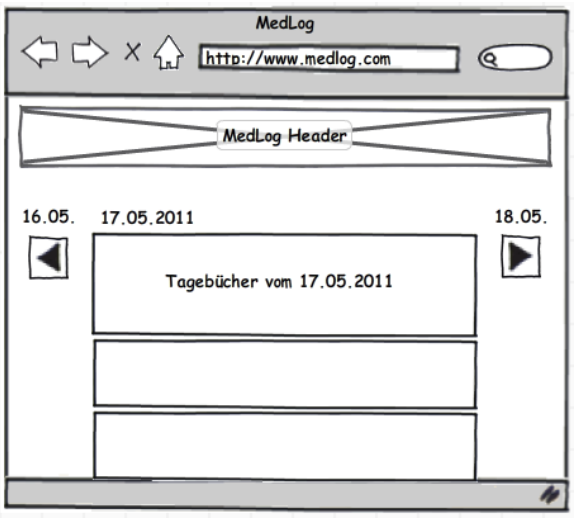


Abbildung 4.8: Schematische Darstellung eines Tagebuchs eines bestimmten Tages mit der Andeutung der Tagebuch-Metapher.

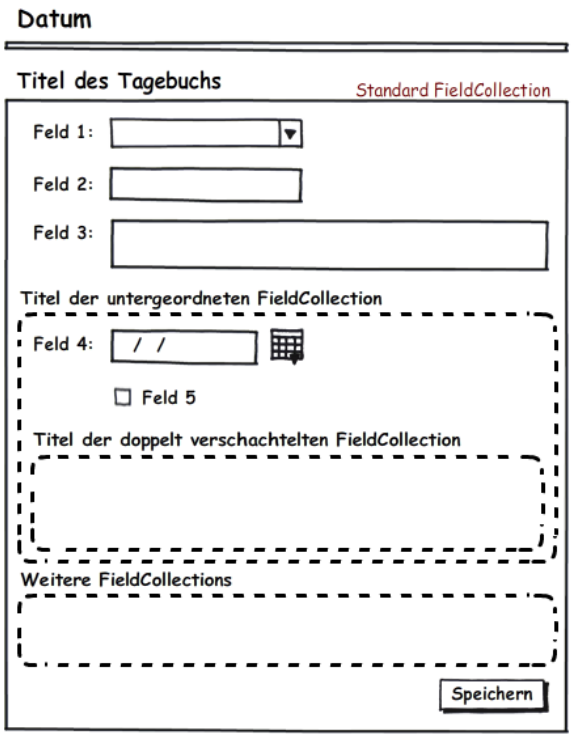


Abbildung 4.9: Schematische Darstellung eines verschachtelten Tagebuchs.

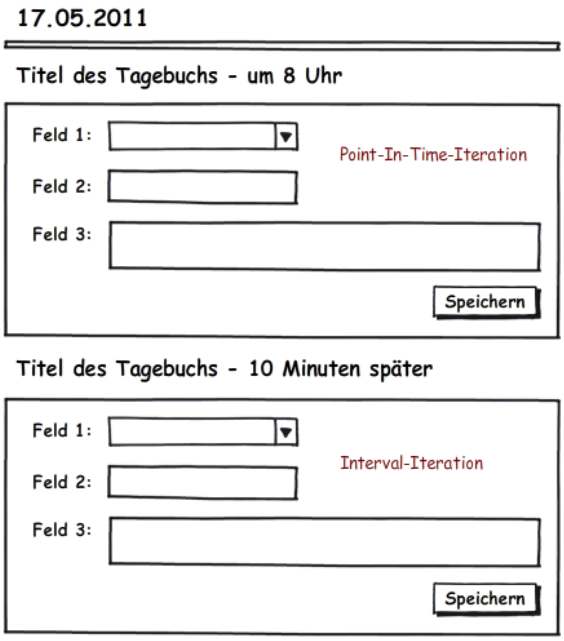


Abbildung 4.10: Schematische Darstellung eines Tagebuchs mit Iterationen.

Ergänzung am Ende des Titels soll die Iteration als solche erkennbar machen. Abbildung 4.10 zeigt ein Beispiel, in dem einer Point-In-Time-Iteration (um 8 Uhr) eines Tagebuchs eine Intervall Iteration (10 Minuten später) folgt.

Eine spezielle Art der Iteration ist die freie Iteration, die es dem Benutzer erlauben soll, ein Tagebuch beliebig oft an einem Tag auszufüllen. Ein Beispiel für den sinnvollen Einsatz einer freien Iteration ist etwa die Angabe der Körpertemperatur. Der Patient kann nach jeder Fiebermessung sofort die Werte in sein Tagebuch eintragen und ist nicht an vorgegebene Messzeiten gebunden. Abbildung 4.11 stellt dieses Beispiel dar. Im Gegensatz zu den anderen Iterationstypen ist dieser der einzige bei dem Inhalt später hinzugefügt werden kann.

Eine weitere Funktionalität ist ein Feld eines beliebigen Datentyps, das es dem Patienten erlaubt nachträglich Wertefelder hinzuzufügen. Dieses Feld ist dann von Interesse, wenn zum Beispiel zu einem bestimmten Tagebucheintrag die eingenommenen Medikamente protokolliert werden sollen. In der Abbildung 4.12 ist das Feld 2 solch ein mehrfaches Feld.

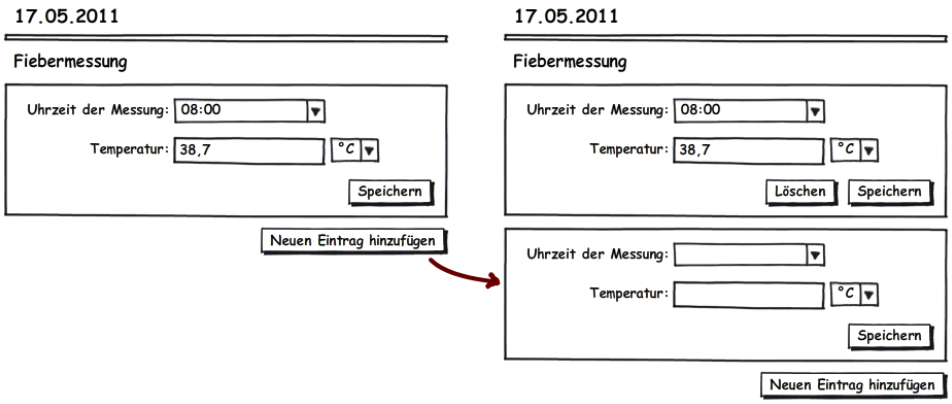


Abbildung 4.11: Funktionsweise einer freien Iteration.

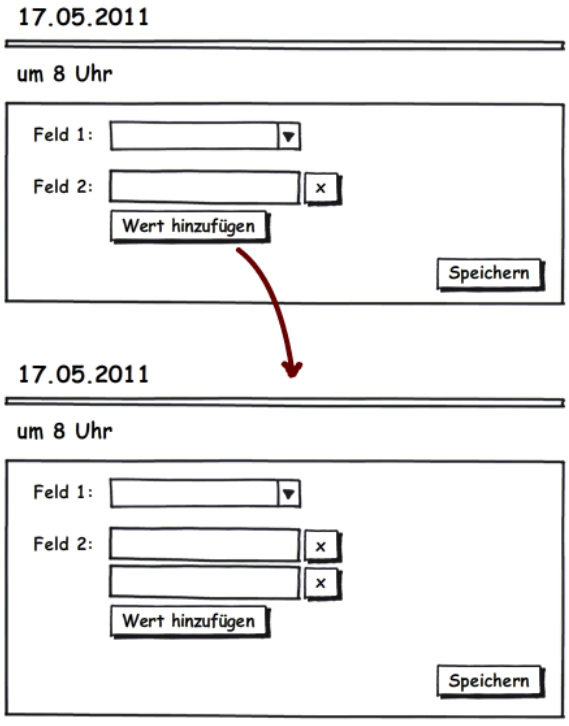


Abbildung 4.12: Schematische Darstellung eines Feldes für beliebig viele Werte.

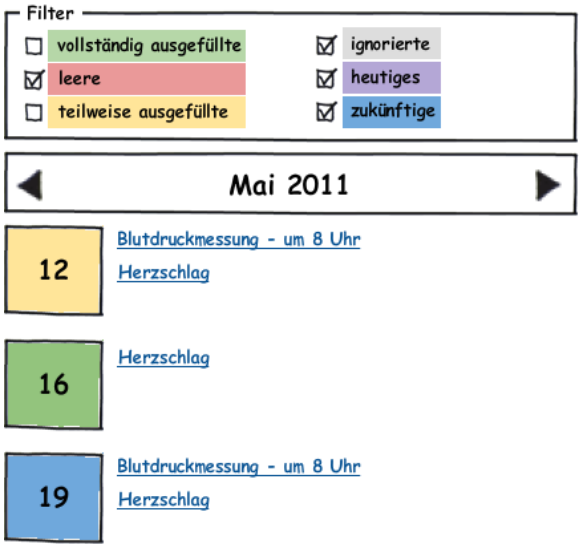


Abbildung 4.13: Kalendarische Übersicht der Vorkommen von Tagebüchern eines Monats.

4.4.1.2 Tagebuchübersicht

Als wichtiges Instrument für die Patienten um keine Einträge zu vergessen und auch für den schnellen Zugriff auf vergangene Einträge, dient eine kalendarische Übersicht der Tagebücher. In dieser werden alle vergangenen sowie zukünftigen Vorkommen der zugewiesenen Tagebücher angezeigt. Durch farbliche Markierung wird außerdem das Auffinden etwa von vergessenen Tagebüchern oder dem heutigen Tagebuch erleichtert. Abbildung 4.13 soll diese Übersicht verdeutlichen.

4.4.1.3 Visualisierung der Daten

Für die verschiedenen Daten der ausgefüllten Tagebücher ist natürlich eine geeignete Visualisierung bei der Interpretation sehr hilfreich. Speziell bei der Analyse der Werte über die Zeit kann eine adäquate Darstellung in Form von Graphen oder Tabellen sehr aufschlussreich sein. Die Definitionen der Graphen und Tabellen zu den einzelnen Tagebüchern und deren Werten übernimmt das medizinische Personal. Zum einen, um den Patienten nicht mit technischen Feinheiten zu überfordern und zum anderen, und das ist ein wesentlicher Aspekt, sollen aus fehlerhaft angeordneten oder unsinnig aufberei-

Abbildung 4.14: Schematische Darstellung der Tagebuchansicht (links) und der Tagebuchübersicht (rechts) auf einem mobilen Gerät.

teten Daten keine Missinterpretationen des eigenen medizinischen Zustands resultieren.

Eine eigene Seite listet dem Benutzer auf, welche aufbereiteten Ansichten für seine Tagebücher zur Verfügung stehen. Diverse Filtermechanismen helfen bei der genauen Analyse der Daten.

4.4.1.4 Interface für mobile Geräte

Gerade durch den Fokus auf eine möglichst Schlichte Darstellung der Tagebücher bietet sich für das Patienten-Modul der Einsatz von mobilen Geräten an. Egal ob als Apps, oder durch den Einsatz speziell aufbereiteter Webseiten sind die Formulare komfortabel auf Smartphones oder Tablet-PCs anzupassen. Abbildung 4.14 zeigt zwei Entwürfe für die Darstellung eines Tagebuchs sowie der Tagebuchübersicht im typischen Layout von Apple iPhone Apps.

Der Fokus dieser Arbeit liegt aber nicht im Entwurf eines mobilen Systems und soll hier auch nicht länger behandelt werden.

4.4.2 Arzt-Interface

Im Vergleich zur Benutzeroberfläche der Patienten ist die des medizinischen Personals wesentlich komplexer. Nachdem aber die Benutzer des Arzt-Interfaces zum einen geschult werden können und zum anderen auch Auflagen an die technische Infrastruktur auferlegt bekommen können, ist die höhere Komplexität vertretbar.

Das User-Interface auf der administrativen Seite verlangt zumindest zwei Module: einen Tagebuch-Editor und die Patientenverwaltung. Für visuelle Aufbereitung der durch die Patienten eingegebenen Daten kann die selbe Funktionalität verwendet werden, die bereits im Patienten-Interface implementiert wurde (siehe 4.4.1.3).

4.4.2.1 Tagebuch-Editor

Zum Erstellen und Bearbeiten von Tagebüchern werden zwei Oberflächen benötigt. Die erste, simple Oberfläche stellt ein Formular für die Detaildaten eines Tagebuchs dar. Also Felder für Titel, Beschreibung usw. Zusätzlich können über Auswahl- bzw. Suchfelder die FieldCollections (Standard und Protokoll) gewählt werden. Der zweite Teil ist der Teil, der für die Bearbeitung der FieldCollections verwendet wird. Da die Struktur der FieldCollections ein Hierarchie darstellt, die zusätzlich eine beliebige Verschachtelung zulässt, ist die Benutzerführung hierfür komplexer.

Eine Option ist, jede Master-Detail Beziehung gesondert darzustellen. Also ein Formular für die Stammdaten eines Objekts mit zusätzlichen Auflistungen der zugewiesenen Unterelemente. Diese Elemente können in den zugehörigen Listen erstellt oder entfernt werden. Die Bearbeitung der Elemente geschieht wiederum in den dafür vorgesehenen Formularen. So zeigt zum Beispiel das Formular einer FieldCollection die Felder für Titel und Beschreibung. Außerdem gibt es jeweils eine Liste mit den zugehörigen Feldern und untergeordneten FieldCollections. Möchte man ein Feld bearbeiten, gelangt man in das Formular des Feldes mit dessen Stammdaten sowie einer Liste aller zugeordneten Einheiten. Dieser Ansatz ist mit herkömmlichen dynamisch generierten Webseiten lösbar.

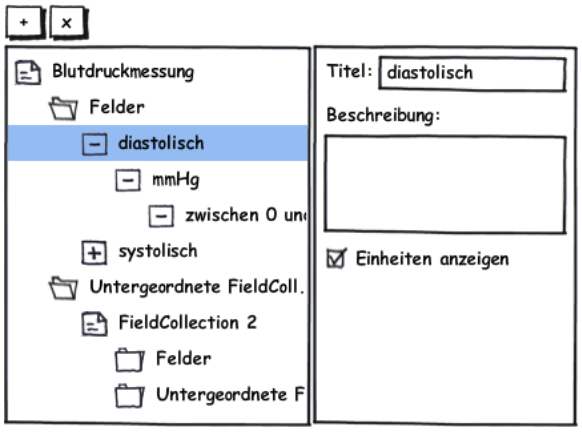


Abbildung 4.15: Schematische Darstellung eines umfangreichen Editors für verschachtelte FieldCollections.

Eine zweite Möglichkeit besteht in einer komplexen Ansicht, die als zentrale Komponente die Hierarchie als Baumstruktur anzeigt. Bei der Auswahl eines Knotens wird das zugehörige Formular angezeigt. Abbildung 4.15 verdeutlicht diese Option. Der Vorteil liegt in der sofortigen Übersicht über die Struktur und dem schnellen Zugriff auf tiefer gelegene Elemente. Während diese Ansicht der geeignete Weg für eine Desktop-Applikation darstellt, ist für die Realisierung als Webseite Javascript einzusetzen.

4.4.2.2 Patientenverwaltung

Um den in Kapitel 4.3 beschriebenen Entwurf des Datenschutzes in der Benutzeroberfläche zu realisieren, benötigt es lediglich einer Liste der Patienten, auf die der Arzt Zugriff hat. Über eine Detailseite gelangt man zu den eingegebenen Daten und deren Auswertung. Für die Zugriffsanfrage genügt ein Formular mit einem E-Mail Feld und einem Feld für einen Benutzernamen. Wird eines der Beiden ausgefüllt, wird eine Anfrage gesendet.

Um einem Patienten ein bestimmtes Tagebuch zuordnen zu können, ist das selbe Formular zu gebrauchen. Jedoch sollte das Feld zum Eingeben eines Benutzernamens die Auswahl aller Patienten anbieten, über die der Arzt bereits die Zugriffsrechte besitzt.

4.5 Schnittstellen

Die Interoperabilität ist ein wichtiger Faktor für den erfolgreichen und verbreiteten Einsatz von PHR-Systemen[26]. Um diese Interoperabilität zu erreichen, benötigt es definierte Standards zur Kommunikation der Daten (siehe 2.5) sowie Schnittstellen die diese Standards umsetzen.

4.5.1 Webservices

Webservices sind die Basis zur Plattform- und Sprach-unabhängigen Implementierung interoperabler Systeme[56]. Für das Patienten-Modul des beschriebenen Systems sind folgende Services essentiell:

1. Eine Auflistung aller vorhandener Tagebucheinträge eines bestimmten Zeitraums
2. Die Abfrage einer Definition eines Tagebuchtyps
3. Die Bereitstellung der eingegebenen Daten eines Tagebucheintrags
4. Die Übertragung von Daten eines Tagebucheintrages

Darüber hinaus werden zusätzlich administrative Services angeboten, um zum Beispiel den Datenschutz zu gewährleisten.

Durch die Konsumation dieser Webservices können beliebige Web- oder Desktop-Anwendungen die Tagebuchfunktionalität bereitstellen. Außerdem können etwaige PHR- oder EHR-Systeme die vorhandenen Daten nutzen indem sie alle Webservices bis auf das Service zur Übertragung von Tagebucheinträgen (4) verwenden.

Soll auch der administrative Bereich durch Webservices genutzt werden können, benötigt es weitere Abfragen wie eine Auflistung und die Manipulation sowie das Erstellen von Archetypen oder die Möglichkeit, Zugriffsanfragen und Tagebuchzuweisungen an Patienten zu schicken.

4.5.2 Verwendung eines Google-Health Accounts

Nicht nur die Bereitstellung der Daten und Services ist eine Möglichkeit zur Interoperabilität. Auch die Verwendung externer PHR- oder EHR-Systeme durch die Konsumation deren Webservices ist eine Variante für die Patienten, die Funktionen und Vorteile verschiedener Systeme nutzen zu können.

Dieses Kapitel zeigt diese Variante am konkreten Beispiel der Verbindung mit den Webservices von Google Health. Diese Services unterstützen einen Teil des Continuity of Care Record Standards (siehe Kapitel 2.5.3). Für das Senden von Tagebucheinträgen ist lediglich das Element „Results" notwendig, das für Testergebnisse vorgesehen ist. Je nach dem, welche Testbeschreibung in der Übertragung angegeben wird, unterteilt Google Health die Einträge in die Bereiche „Wellness" oder „Test results". Das folgende XML-Dokument ist ein Beispiel für die Übertragung eines Blutdruckwertes an die Google Health Webservices:

```
<ContinuityOfCareRecord xmlns='urn:astm-org:CCR'>
  <Body>
    <Results>
      <Result>
        <DateTime>
          <Type>
            <Text>Collection start date</Text>
          </Type>
          <ExactDateTime>2010-01-31</ExactDateTime>
        </DateTime>
        <Test>
          <Description>
            <Text>Systolic Blood Pressure</Text>
          </Description>
          <TestResult>
            <Value>120</Value>
            <Units>
              <Unit>mmHg</Unit>
            </Units>
          </TestResult>
```

```
21          </Test>
22          <Test>
23            <Description>
24              <Text>Diastolic Blood Pressure</Text>
25            </Description>
26            <TestResult>
27              <Value>80</Value>
28              <Units>
29                <Unit>mmHg</Unit>
30              </Units>
31            </TestResult>
32          </Test>
33        </Result>
34      </Results>
35    </Body>
36  </ContinuityOfCareRecord>
```

Solch ein Dokument wird immer dann abgeschickt, wenn der Benutzer in diesem System einen Tagebucheintrag speichert. Die Kommunikation funktioniert natürlich auch in die andere Richtung, um zum Beispiel in Google Health eingegebene Werte automatisch in die entsprechenden Tagebücher einzutragen. Da jedoch Google Health keine bekannten Archetypen verwendet ist die Zuordnung extrem komplex bzw. nicht immer eindeutig möglich.

Um Zugriff auf die Webservices zu erlangen wird unter anderem die AuthSub Autorisierung zur Verfügung gestellt[57]. Dabei wird zur erstmaligen Anmeldung auf eine Webseite von Google Health verwiesen, die die Eingabe von Benutzername und Passwort des Google Health Kontos bereitstellt. Meldet sich der Patient erfolgreich an, wird er inklusive einem Sicherheitstoken zurück zur ursprünglichen Webseite geleitet. Mit diesem Token ist es dem System anschließend möglich, Daten dieses Google Health Kontos abzurufen. Der Benutzer hat somit sein Passwort nicht preisgegeben und kann außerdem den Token und damit den Zugriff des Systems jederzeit deaktivieren.

4.5.3 Dateneingabe mit medizinischen Geräten

Eine ganz andere Art von Schnittstellen, nämlich Hardware-Schnittstellen können dem System ermöglichen diverse Werte direkt durch das Anschließen medizinischer Geräte ausfüllen zu lassen. Diese medizinischen Geräte können Blutdruck- oder Blutzuckermessgeräte mit entsprechender Hardwareschnittstelle sein. Aber auch Schrittzähler, Pulsuhren oder GPS-Geräte sind denkbar um zum Beispiel den Trainingsverlauf zu dokumentieren.

Um möglichst viele verschiedene Geräte zu unterstützen sind zwei Teile notwendig. Eine Abstraktionsebene die eine einheitliche Schnittstelle zur Verfügung stellt, um Daten an die Applikation zu schicken und eine Möglichkeit für den Benutzer ein Gerät mit einem oder mehreren Feldern eines Tagebuchtyps zu verknüpfen. Anschließend müssen für die verschiedenen Geräte geeignete Plug-Ins entwickelt werden, die die Abstraktionsschicht nutzen.

Kapitel 5

Implementierung

In diesem Kapitel werden einzelne relevante Details zur Implementierung des Entwurfs aus Kapitel 4 ausgeführt. Die Details betreffen vor allem programmiertechnische Einzelheiten, die für die Realisierung wichtig sind.

5.1 ASP.NET MVC 3

Für die Implementierung des praktischen Teils der Arbeit wird die Web-Programmiersprache ASP.NET des Microsoft .NET 4 Frameworks herangezogen. Darauf aufbauend kommt eine von Microsoft entwickelte und gut getestete Implementierung eines Architekturmuster zum Einsatz[58]. ASP.NET MVC 3 stellt die strukturelle Vorgabe sowie ein enorm vielfältiges Framework zur Verfügung, das abgesehen von den Basiselementen des Model-View-Controller (MVC) Musters eine Reihe weiterer Entwurfsmuster unterstützt.

Im Gegensatz zum klassischen ASP.NET ist der Page Lifecycle einer MVC Anwendung anders[59]. Ein Grund dafür ist, dass bei üblicher Verwendung kein Viewstate zum Einsatz kommt. Speziell bei dynamischen Seiten mit mehreren Objekten kann dies zu einem enormen Performancegewinn führen. Zusätzlich wird bei jedem Seitenaufruf ein Routing-Schritt durchgeführt. Anschließend wird der Controller instanziiert und dessen jeweilige Aktion ausgeführt, die in den meisten Fällen zum Rendern der View führt. Abbildung 5.1 zeigt den Page Lifecycle eines ASP.NET MVC Requests.

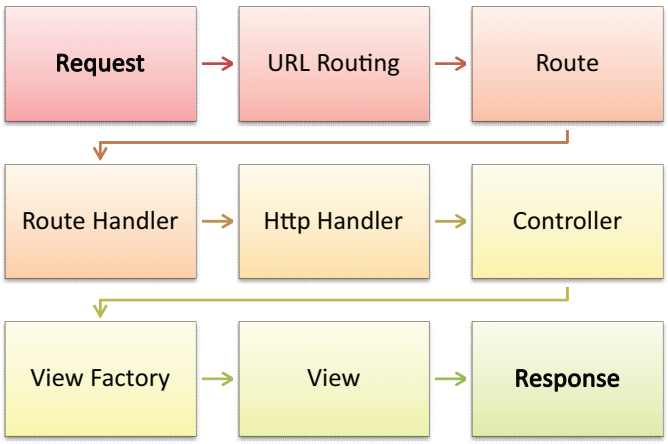


Abbildung 5.1: Lifecycle eines ASP.NET MVC Requests.

5.2 Dependency Injection

Das Entwurfsmuster Dependency Injection (DI) ist eine Form der Inversion of Control (IoC) mit dem Ziel einzelne Komponenten eines Projektes möglichst unabhängig zu gestalten[60]. Kommen zum Beispiel in einer Entwicklung Best Practices zum Einsatz, werden zur Steigerung der Wiederverwendbarkeit diverse Methoden in Services ausgelagert und durch Interfaces abstrahiert. Dadurch können verschiedene Services, die gleiche Resultate auf unterschiedliche Weise produzieren ausgetauscht werden.

Ein häufiges Beispiel dafür ist die Kommunikation mit der Datenschicht. Dabei wird ein Interface erstellt, das Methoden zum Lesen und Schreiben der Datenschicht bereitstellt. Darauf aufbauend können verschiedene Services auf unterschiedliche Datenschichten wie SQL Datenbanken, XML Dateien oder Webservices zugreifen. Im Programmteil, der solch ein Service nutzt kann so das instanziierte Service einfach durch ein anderes ausgetauscht werden. Speziell bei Projekten mit mehreren Entwicklern oder bei der Verwendung des Test Driven Development (TDD) Ansatzes liegt genau an diesem Punkt das Problem, da der Entwickler des nutzenden Programmteils bestimmt, welcher Service verwendet wird. Andere Entwickler des Projektteams, die ein alternatives Service entwickelt haben, müssten in den fremden Source-Code eingreifen. Beim Schreiben von Unit-Tests will man oft diese Datenkommunikation durch ein Mock-up ersetzen, was aus dem gleichen

Grund so nicht möglich ist. DI kehrt deshalb die Kontrolle um (Inversion of Control) und lässt die Entwickler der Services entscheiden, welche Implementierungen des Interfaces genutzt werden sollen. Der Nutzer des Service weiß über die Implementierung nicht Bescheid und vertraut darauf, dass die Methoden des Interfaces korrekt ausgeführt wurden. Die Instanziierung selbst übernimmt ein DI Container.

5.2.1 Unity Application Block

Wird .NET verwendet bietet sich der Unity Application Block (kurz Unity)[1] an, der von Microsoft selbst entwickelt wird und Teil der Microsoft Enterprise Library zur optimalen Entwicklung von Enterprise Projekts ist. Unity stellt eine Klasse `UnityContainer` bereit, die für das Instanziieren der Services mit Hilfe der Methode

```
public object Resolve(Type serviceType)
```

verantwortlich ist. Welche Objekte für welche Interfaces erstellt werden, wird in Unity entweder durch die Methode

```
RegisterType<TFrom, TTo>()
```

zur Laufzeit oder durch eine Konfigurationsdatei bestimmt. Für Definition einer Abhängigkeit gibt es in Unity zwei Arten: die Constructor Injection und die Setter Injection. Bei der Constructor Injection werden abhängige Komponenten als Parameter des Konstruktors definiert. Bei der Instanziierung der Klasse durch den Container werden die Parameter entsprechend der Konfiguration erzeugt und übergeben. Als Variante davon werden bei der Setter Injection die abhängigen Interfaces als öffentliche Setter mit dem Attribut `DependencyAttribute` versehen, die der Container bei Bedarf befüllt. Der folgende Programmteil zeigt die Definition der Abhängigkeiten unter Unity.

```
1 using Microsoft.Practices.Unity;
2
3 public class ServicesUsingClass
4 {
5   [Dependency]
```

[1] http://unity.codeplex.com/

```
6     public IServiceClass Service { get; set; }
7
8     private readonly IOtherServiceClass _otherService;
9     public ServiceUsingClass(IOtherServiceClass otherService)
10    {
11      _otherService = otherService;
12    }
13  }
```

Die Zeilen 5 und 6 entsprechen einer Setter Injection. In den Zeilen 8 bis 12 wird eine typische Implementierung einer Constructor Injection gezeigt.

Mit Unity kann zusätzlich durch die Verwendung verschiedener LifetimeManager geregelt werden, ob ein Objekt neu instanziiert wird oder ein bereits durch einen vorherigen Aufruf erstelltes Objekt zurückgeliefert wird. So kann zum Beispiel durch die Konfiguration einer Abhängigkeit unter Verwendung des `ContainerControlledLifetimeManager` das Singleton Entwurfsmuster realisiert werden, da bei jedem Aufruf von `Resolve` das selbe Objekt retourniert wird. Weitere LifetimeManager erstellen zum Beispiel für jeden Aufruf oder pro Thread ein eigenes Objekt.

5.2.2 DI unter ASP.net MVC 3

Seit der Version 3 des ASP.NET MVC Frameworks wird weitgehend das Entwurfsmuster Dependency Injection verwendet[61]. Um verschiedene DI Frameworks zu unterstützen, muss der jeweilige Container in eine Implementierung des Interfaces `IDependencyResolver` gekapselt werden. Zusätzlich ist für den Einsatz von Unity in ASP.NET MVC ein selbst entwickelter Lifetime Manager hilfreich, der Objekte nur einmal innerhalb eines HTTP Requests erzeugt. Danach ist das Framework so konzipiert, dass Controller sowie Views durch den DI Container erzeugt und deren Abhängigkeiten ebenfalls aufgelöst werden.

Bei kontinuierlicher Verwendung des Entwurfsmusters ergibt dies in Verbindung mit dem MVC Architekturmuster eine Aufteilung des Codes in Controller, Models und Services. Die Models repräsentieren die Datenstruktur.

Die Services extrahieren die Business Logik aus den Controllern die wiederum lediglich die Methoden der Services nutzen. Als Resultat ergibt diese Vorgehensweise ein hohes Maß an Wiederverwendbarkeit, klar strukturierte Datenmodellklassen, die speziell im Hinblick auf das OR-mapping (siehe 5.3) gut geeignet sind, sowie einfach zu lesende Controller.

5.3 OR-mapping mit dem Entity Framework 4

Für das Object-relational mapping (OR-mapping) wird das Microsoft Entity Framework 4 (EF4) verwendet. Das Zusatzmodul „code-first library" (public community technical preview - CTP) baut darauf auf[62]. Dieser „code-first" Entwicklungsansatz versucht, aus den entwickelten Modellklassen automatisch die Datenbankstruktur sowie das Mapping zu erzeugen. Dabei gilt „convention over configuration", also wenn gegebene Konventionen vom Entwickler eingehalten werden, benötigt es keine zusätzliche Konfiguration. Sind gewisse Aspekte durch den Mapping-Algorithmus nicht gedeckt, können die Konventionen überschrieben werden.

Eine Modellklasse wird als „plain old CLR object" oder POCO definiert. Sie benötigen keine Basisklassen oder spezielle Konfigurationsaufrufe, sondern lediglich die Eigenschaften des Modells als öffentliche Properties. Referenzen auf andere Modellklassen werden als virtuelle Properties gekennzeichnet. Dadurch wird auch das Lazy-Loading ermöglicht, das erst bei einem Zugriff auf die Eigenschaft die Datenbankabfrage auslöst. Collections auf Referenzmodelle werden durch virtuelle `System.Collections.Generic.ICollection<ReferenceType>` ermöglicht. Eine 1-zu-m-Beziehung sieht wie folgt aus:

```
1  using System.Data.Entity;
2
3  public class MasterClass
4  {
5    public int ID { get; set; }
6    public string Text { get; set; }
7
8    public virtual ICollection<DetailClass> Details { get; set; }
9  }
10
```

```
11 public class DetailClass
12 {
13   [Key]
14   public int DetId { get; set; }
15
16   public virtual MasterClass Master { get; set; }
17 }
```

Das Entity Framework generiert daraus zwei Datenbanktabellen `MasterClass` und `DetailClass`. Per Konvention wird eine Integer Eigenschaft „ID" als Primary Key verwendet (Zeile 5). Durch den Einsatz des KeyAttribute kann aber jede beliebige Eigenschaft als Primärschlüssel angegebene werden (Zeile 13). Die Beziehung der beiden Tabellen wird durch die Referenz auf die jeweils andere Klasse erkannt (Zeilen 8 und 16) und als Foreign Key umgesetzt.

Für die Spezifikation des Datenmodells benötigt es zusätzlich eine Kontextklasse, die als Ausgangspunkt der Modelldefinition gesehen werden kann. Diese Klasse wird von `System.Data.Entity.DbContext` abgeleitet und enthält alle Modellklassen als `System.Data.Entity.DbSet<ModelType>` Properties. Für die Modellklassen des oberen Beispiels sieht die Kontextklasse wie folgt aus:

```
1 using System.Data.Entity;
2
3 public class TheContext : DbContext
4 {
5   public DbSet<MasterClass> Masters { get; set; }
6   public DbSet<DetailClass> Details { get; set; }
7 }
```

Damit ist das gesamte OR-mapping erledigt. Nachdem in der Anwendungskonfiguration ein ConnectionString zu einer Datenbank unter Verwendung der Klassenbezeichnung der Kontextklasse definiert ist, kann die Datenbank bereits verwendet werden.

```
1 var myContext = new TheContext();
2 var aMaster = myContext.Masters.Where(m => m.ID == 1);
3 var aMastersDetails = aMaster.Details;
```

Für besonders komplexe Situationen im Modell, die nicht automatisch aus den Modellklassen eruiert werden können, kann die `OnModelCreating` Methode der Kontextklasse überschrieben werden. Darin können mithilfe des ModelBuilders und dessen fluent-API solche Situationen aufgelöst werden. Ein Beispiel aus dem praktischen Teil ist die rekursive m-zu-m-Beziehung von FieldCollection um die Verschachtelung zu ermöglichen. Auch wenn die code-first library eine m-zu-m-Beziehung noch auflösen kann, selbst wenn sie rekursiv ist, muss innerhalb der Relation die Reihenfolge der untergeordneten FieldCollections in der übergeordneten gespeichert werden. Deshalb wurde eine eigene Relationsklasse `FieldCollectionLink` entworfen, die zusätzlich zu den Relationen auf FieldCollection noch einen Integer-Wert für die Sortierreihenfolge enthält. Diese Relationen sind nicht mehr durch Konventionen abgedeckt und müssen deshalb konfiguriert werden. Die Kontextklasse sieht dementsprechend so aus:

```
1  public class MedLogDb : DbContext
2  {
3    // Definition der Modellklassen als Properties
4
5    protected override void OnModelCreating(ModelBuilder
         modelBuilder)
6    {
7      modelBuilder.Entity<FieldCollection>()
8        .HasMany(fc => fc.ChildLinks)
9        .WithRequired(cl => cl.Parent)
10       .HasForeignKey(cl => cl.ParentId)
11       .WillCascadeOnDelete(false);
12
13     modelBuilder.Entity<FieldCollection>()
14       .HasMany(fc => fc.ParentLinks)
15       .WithRequired(cl => cl.Child)
16       .HasForeignKey(cl => cl.ChildId)
17       .WillCascadeOnDelete(false);
18   }
19 }
```

Die code-first library überprüft beim ersten Aufruf des Datenmodells, ob dieses noch dem der Datenbank entspricht. Dazu wird eine zusätzliche Tabelle „EdmMetadata" erzeugt, die einen Hashwert des Datenmodells enthält.

Ist noch keine Datenbank vorhanden oder wurde das Datenmodell geändert, wird eine Initialisierungsmethode ausgeführt. Für dieses Projekt wurde eine eigene Initialisierungsmethode entwickelt, die die Datenbank löscht und neu erzeugt. Danach werden die einzelnen Tabellen mit den benötigten Lookup-Werten gefüllt.

5.4 Verwendung der Rendering-Engine „Razor“

Gleichzeitig zum ersten Release der ASP.NET MVC 3 Beta 2010 stellte der Entwicklungsleiter von ASP.NET Scott Guthrie eine neue Rendering Engine mit dem Codenamen „Razor” vor[63]. Diese Engine verfolgt einen Templating-Ansatz zur Erzeugung von HTML, der auf die Verwendung von .NET Code optimiert ist. Sie kann optional anstelle der Standard View-Engine von ASP.NET verwendet werden.

Die Eigenschaften von Razor sind:

Kompakt und flüssig: Die Anzahl der benötigten Zeichen eines Razor-Templates sind geringer als in vergleichbaren Engines. Der Entwickler muss Code-Blöcke nicht explizit als solche kennzeichnen, da der Parser erkennt, wann ein Code-Block zu Ende ist.

Einfach zu lernen: Die Razor-Syntax ist sehr klar und einfach aufgebaut und daher schnell erlernbar. Einige Teile sind aus der ASP.NET Rendering Engine übernommen worden. Die Code-Blöcke sind normaler C#- bzw. Visual Basic-Code.

Testbar: Die erzeugten Views sind mit Unit-Tests testbar, ohne dafür einen Controller oder einen Web-Server zu benötigen.

Im Vergleich zur klassischen ASP.NET View Syntax verwendet Razor das Zeichen `@` anstatt `<%` um den Beginn eines Code-Blocks zu markieren. Ist das Statement ein einfacher Aufruf einer Methode oder Eigenschaft, erkennt Razor das Ende des Code-Blocks. So sieht der ASP.NET View Code

```
<span>Hello <%=name %>,
  it's <%=DateTime.Now.TimeOfDay %></span>
```

mithilfe der Razor Syntax so aus:

```
<span>Hello @name, it's @DateTime.Now.TimeOfDay</span>
```

Mehrteilige Statements können durch die `@()` Syntax in der Form

```
<span>The result is @(number1 + number2)</span>
```

erzeugt werden.

Mehrzeilige Code-Blöcke können durch `@{}` deklariert werden. Razor erkennt auch innerhalb der Code-Blöcke HTML-Elemente und gibt diese korrekt aus, was vor allem bei Schleifen und Bedingungen von Vorteil ist. So wird einen einfache Schleife, die unter der ASP.NET View Engine so aussieht

```
<ul>
  <% foreach(var p in products) { %>
    <li><%=p.Name%> ($<%=p.Price%>)</li>
  <% } %>
</ul>
```

mithilfe der Razor-Engine zu

```
<ul>
  @foreach(var p in products) {
    <li>@p.Name ($@p.Price)</li>
  }
</ul>
```

Razor erkennt die HTML-Tags innerhalb der Code-Blöcke und rendert dementsprechend. In dem speziellen Fall, dass innerhalb eines Code-Blocks nur Text ohne HTML-Tag vorkommen soll, führt Razor einen eigenen Tag `<text>` ein, der das ermöglicht. So kann die folgende Bedingung dargestellt werden:

```
<span>
  It is
  @if(DateTime.Now.Hour >= 12) {
    <text>past</text>
  } else {
    <text>before</text>
  }
  noon.
</span>
```

Razor Views werden je nach verwendeter .NET-Programmiersprache als cshtml (C#) oder vbhtml (Visual Basic) Dateien gespeichert. Da dieses Projekt ausschließlich in C# geschrieben wurde, werden nur mehr cshtml Dateien erwähnt.

5.4.1 Razor unter ASP.NET MVC

Unter ASP.NET MVC gibt es einige Besonderheiten, die das Schreiben der Views erleichtern sollen.

Eine davon ist die Seiteneigenschaft `Layout`, der eine weitere Razor-View zugewiesen werden kann. In dieser Layout-View muss genau einmal die Funktion `@RenderBody()` aufgerufen werden um den Inhalt der aufrufenden View zu rendern. Auf diese Weise kann auf einfache Weise das Seitenlayout von den einzelnen Views ausgelagert werden.

Eine weiteres Feature ist die Razor-Datei „_ViewStart.cshtml", die im Projektordner Views liegt. Der Code dieser Datei wird am Anfang jeder Razor-View eingefügt. Somit ist es zum Beispiel möglich, die oben beschriebene Layout-Eigenschaft global zu setzen und nur bei Bedarf in der View selbst zu überschreiben.

Wird einer View Daten aus dem Controller zur Anzeige übergeben, geschieht dies über das Model das seit .NET 4 ein dynamischer Parameter ist. Das bedeutet aber auch, dass die View erst zur Laufzeit den Typ des Models kennt. Damit der Compiler Syntaxfehler erkennen kann, bzw. das Visual Studio Intellisense dem Entwickler helfen kann, muss also in der View der Model-Typ deklariert werden. Dies geschieht durch die Zeile

```
@model ModelType
```

innerhalb der View.

5.4.2 Rendering eines Tagebuchs

Das Rendern eines Tagebuchs ist durch die mögliche Verschachtelung sowie eventuell multipler Iterationen sehr komplex. Als Lösung dazu wurde zu jedem Element der Hierarchie, also Tagebuch, FieldCollection und Feld, eine

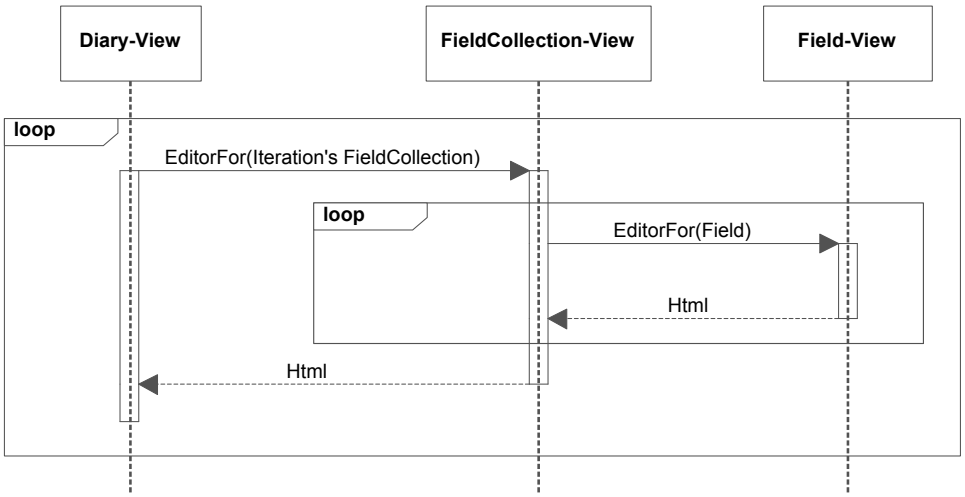


Abbildung 5.2: Sequenzdiagramm des Renderings eines Tagebuchs.

eigene View erstellt. Jede dieser Views wird als Editor Template in dem dazugehörigen Ordner abgelegt. Zusätzlich gibt es für jeden Feldtyp ein eigenes Editor Template. Durch den Befehl `Html.EditorFor(model)` der ASP.NET MVC Hilfsklasse wird eine View gesucht, die das übergebene Model verarbeitet.

Soll ein Tagebuch gerendert werden, wird die Tagebuch-View mit einem Tagebuch-Objekt aufgerufen. Diese View gibt die Ausgabesteuerung für jede Iteration des Tagebuchs an die FieldCollection-View mit der FieldCollection als Model weiter. Zusätzlich wird, wenn definiert, die Protokoll-FieldCollection zum Schluss des Tagebuchs gerendert. Die FieldCollection-View erzeugt den HTML-Code für den Container sowie für Titel und Beschreibung. Anschließend wird für jedes Feld die Field-View aufgerufen. Abbildung 5.2 zeigt diesen Rendering-Vorgang.

Um für die verschiedenen Feldtypen jeweils einen eigenen Editor anzuzeigen (zum Beispiel eine Checkbox für Ja/Nein-Felder oder eine Textarea für Beschreibungsfelder) ist für jeden Feldtyp eine eigene View erstellt worden, an die die Field-View die Kontrolle weitergibt. Der Vorteil liegt in der Erweiterbarkeit durch neue Feldtypen, da nur der Feldtyp in die Lookup-Tabelle der Datenbank eingetragen sowie eine View für diesen Typ bereitgestellt werden muss.

5.5 Javascript User-Interface

Das User-Interface sowohl des Patienten- als auch des Arzt-Moduls wurde so entwickelt, dass auch Browser ohne Javascript-Unterstützung das System vollständig unterstützen. Trotzdem sind einige Komponenten mit Javascript ergänzt, um die Verwendung des Systems für den Benutzer noch effizienter zu gestalten.

5.5.1 Telerik MVC Controls

Telerik liefert mit den ASP.NET MVC Extensions[2] eine sehr umfangreiche Bibliothek an Steuerelementen, die speziell an die Entwicklungstechniken bei MVC-Projekten angepasst ist. Zudem hat das Unternehmen schnell auf die Veröffentlichung der Rendering-Enginge Razor reagiert und eine Lösung veröffentlicht, wie die Steuerelemente zu verwenden sind[64].

Für die Entwicklungen in dieser Arbeit sind vor allem die folgenden Editoren interessant, da sie zum einen dem Benutzer hilfreiche Optionen zur Eingabe von Daten bieten und zum anderen die Client-seitige Validierung sicherstellen.

Editor Texteditor, der eine Formatierung des Textes erlaubt und HTML-Code an das System zurückliefert.

DateTimePicker Textfeld mit einem Auswahlwerkzeug für Datum und Zeit.

NumericTextBox Textfeld, das ausschließlich numerische Werte zulässt und über zwei Links das Erhöhen bzw. Verringern des aktuellen Wertes ermöglicht.

Auf das multifunktionale Auswahlfeld ComboBox wurde verzichtet, da sie ohne aktiviertes Javascript den Benutzer einschränkt.

Wie bereits erwähnt, sind die Steuerelemente an das MVC Framework angepasst, was sich bei deren Einsatz in den Views zeigt. Die Fluent-API die

[2] http://www.telerik.com/products/aspnet-mvc.aspx

zum Einsatz kommt ermöglicht die Darstellung sowie die Konfiguration des Steuerelements zugleich. Das folgende Beispiel rendert eine NumericTextBox und konfiguriert sie:

```
@Html.Telerik().NumericTextBox()
  .DecimalSeparator(".")
  .IncrementStep(.5)
  .DecimalDigits(2)
  .MinValue(0)
  .Value(Model.ImportantValue)
```

Speziell für den starken Einsatz von Javascript und CSS sind die von der Telerik Library zur Verfügung gestellten Web Asset Managers eine große Hilfe. Durch die Verwendung dieser Komponenten wird das Hinzufügen von CSS und Javascript Dateien von jeder Stelle einer View ermöglicht. Noch interessanter ist die Möglichkeit die einzelnen Dateien zu kombinieren. So werden im Optimalfall nur ein Javascript-Request sowie ein Stylsheet-Request pro Seitenaufruf durchgeführt, was die Ladezeiten deutlich verkürzt. Zusätzlich wird mithilfe dieser Funktion das lokale Caching dieser Requests definiert sowie eine Kompression der Responses angeboten. Im Falle von JQuery sowie anderer weit verbreiteter Javascript Libraries die zum Einsatz kommen, kann ein Content Delivery Network spezifiziert werden, von dem die Dateien geladen werden sollen.

5.5.2 Collapsible FieldCollections

Bei sehr umfangreichen Tagebüchern die entweder eine sehr tiefe Hierarchie an geschachtelten FieldCollections aufweisen oder viele Iterationen anzeigen, kann es aufgrund der Darstellung auf einer Webseite zu enorm langen Formularen kommen. Deshalb wurde mit Hilfe von JQuery[3] eine Möglichkeit geschaffen, einzelne Iterationen auszublenden.

Die visuelle Indikation dafür geben die aus zahlreichen User-Interfaces bekannten umrahmten Plus und Minus Zeichen am linken Rand des Tagebuchtitels. Zusätzlich führt eine Linie über die gesamte Höhe der obersten

[3] http://jquery.com/

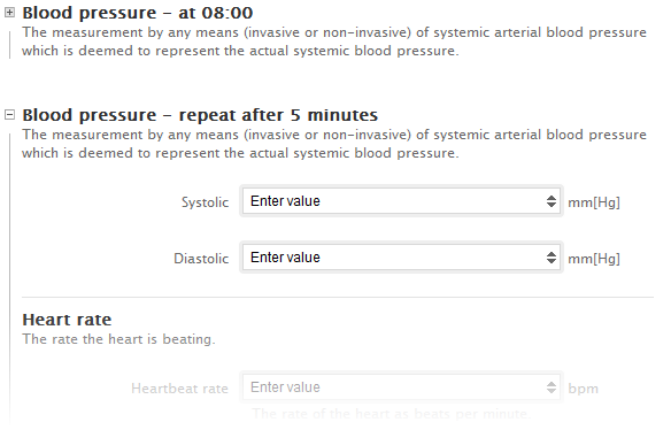


Abbildung 5.3: Tagebuch mit mehreren Iteration aus dem implementierten Prototyp. Die obere Iteration wurde ausgeblendet.

FieldCollection einer Iteration, was den Benutzer bei der Identifikation einer Iteration helfen soll. Wird eine Iteration verborgen bleiben Titel und Beschreibung angezeigt um dem Benutzer zu zeigen, welche Iteration sich dahinter verbirgt. Gespeichert kann eine Iteration nur dann werden, wenn sie sichtbar ist. Somit wird verhindert, dass Benutzer versehentlich falsche Daten übermitteln. Abbildung 5.3 zeigt ein Tagebuch mit mehreren Iteration, wobei die obere ausgeblendet wurde.

5.5.3 Visualisierung der Daten durch Diagramme

Für die graphische Aufbereitung der Daten mittels Diagrammen kommt mit „Flot" eine weitere Open-Source Javascript Library als Plug-in für JQuery zum Einsatz[4]. Da bereits das MVC Framework sowie die Telerik Steuerelemente auf JQuery setzen, erspart diese Bibliothek einiges an Ladezeit beim Seitenaufruf. Die Bibliothek bietet Linien-, Punkt-, Flächen- und Balkendiagramme an, kann aber durch Plug-ins um weitere Diagrammtypen sowie spezielle Funktionalitäten erweitert werden.

Erwähnenswert ist auch die Unterstützung beliebig vieler Datenserien innerhalb eines Diagramms. Besonders interessant ist die hervorragende Konfiguration sowie Darstellung von Daten über die Zeit, was im Falle der Tagebuchdaten fast immer eine wichtige Komponente ist.

[4] http://code.google.com/p/flot/

Für die verschiedenen Arten von Daten wurden verschiedene Seiten zur Visualisierung entwickelt. In der Datenbank werden die Visualisierungen mit den Tagebüchern und deren Felder verknüpft. Für den Patienten werden in einem Menü alle mit seinen Tagebüchern verknüpften Visualisierungen dargestellt, die er dann mit seinen Daten anzeigen lassen kann. Kapitel 5.7 zeigt drei Beispiele graphischer Aufbereitungen in der Abbildung 5.8.

Um Benutzer, die mit deaktiviertem Javascript oder Browser ohne Javascript-Unterstützung die Visualisierung aufrufen keine leere Seite zu präsentieren, werden die verschiedenen Diagramme in zwei Schritten aufgebaut. Zuerst werden die Daten vom Server in eine Tabelle geschrieben, die für den Benutzer auch lesbar ist. Wenn kein Javascript vorhanden ist, ist die Tabelle die gesamte Ausgabe der Seite. Wird jedoch der Javascript-Code durch den Browser ausgeführt, verarbeitet dieser die Daten der Tabelle zu einem Diagramm und versteckt anschließend die Tabelle.

5.6 Periodisch wiederkehrende Ereignisse (nach RFC 5545)

Zusätzlich zu den eHealth-Grundlagen ist für diese Arbeit ein Teil der „internet calendaring and scheduling core object specification", des iCalendar-Standards notwendig[65]. Um die Möglichkeit zu schaffen, bei einem Tagebuch zu definieren, in welchen Abständen es erneut auszufüllen ist, wird die „recurrence rule specification" des Standards verwendet. Dieser Teil der Spezifikation soll komplexe Angaben über periodisch wiederkehrende Ereignisse erlauben. Einfache Beispiele dafür sind:

- An jedem dritten Tag Blutdruckmessen.
- An jedem ersten Tag des Monats Körpergewicht messen.
- Jeden Montag und Mittwoch Details über die ausgeführten sportlichen Aktivitäten protokollieren.

Diese Angaben scheinen noch leicht ohne komplizierte Standards realisierbar. Aber auch folgende Beispiele sind eindeutig spezifizierbar:

- Jeden letzten Sonntag im Monat den Body-Mass-Index aufzeichnen.
- Jeden letzten oder vorletzten Werktag des Jahres die Werte eines vollständigen Blutbildes eintragen.

Die Spezifikation definiert dafür mehrere Komponenten, die in Kombination verwendet, einfache aber auch solch komplexe Angaben wie oben ermöglicht. Der Syntax der recurrence rule sieht wie folgt aus:

```
1  recur           = recur-rule-part *( ";" recur-rule-part )
2  recur-rule-part = ( "FREQ" "=" freq )
3                  / ( "UNTIL" "=" enddate )
4                  / ( "COUNT" "=" 1*DIGIT )
5                  / ( "INTERVAL" "=" 1*DIGIT )
6                  / ( "BYSECOND" "=" byseclist )
7                  / ( "BYMINUTE" "=" byminlist )
8                  / ( "BYHOUR" "=" byhrlist )
9                  / ( "BYDAY" "=" bywdaylist )
10                 / ( "BYMONTHDAY" "=" bymodaylist )
11                 / ( "BYYEARDAY" "=" byyrdaylist )
12                 / ( "BYWEEKNO" "=" bywknolist )
13                 / ( "BYMONTH" "=" bymolist )
14                 / ( "BYSETPOS" "=" bysplist )
15                 / ( "WKST" "=" weekday )
```

Die einzelnen Teile stellen also Schlüssel-Wert-Paare dar, die durch „;" getrennt werden. Kein Teil darf mehr als einmal in einer Regel vorkommen. Zusätzlich gilt, dass der Teil `FREQ` (Zeile 2) verpflichtend ist und die Teile `UNTIL` (Zeile 3) und `COUNT` (Zeile 4) nicht in der selben Regel vorkommen dürfen. Die Reihenfolge der Schlüssel-Wert-Paare ist nicht relevant.

Da sich diese Arbeit auf ganztägige Ereignisse beschränkt, da ein Tagebuch an einem bestimmten Tag angezeigt werden soll, ist es der Einfachheit halber möglich, alle Teile der Spezifikation, die sich auf Stunden, Minuten oder Sekunden beziehen zu ignorieren. Somit entfallen die Regel-Teile `BYHOUR`, `BYMINUTE` und `BYSECOND`. Auch das spätere automatisierte Verarbeiten der Regeln ist wesentlich simpler und somit performanter.

Der erste Teil `FREQ` bestimmt die Art der Regel. Mögliche Werte sind `DAILY`, `WEEKLY`, `MONTHLY` und `YEARLY`[5]. Der `INTERVAL` Teil (Zeile 5) bestimmt dann das Intervall der Frequenz. Wird dieser Teil nicht definiert, ist das Intervall 1. Mit diesen zwei Regel-Teilen ist bereits das erste Beispiel „An jedem dritten Tag Blutdruckmessen" in der Form

```
FREQ=DAILY;INTERVAL=3
```

lösbar. Die Teile beginnend mit `BY...` (ausgenommen `BYSETPOS`) (Zeilen 9 - 13) dienen danach zur weiteren Einschränkung der angegebenen Frequenz und ihrem Intervall. Die Teile können dabei mehrere Werte durch „," getrennt beinhalten. Mit diesen Informationen sind auch die beiden Beispiele „An jedem ersten Tag des Monats Körpergewicht messen" und „Jeden Montag und Mittwoch Details über die ausgeführten sportlichen Aktivitäten protokollieren" durch

```
FREQ=DAILY;BYMONTHDAY=1
```

und

```
FREQ=DAILY;BYDAY=MO,WE
```

definierbar. Verschiedene Probleme sind auf mehrere Arten lösbar. So kann das Beispiel „An jedem ersten Tag des Monats Körpergewicht messen" einfach durch

```
FREQ=MONTHLY
```

gelöst werden.

Die einschränkenden Teile `BYMONTHDAY`, `BYYEARDAY`, `BYWEEKNO` und `BYDAY` können auch ein Vorzeichen tragen, um spezielle Lösungen zu erzeugen. `BYMONTHDAY=-2` würde also auf den vorletzten Tag des Monats einschränken. Da der `BYDAY` Teil keine Zahlenwerte enthält wird das Vorzeichen samt Multiplikator vor das Wochentagskürzel gestellt. Das Beispiel „Jeden letzten Sonntag im Monat den Body-Mass-Index aufzeichnen" wird somit durch

```
FREQ=MONTHLY;BYDAY=-1SO
```

[5] `HOURLY`, `MINUTELY` und `SECONDLY` werden wie oben erwähnt auf Grund der ganztägigen Ereignisse nicht behandelt, sind aber Teil des Standards.

repräsentiert.

Ähnlich verhält es sich mit `BYSETPOS`. Dieser Spezialteil extrahiert einzelne Ergebnisse aus der durch die anderen Teile erzeugten Resultate. Das letzte Beispiel „Jeden letzten oder vorletzten Werktag des Jahres die Werte eines vollständigen Blutbildes eintragen" kann also mit Hilfe dieses Teils als

```
FREQ=YEARLY;BYDAY=MO,TU,WE,TH,FR;BYSETPOS=-1,-2
```

realisiert werden. Die ersten beiden Teile dieser Spezifikation ergeben alle Werktage des Jahres. Der `BYSETPOS` Teil nimmt aus dieser Liste anschließend die letzten beiden Elemente heraus.

Die Regelteile `UNTIL` und `COUNT` sind einfache Einschränkungen. Während es `UNTIL` durch die Angabe eines Datums ermöglicht, die Wiederholung zu einem bestimmten Zeitpunkt enden zu lassen, schränkt `COUNT` die Ergebnisse schlicht auf eine maximale Anzahl ein. Mit dem letzten Teil `WKST` kann abschließend noch der erste Tag der Woche angegeben werden. Das ist zum Beispiel bei

```
FREQ=YEARLY;BYWEEKNO=1;WKST=MO
```

also, der erste Tag der ersten Woche jedes Jahres, signifikant. In diesem Fall ist es der Montag, was auch als Standard gilt.

Für die Berechnung der einzelnen Ergebnisse eines wiederkehrenden Ereignisses ist außer der Regel noch das Anfangsdatum wichtig, also jenes Datum, ab dem die Berechnung startet. Dabei ist es nicht notwendig, dass das Anfangsdatum ein gültiges Ergebnis ist.

5.7 Prototyp

Als Resultat des beschriebenen Entwurfs sowie der erarbeiteten technischen Details in diesem Kapitel, ist eine funktionsfähige prototypische Implementierung entstanden, die den aufgestellten Anforderungen entspricht. Die folgenden Abbildungen sollen exemplarisch die tatsächliche Implementierung verdeutlichen. Im Benutzerhandbuch im Kapitel 7.1 sind weitere Darstellungen sowie eine Erklärung zur Verwendung der Applikation aus Sicht eines Patienten angeführt.

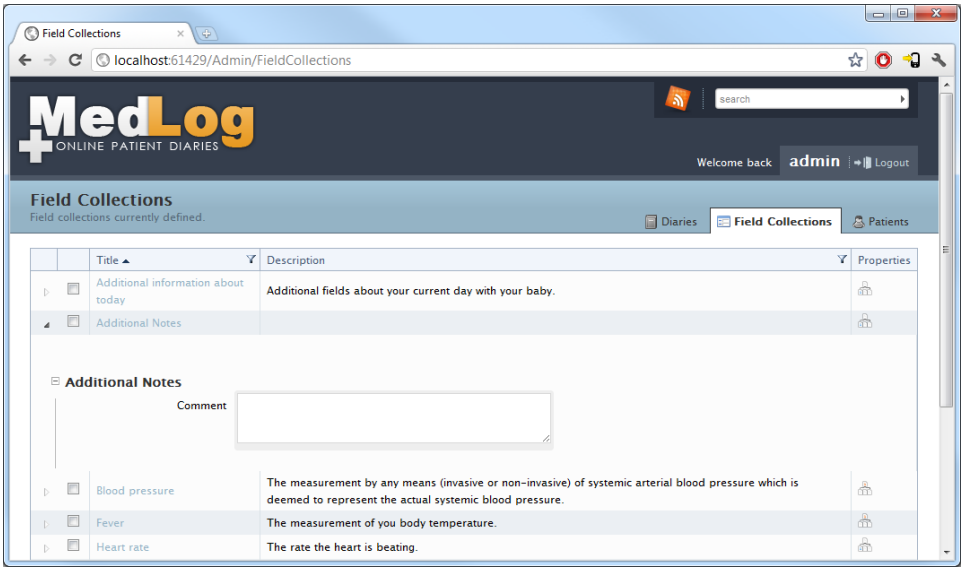


Abbildung 5.4: Auflistung der wiederverwendbaren Tagebuch-Abschnitte (Field Collections)

5.7.1 Administratives Modul

Das administrative Modul ist für die Gestaltung und die Zuweisung und Auswertung der Patiententagebücher verantwortlich. Die Abbildungen 5.4 und 5.5 zeigen die Auflistung der wiederverwendbaren Tagebuch-Abschnitte (Field Collections) und ein Formular, um ein Eingabefeld für solch einen Abschnitt zu erzeugen.

5.7.2 Patientenmodul

Auf Seiten der Patienten stehen drei wichtige Bereiche zur Verfügung. Der zentrale Bereich und auch Startseite eines jeden Patienten ist die Ansicht des heutigen Tagebuchs, von dem aus der Benutzer das nächste sowie das vorherige Formular eines Tagebuchs aufrufen kann (Abbildung 5.6). Die Tagebuchübersicht in Abbildung 5.7 zeigt eine Zusammenfassung der Tage eines bestimmten Datumsbereichs, an denen ein Tagebuch auszufüllen ist. Eine farbliche Markierung und eine Filterfunktion sollen beim Finden zum Beispiel eines vergessenen Tagebuchs helfen.

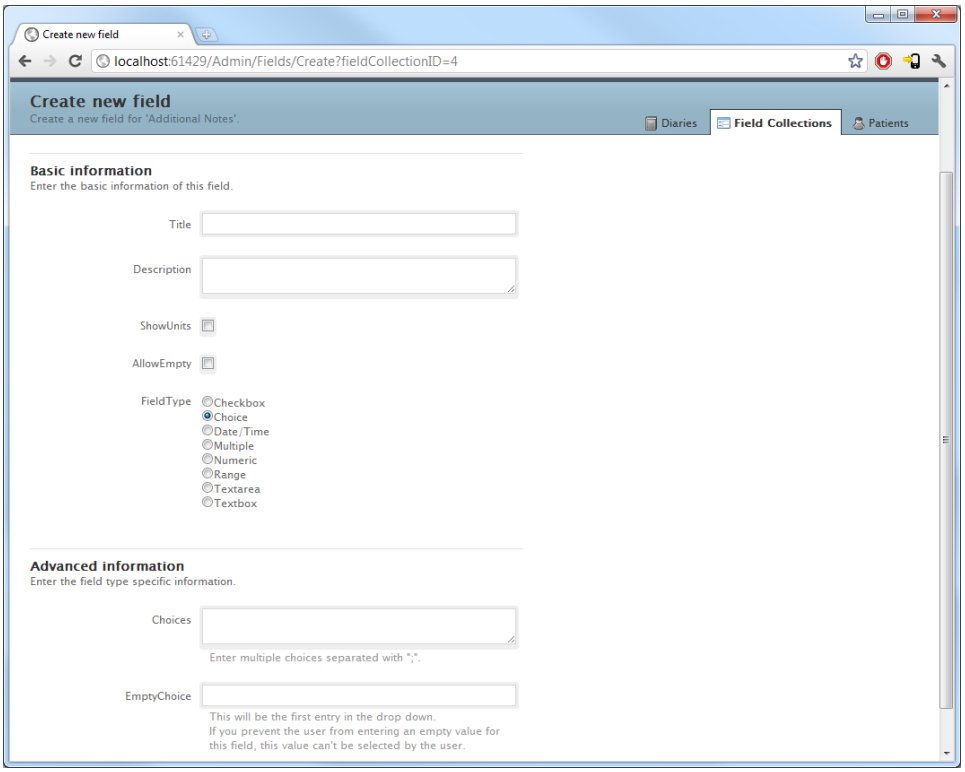


Abbildung 5.5: Formular zur Definition eines neuen Eingabefeldes eines Tagebuch-Abschnitts

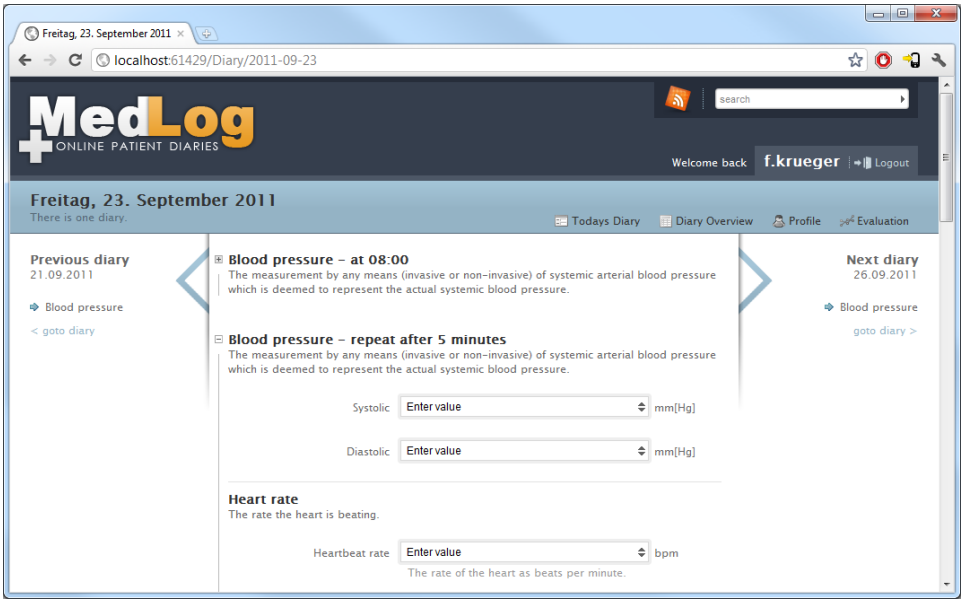


Abbildung 5.6: Tagebuchformular als Standardansicht eines Patienten

Als dritter Bereich soll die Datenauswertung bei der Interpretation der ausgefüllten Tagebücher helfen. Verschiedene graphische Aufbereitungen sollen dabei unterschiedliche Aspekte verdeutlichen. Abbildung 5.8 zeigt verschiedene Arten von Diagrammen mit konkreten Beispielen aus dem Prototyp. Das erste Diagramm stellt eine Fieberkurve dar. Zusätzlich werden mit den blauen Streifen die Zeitpunkte einer Medikation markiert. Die beiden anderen Diagramme der Abbildung verwenden die gleichen Daten, bereiten diese jedoch unterschiedlich auf. Die zweite Darstellung zeigt die Verwendung eines Plug-ins um Kreisdiagramme anzuzeigen. In der letzten Abbildung werden die Daten als Balkendiagramm zu einer Tagesübersicht angeordnet.

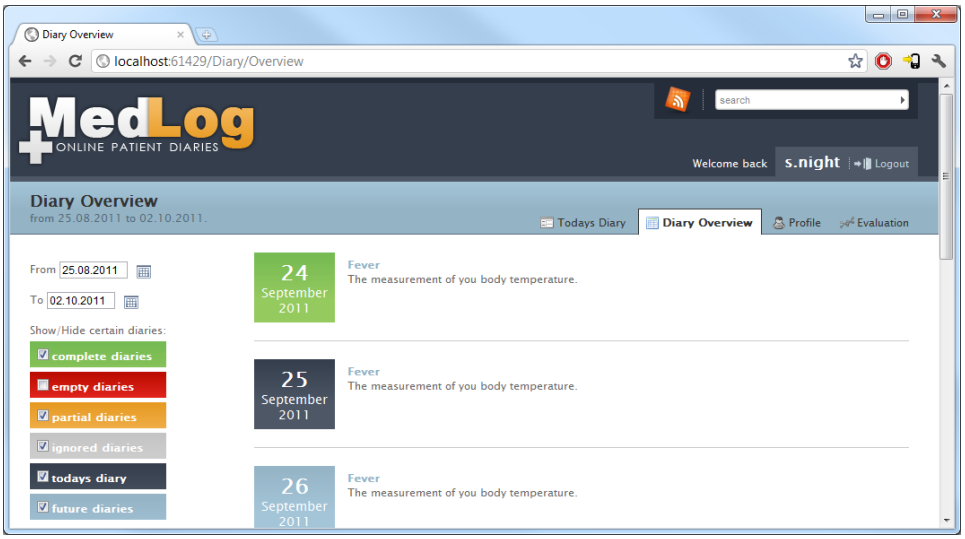


Abbildung 5.7: Übersicht über Tage, an denen Tagebücher auszufüllen sind. Der Filter links hilft beim Auffinden bestimmter Tage.

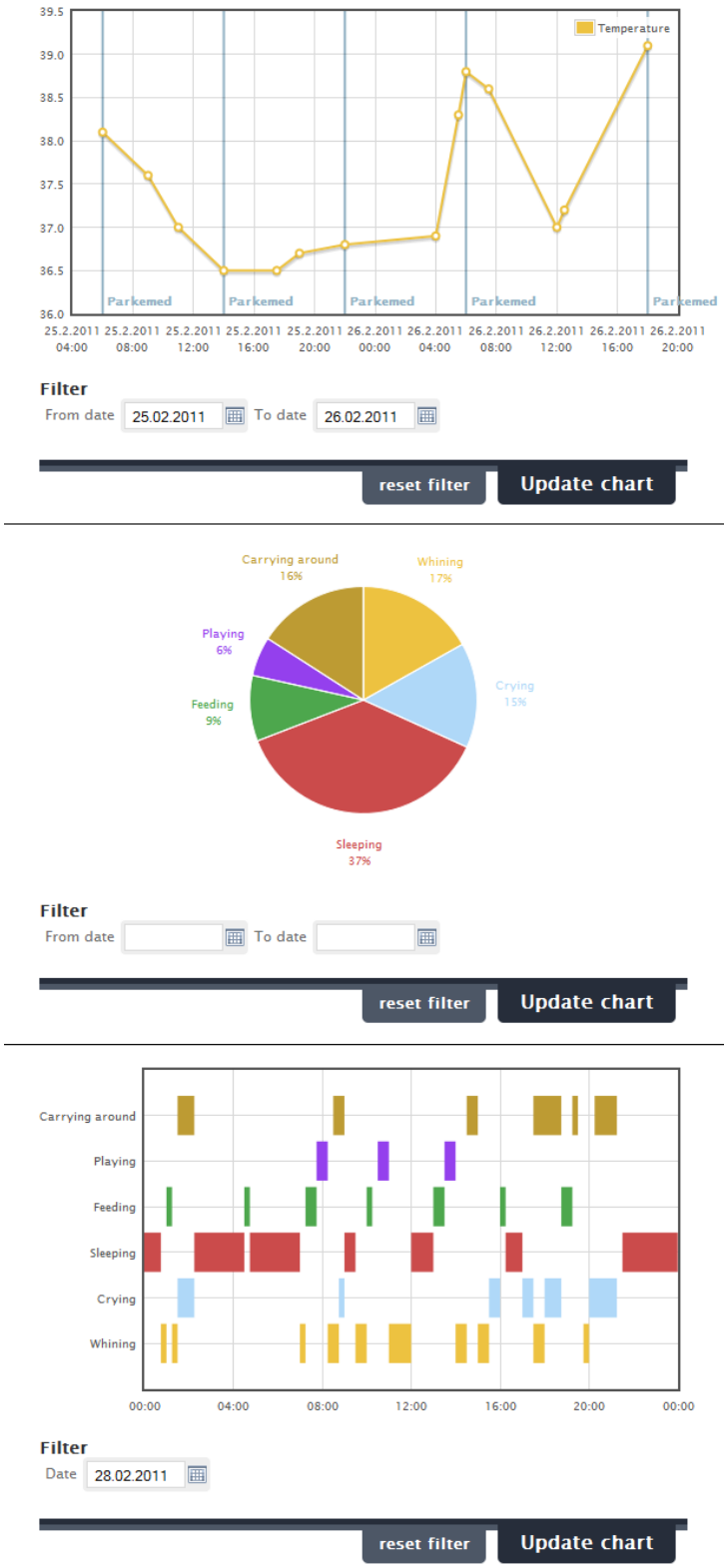


Abbildung 5.8: Visualisierung der Daten konkreter Beispiele im implementierten Prototyp.

Kapitel 6

Methoden zur Steigerung der Akzeptanz

Noch vor fünf Jahren gab es ernsthafte Zweifel, ob medizinische Einrichtungen, Versicherungen aber auch Konsumenten an der Einführung eines PHR-Systems interessiert wären. Trotz des offensichtlichen Potenzials fehlte es an konkreten Umsetzungen und den Erfahrungen daraus. Auf beiden Seiten war nicht klar, welche Kosten im Vergleich zum erwartenden Wert entstehen[6].

Bereits kurze Zeit später ließen erste Studien erkennen, dass der Einsatz von PHRs in gewissen Konstellationen durchwegs großes finanzielles Potential besitzt. Und auch das Aufkommen von frei zugänglichen Anwendungen wie Google Health, Microsoft HealthVault und UPMC HealthTrak zeigen, dass ein Business Case besteht[28, 66].

Auf der Seite der Konsumenten ist ebenfalls eine deutliche Steigerung des Interesses erkennbar. Laut einer Studie aus den Vereinigten Staaten von 2008 sind 78% der befragten Personen an einem online Zugang zu ihrer medizinischen Akte sowie zu Testergebnissen von Ärzten interessiert. 72% zeigten außerdem Interesse an einem Online-System zur Vereinbarung von Terminen[67]. Besonders Patienten mit chronischen Krankheiten, die sehr häufig in Kontakt mit dem Gesundheitssystem kommen, sind daran interessiert, auf möglichst einfache Art und Weise zu den benötigten Informationen zu gelangen[66].

Dieses Kapitel gibt einige denkbare Ansätze, um die Akzeptanz des für diese Arbeit entwickelten Systems sowohl für die Konsumenten als auch für die Serviceanbieter zu steigern.

6.1 Unterstützung der Patienten

Abgesehen von der Hilfestellung für Konsumenten, ihren eigenen medizinischen Zustand zu kennen und protokollieren zu können, würden einige zusätzliche Funktionen den Wert des Systems für die Benutzer erhöhen bzw. Unzufriedenheiten vermeiden.

6.1.1 Benachrichtigungs- und Erinnerungsfuntionen

Ein sehr einfach zu implementierendes Feature sind Benachrichtigungs- und Erinnerungsfunktionen. Solche Funktionen kommen bereits in zahlreichen PHR-Systemen vor um zum Beispiel auf die Einnahme von Medikamenten, die nächste Impfung oder einen vereinbarten Termin zu erinnern[18, 68].

In diesem konkreten Fall der Tagebücher bietet sich die Erinnerungsfunktion besonders an. Nicht alle Tagebücher kommen in kurzen, regelmäßigen Abständen vor. Einträge, die nur wöchentlich, monatlich oder sogar noch seltener getätigt werden sollen, sind leicht zu vergessen. Eine Erinnerung per E-Mail oder SMS auf das Mobiltelefon des Benutzers würden die vergessenen Einträge minimieren und somit auch die Qualität der Daten steigern.

Benachrichtigungen können für beide Benutzergruppen eingesetzt werden. Patienten könnten Benachrichtigungen von behandelnden Ärzten als Teil der Arzt-Patienten-Kommunikation erhalten. Hat der Arzt etwa Ergebnisse oder Anmerkungen für den Patienten die seine Tagebuchdaten betreffen, könnten diese als einfache Textnachrichten an den Patienten weitergeleitet werden. Die Übermittlung findet zugleich im Hinblick auf den Datenschutz in einer bereits gesicherten Umgebung statt. Benachrichtigungen für den Arzt können automatisch generiert werden sobald Tagebuchdaten einer seiner Patienten hinzugefügt wurden, was zu einer schnelleren Bearbeitung dieser führen könnte. Ein Möglichkeit für den Arzt diese automatischen Benachrichti-

gungen zu deaktivieren sollte zusätzlich davor schützen, dass der Arzt mit Nachrichten überschüttet wird.

Eine bereits implementierte Benachrichtigungsfunktion ist das Versenden von E-Mails und systeminternen Meldungen bei einer Zugriffsanfrage durch medizinisches Personal. Siehe dazu die Ausführungen aus Kapitel 4.3 bzw. in der Abbildung 4.7.

6.1.2 Integration von Wissensdatenbanken

Die Integration von Datenbanken mit medizinischem Wissen in Krankenhausinformationssystemen ist ein wichtiges Hilfsmittel für das medizinische Personal. Aber auch für Konsumenten ohne medizinischem Know-How kann eine dafür aufbereitete medizinische Wissensdatenbank hilfreich sein[69]. Bei einer Untersuchung von 91 PHR-Systemen in den USA haben bereits 49% Konsumenten-gerichtete Gesundheitsinformationen in die Anwendung integriert[26]. Die Datenbanken sind dabei zum Großteil eigene Wissensportale, die Schnittstellen für externe Systeme anbieten, wie Healthwise, MedlinePlus oder WebMD. Auch Informationen von Vereinen oder Gesellschaften die sich auf eine bestimmte Krankheit konzentrieren werden vereinzelt verwendet. Einige Systeme bieten außerdem noch eigens generierten Inhalt für ihre Benutzer an.

Die Verbindung einzelner Teile eines Tagebuchs zu den jeweiligen Inhalten einer Wissensdatenbank geschieht durch den Einsatz verschiedener medizinischer Nomenklaturen wie der ICD-10 (International Classification of Diseases and Related Health Problems) oder der SNOMED-CT (Systematized Nomenclature of Human and Veterinary Medicine - Clinical Terms). Da in dieser Arbeit openEHR verwendet wird und Werte aus den Nomenklaturen Teilen eines openEHR Archetyps zugewiesen werden können[36], ist die technische Umsetzung in erster Linie von den Webservices der Wissensdatenbanken abhängig.

6.2 Finanzielle Anreizmodelle

Eine weitere Möglichkeit, Konsumenten zur Verwendung eines PHR-Systems im Allgemeinen oder dieses elektronischen Tagebuchs im Speziellen zu motivieren, ist ein Modell das finanzielle Anreize bietet. Gerade bei Gesundheitsversicherungen wäre solch ein Modell optimal anwendbar, da bei Patienten, die sich aktiv mit ihrer Gesundheit beschäftigen ein durchschnittlich besserer medizinischer Zustand zu erwarten ist. Das führt wiederum zu geringeren Kosten die der Versicherung entstehen. Laut einer Studie der Universität Greifswald von 2007 liegt das Einsparungspotential der deutschen Krankenkassen durch die telemedizinische Überwachung von Diabetikern bei 430 Millionen Euro jährlich[70].

Solche Anreizmodelle sind in einigen Gesundheitsversicherungen bereits inkludiert. Sie versprechen finanzielle Boni nach der Begutachtung des medizinischen Zustands oder bei der Teilnahme an Gesundheitsinitiativen und Programmen zur Krankenbetreuung um sich im Falle einer chronischen Krankheit selbst versorgen zu können[71]. Besonders in den USA sind diese Methoden verbreitet. Dort wurde 2009 mit dem „Safeway Amendment" die Möglichkeit geschaffen Angestellten einen Teil der Versicherungsprämien zurück zu erstatten, wenn diese an einem Gesundheitsprogramm des Unternehmens teilnehmen[72].

Kapitel 7

Benutzerhandbuch

Beim Entwurf der Anwendung und im Speziellen des User-Interfaces wurde besondere Rücksicht auf die einfache und intuitive Handhabung durch die Patienten genommen. Trotzdem gibt es einige nicht triviale Funktionen, die einer Erklärung bedürfen. Aus diesem Grund, und um den Benutzern einen möglichst schnellen und reibungslosen Einstieg zu ermöglichen, ist ein User-Guide erstellt worden. Zusätzlich erfüllt der User-Guide auch die Funktion dem Benutzer die Möglichkeiten des Systems näher zu bringen. Dabei soll die Anleitung dem Leser aber nicht das Gefühl vermitteln, die Anwendung wäre so komplex, dass ein so umfangreicher User-Guide benötigt wird. In den folgenden Kapiteln ist ein Teil des User-Guides zu den entwickelten Funktionen zu lesen.

Der Aufbau des User-Guides in kleine, möglichst abgeschlossene Einheiten erlaubt es, aus der Anwendung heraus auf die einzelnen Teile zu verweisen und so eine direkte Verbindung zwischen einer Funktionalität und deren Beschreibung zu schaffen. Somit ist es dem Benutzer möglich, diese kontextsensitive Hilfe nur bei Bedarf in Anspruch zu nehmen.

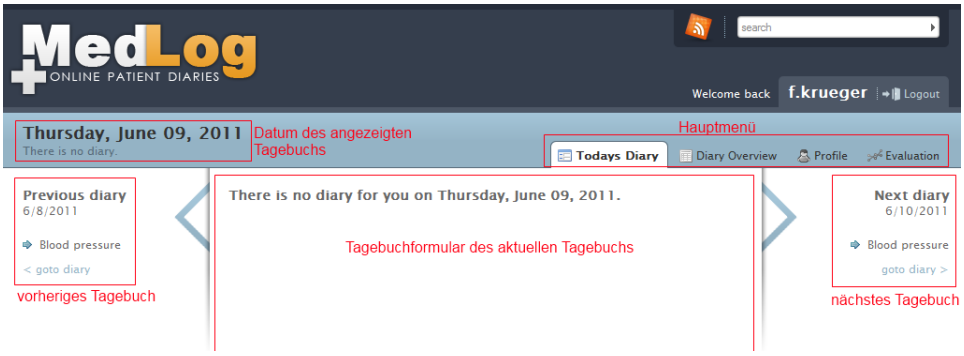
Der User-Guide kann natürlich nicht auf Inhalte der Tagebücher eingehen. Dazu ist eine zusätzliche Systemkomponente verfügbar, die es erlaubt zu den Tagebüchern frei definierbare Texte zu verknüpfen. Kapitel 7.2 bringt diese Funktionalität näher.

7.1 User-Guide

Dieser Ausschnitt aus dem User-Guide soll dessen Aufbau zeigen. Es werden die einzelnen Bereiche der Benutzeroberfläche abgearbeitet. Dabei werden die Konzepte des Systems und deren Nutzen erläutert.

Die Standardansicht

Die folgende Abbildung zeigt die Standardansicht nachdem Sie sich angemeldet haben.



Der Bereich besteht aus den folgenden Teilen:

Datum des angezeigten Tagebuchs Hier wird das Datum des angezeigten Tagebuchs angezeigt. Beim ersten Seitenaufruf nach der Anmeldung wird immer das aktuelle Datum angezeigt.

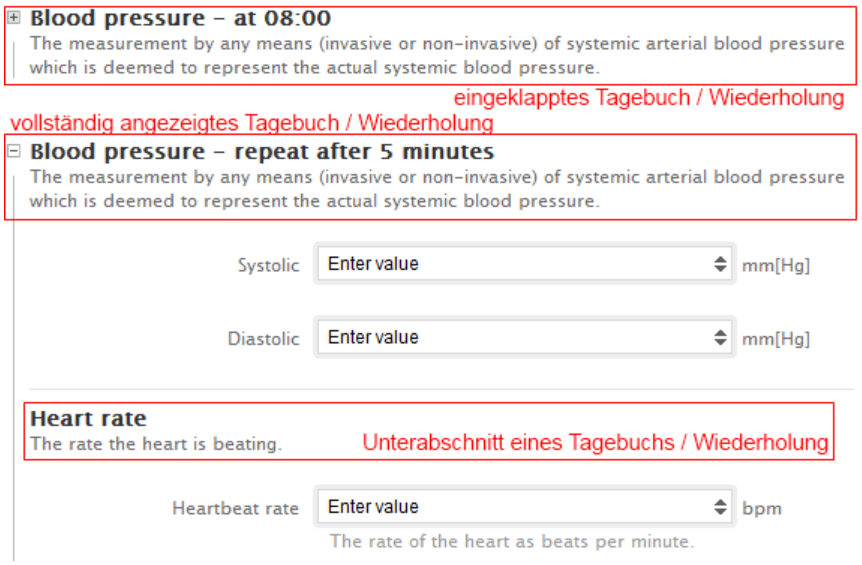
Hauptmenü Das Hauptmenü gibt Ihnen schnellen Zugriff auf die wichtigen Elemente des Systems. So gelangen Sie auf die Tagebuchansicht des heutigen Tages (Today's Diary), die Tagebuchübersicht (Diary Overview), Ihr Profil und die Auswertungen der Tagebuchdaten.

Tagebuchformular des aktuellen Tagebuchs Hier wird das Formular des Tagebuchs angezeigt. Beim ersten Seitenaufruf wird immer das aktuelle Datum angezeigt. Sollte an diesem Tag keine Eingabe vorgesehen sein, wird dies durch eine Meldung angezeigt.

Vorheriges/Nächstes Tagebuch Links und rechts des Formularbereichs finden Sie eine kurze Zusammenfassung des vorherigen bzw. des nächsten Tagebuchs. So sehen Sie, ob Sie eventuell eine Eingabe übersehen haben bzw. welche Daten und wann Sie diese das nächste Mal benötigen.

Das Tagebuchformular

Das Herzstück von MedLog ist das Formular der Tagebuchansicht. Es zeigt alle Felder aller Wiederholungen aller an diesem Tag auftretenden Tagebücher an. Die folgende Abbildung zeigt ein Formular mit einem Tagebuch und mehreren Wiederholungen:



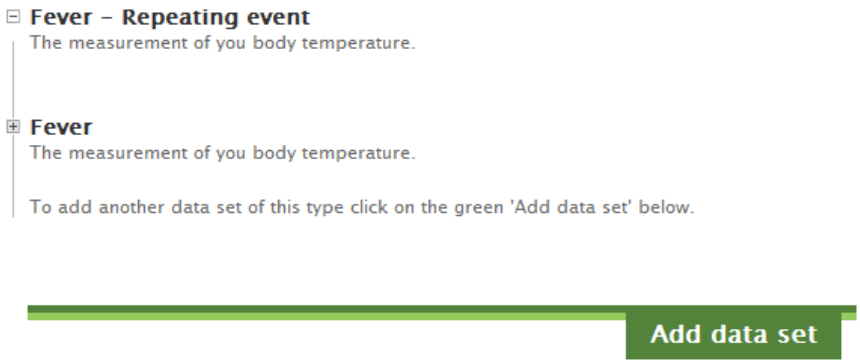
Verschiedene Tagebücher und mehrere Wiederholungen eines Tagebuchs werden im Formular gleich dargestellt und behandelt. Sie erkennen Wiederholungen daran, dass Sie den gleichen Titel tragen und ihn mit der Bezeichnung der Wiederholung ergänzen (z.B. Tagebuchtitel - um 8 Uhr). Sind sehr viele Wiederholungen oder umfangreiche Tagebücher auf der Seite angezeigt, kann es sein, dass Sie sehr viel scrollen müssten. Aus diesem Grund finden Sie neben jeder Überschrift dieser Elemente ein Steuerelement, das seine

Eingabefelder ausblendet. Achtung: Für diese Funktion ist aktiviertes Javascript notwendig. Besitzt eine Überschrift kein Steuerelement, ist es ein Unterabschnitt eines Tagebuchs. Diese können Sie nicht separat ausblenden.

Am Ende jedes Tagebuchs bzw. jeder Wiederholung befindet sich der Button „Save", mit dem die eingegebenen Daten gespeichert werden. Achtung: Es werden nur die Daten des Tagebuchs oder der Wiederholung gespeichert für die Sie „Save" drücken. Jeder andere geänderte Wert geht durch den Speichervorgang verloren.

Freie Wiederholungen

Manche Tagebücher können an einem Tag beliebig oft wiederholt werden. Diese Tagebücher bieten freie Wiederholungen. Sie sind gekennzeichnet durch die Ergänzung „Repeating event" beim Titel des Tagebuchs, sowie durch eine grüne Leiste am Ende des Tagebuchs mit einem Button „Add data set". Mit einem Klick darauf wird eine weitere Wiederholung angezeigt. Achtung: ist bereits eine leere freie Wiederholung angezeigt, wird keine neue erstellt.



Eingabefelder

Das Formular kann verschiedene Arten von Eingabefelder enthalten, die unterschiedliche Eingaben erfordern.

Textfeld Hier können beliebige Zeichen eingegeben werden. Das Feld ist eventuell in der Anzahl der Zeichen begrenzt.

Mehrzeiliges Textfeld Hier können Sie auch mehrzeilige Texte eingeben.

Zahlenfeld Dieses Feld akzeptiert nur Zahlen. Die Anzahl der Kommastellen sowie das Minimum und das Maximum des Wertes kann limitiert sein. Benutzen Sie die zwei Pfeile nach oben und unten um den aktuellen Wert zu erhöhen bzw. vermindern.

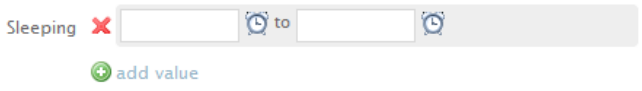
Ja/Nein-Feld Aktivieren oder deaktivieren Sie das Feld um den dementsprechenden Wert anzugeben.

Datums-/Zeitfeld Dieses Feld akzeptiert entweder ein Datum, eine Uhrzeit oder beides. Wird das Feld aktiviert, oder klicken Sie auf das Kalender- bzw. Uhr-Symbol, wird ein Auswahldialog geöffnet, der Ihnen bei der Angabe des Wertes hilft.

Auswahlfeld Wählen Sie einen Wert aus den vorgegebenen Möglichkeiten aus. Klicken Sie dazu auf das Feld und danach auf den gewünschten Eintrag.

Bereichsfeld Dieses Feld ist eine Kombination aus zwei der bereits erwähnten Feldarten, um einen Bereich angeben zu können (z.B. zwei Zahlenfelder um einen Wertebereich zu definieren, oder zwei Zeitfelder um einen Zeitbereich angeben zu können).

Mehrfach-Feld Dieses spezielle Feld erlaubt es, mehrere Werte für ein Feld anzugeben. Durch einen Klick auf „add value" wird eine weitere Eingabemöglichkeit zu diesem Feld hinzugefügt. Wollen Sie einen bestimmten Wert löschen, klicken Sie links neben dem Eingabefeld auf das rote „X". Die folgende Abbildung zeigt ein Mehrfach-Feld, das mehrere Zeitbereiche erlaubt (in diesem Fall, um Schlafzeiten zu dokumentieren).



Einheiten

Die meisten Felder besitzen entweder keine oder genau eine mögliche Einheit. Manche Felder bieten aber die Möglichkeit, zwischen mehreren Einhei-

ten auszuwählen. So können zum Beispiel Längenmaße in Meter oder Fuß angegeben werden. Werden mehrere Einheiten unterstützt, wird anstelle des Einheitentexts ein Auswahlfeld angezeigt.

Die Tagebuchübersicht

Wenn Sie im Hauptmenü auf „Diary Overview" klicken, gelangen Sie in die Tagebuchübersicht. In dieser Liste werden die einzelnen Tage aufgeführt, an denen Sie einen Tagebucheintrag vornehmen sollten. Neben einem Datum sehen Sie jeweils die Tagebücher, die für diesen Tag anstehen sowie eine kurze Beschreibung dieser. Durch einen Klick auf das Datum bzw. einen Tagebuchtitel gelangen Sie zum Formular dieses Tages. Die Einträge sind mit verschiedenen Farben gekennzeichnet, um ihren Status hervorzuheben.

Grün Die Tagebücher dieses Tages sind vollständig ausgefüllt.

Rot Keines der Tagebücher dieses Tages ist ausgefüllt.

Orange Mindestens ein Tagebuch dieses Tages ist nur teilweise ausgefüllt.

Grau Sie haben diese Tagebücher als „ignoriert" markiert.

Dunkelblau Dies sind die heutigen Tagebücher.

Hellblau Dies sind die zukünftigen Tagebücher.

Auf der linken Seite der Tagebuchübersicht können Sie bestimmen, für welchen Zeitraum Sie die Übersicht sehen möchten. Zusätzlich können Sie die Liste nach Tagen mit bestimmten Farben filtern. Deaktivieren Sie eine Tagebuchart, wird diese nach einem Klick auf den Button „update" nicht mehr angezeigt. So können Sie sich zum Beispiel alle Tage anzeigen lassen, an denen Sie vergessen haben das Formular auszufüllen.

Die Datenauswertung

Der Bereich der Datenauswertung zeigt auf der linken Seite ein Untermenü mit allen verfügbaren Darstellungen Ihrer eingegebenen Daten. Diese Darstellungen sind vordefiniert und können nicht selbst erstellt werden. Klicken

Sie auf einen der angezeigten Links und die jeweilige Aufbereitung wird angezeigt.

Manche Darstellungen wie zum Beispiel diverse Diagramme unterstützen die Möglichkeit, den Zeitraum für den die Daten angezeigt werden zu filtern. Wählen Sie dazu in den vorgesehenen Datum-Feldern den gewünschten Zeitraum aus und klicken auf „Update chart". Zusätzlich unterstützen manche Darstellungen besondere Funktionen, die die Werte der einzelnen Tage zusammenfassen. Zum Beispiel kann durch die Funktion „Average" anstelle mehrerer Werte eines Tages ein Durchschnittswert dieses Tages angezeigt werden. Diese Funktion ist vor allem bei sehr großen Zeitspannen sinnvoll, da die Änderungen innerhalb weniger Stunden meist nicht mehr relevant sind.

Ihr Benutzerprofil

Durch einen Klick im Hauptmenü auf „Profile" gelangen Sie zu Ihrem Benutzerprofil. Hier werden alle Informationen zu Ihrem Benutzerkonto angezeigt.

Um Ihre E-Mail Adresse zu ändern, geben Sie im Feld „Email" die gewünschte Adresse ein und klicken Sie darunter auf „Save".

Tagebücher verwalten

Im Abschnitt „Diaries" Ihres Benutzerkontos sehen Sie alle Tagebücher, die Ihrem Profil erfolgreich zugewiesen wurden. Für jedes Tagebuch wird eine eigene Übersicht angezeigt, die die wichtigsten Daten zusammenfasst. Darunter stehen drei Aktionen je Tagebuch zur Verfügung: Preview, Disable und Statistics.

Die Aktion „Preview" öffnet ein neues Fenster in dem eine Vorschau des Tagebuchs angezeigt wird. Ein Klick auf „Statistics" bringt Sie zur Datenauswertung dieses Tagebuchs.

Tagebuch (de)aktivieren

Ihrem Profil zugewiesene Tagebücher können deaktiviert werden. Die Formulare dieser Tagebücher erscheinen danach nicht mehr in der Standardansicht oder in der Datenauswertung. Auf diese Weise können Sie ein Tagebuch entfernen. Die Daten, die Sie bereits für dieses Tagebuch eingegeben haben bleiben erhalten.

Um ein Tagebuch zu deaktivieren, klicken Sie auf „Disable" in der jeweiligen Übersicht auf der Profilseite. Bei einem deaktivierten Tagebuch erscheint neben dem Titel ein rotes „disabled". Sie können ein deaktiviertes Tagebuch wieder reaktivieren, indem Sie auf „Enable" klicken.

Tagebuch hinzufügen

Sie können Ihrem Profil ein neues Tagebuch zuweisen. Klicken Sie dazu auf „Add a diary" im Bereich „Diaries" der Profilseite. Danach wird eine Liste mit allen Tagebüchern angezeigt, die in keine Kategorie eingeordnet sind. Am oberen Bereich sehen Sie alle Unterkategorien der aktuellen Kategorie. Klicken Sie auf die Kategorietitel, um durch die Kategorien zu navigieren.

Haben Sie das Tagebuch gefunden, dass Sie zu Ihrem Profil hinzufügen wollen, klicken Sie auf „Add this diary". Achtung: Dieser Vorgang kann nicht rückgängig gemacht werden. Ein zugewiesenes Tagebuch kann lediglich deaktiviert werden. Sind Sie sich nicht ganz sicher, ob es das richtige Tagebuch ist, benutzen Sie die „Preview" bevor Sie das Tagebuch hinzufügen.

Tagebuchzuweisung bearbeiten

Wurde Ihnen durch Ihren Arzt ein Tagebuch zugewiesen, werden Sie bei Ihrem nächsten Login aufgefordert, die Zuweisung zu akzeptieren oder zu verwerfen. Sie können, um sicher zu gehen, vorher mit einem Klick auf „Preview" eine Voransicht des Tagebuchs vornehmen.

Um die Zuweisung anzunehmen, klicken Sie auf „Confirm",um sie zu verwerfen auf „Reject". Achtung: Wenn Sie das Tagebuch zuweisen, wird dem Arzt,

der die Zuweisung erstellt hat automatisch ein Zugriffsrecht auf Ihre Daten eingeräumt.

Zugriffsberechtigungen

Damit medizinisches Personal Zugriff auf die Daten bekommt, die Sie in den Tagebüchern erstellt haben, benötigt es Ihre Berechtigung. Im unteren Bereich „Permissions" der Profilseite sehen Sie alle Benutzer, die derzeit Zugriff auf Ihr Konto besitzen. Hier können Sie jederzeit durch einen Klick auf „Reject" die Berechtigung einem Benutzer entziehen.

Wenn Sie einer Tagebuchzuweisung zustimmen, geben Sie dem zuweisenden Arzt automatisch die Zugriffsrechte. Sollte ein weiterer medizinischer Benutzer Zugriff erbeten, erscheint die Anfrage gelb hinterlegt am Ende der Liste der Berechtigungen. Klicken Sie auf „Confirm", wenn Sie dem Benutzer den Zugriff auf Ihre Eingaben erlauben wollen. Andernfalls klicken Sie auf „Reject". Achtung: Existiert für Ihr Konto eine neue Berechtigungsanfrage, werden Sie immer automatisch auf Ihre Profilseite geleitet, bis Sie die Anfrage annehmen oder ablehnen.

7.2 Tagebuchanleitungen

Um den Benutzer über spezielle Aspekte eines bestimmten Tagebuchs zu informieren, stellt die Anwendung die Möglichkeit bereit, Nachrichten mit den Tagebüchern mitzuliefern. Dazu wird über den Tagebucheditor die in HTML codierte Nachricht zu einem Tagebuch gespeichert. Durch die Verwendung von HTML ist die Nachricht vollkommen frei definierbar. Zusätzlich kann der Erzeuger der Tagebuchanleitung multimediale Objekte wie Bilder oder Videos einbinden. Auf der Oberfläche der Administration hilft ein Javascript-Steuerelement bei der Bedienung bzw. Gestaltung. Optional können Größenangaben zur Ausgabe der Meldung gemacht werden und der Administrator kann einen modalen Dialog erzwingen.

Auf der Seite des Patienten wird solch eine Meldung dann angezeigt, wenn das zugehörige Tagebuch aufgerufen wird. Dazu wird mittels Javascript ein

Fenster mit dem Inhalt der Nachricht angezeigt. Hat der Patient die Nachricht gelesen, kann er das Fenster schließen und mit der Eingabe der Tagebuchdaten fortfahren. Der Patient kann außerdem festlegen, dass in dieser Session diese Nachricht kein weiteres Mal erscheint. Abbildung 7.1 zeigt zwei beispielhafte Meldungen.

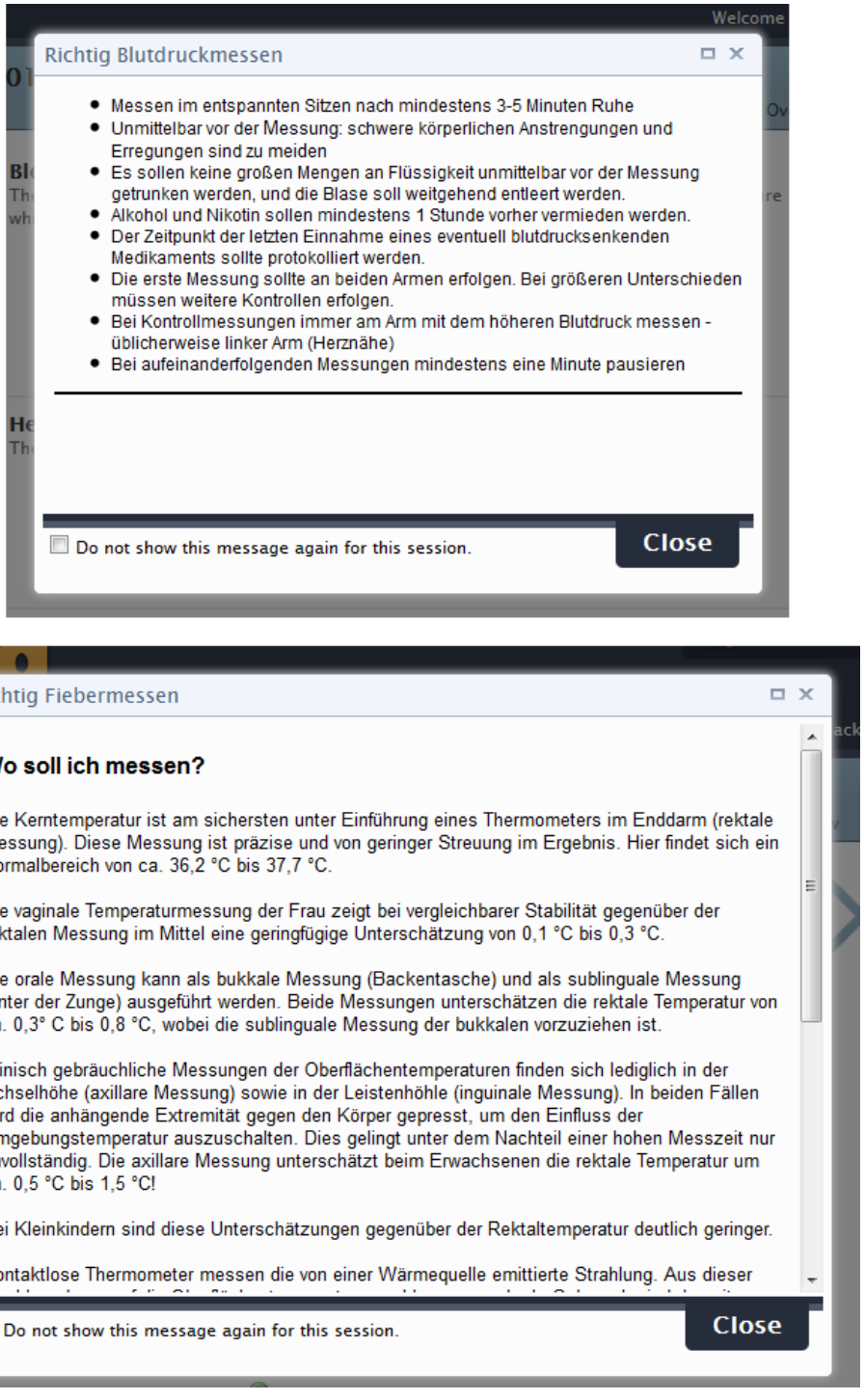


Abbildung 7.1: Zwei Beispiele von Tagebuchanleitungen als modale Dialoge.

Kapitel 8

Zusammenfassung und Ausblick auf zukünftige Aspekte und Entwicklungen

Die kontinuierlichen Entwicklungen der elektronischen sowie der persönlichen Gesundheitsakten und deren grundlegenden Technologien zeigen, dass dieser wissenschaftliche Bereich eine wichtige Rolle in der individuellen Gesundheitsversorgung darstellt. Neben den großen Gesundheitsportalen wie Google Health, Microsoft HealthVault, Indivo X etc. sind auch auf staatlicher Ebene Projekte im Gange, die die Datenhaltung medizinischer Informationen der Bürger ermöglichen sollen. Aus diesen verschiedenen Projekten sind diverse Standards, vor allem für die Kommunikation der medizinischen Daten entstanden, ohne die eine Zusammenarbeit der einzelnen Systeme mit dem Ziel, die Patienten und deren behandelnden medizinischen Einrichtungen optimal mit Informationen zu versorgen, nicht möglich wäre.

Diese Arbeit zeigt durch die prototypische Entwicklung eines elektronischen Tagebuchs eine mögliche Applikation, die durch den Einsatz eines der etablierten Standards, in der Lage ist, sich in ein Netz aus medizinischen Informationssystemen einzugliedern. Der Einsatz eines Datenmodells zur Abbildung von Archetypen, das als Teil des openEHR Standards definiert ist, erlaubt die Konfiguration verschiedenster Tagebücher die sowohl auf die

Bedürfnisse eines einzelnen Patienten als auch auf die Eigenschaften einer Krankheit eingehen.

Die Implementierung von Webservices als Kommunikationsinterface zu anderen Systemen kann die Anwendung effektiver in das entstehende Netz medizinischer Informationssysteme einbinden, indem die Daten zur Verfügung gestellt werden. So könnten Betreiber eines Krankenhausinformationssystems die Applikation anbieten und deren Daten im eigenen System nutzen, ohne die Applikation in das KIS aufwendig integrieren zu müssen. Bei der Evaluierung des bestehenden Prototypen ist diese Implementierung die wesentliche Erweiterung um die Entwicklung des Systems voranzutreiben.

Nicht nur diese zusätzlichen Zugriffe auf die sensiblen personenbezogenen Daten, auch die derzeitige Situation mit Datenabfragen ausschließlich aus dem Administrationsmodul verlangen nach einer Erweiterung des Datenschutzkonzepts. Während im entwickelten Prototyp eine Zugriffserlaubnis Zugang auf die gesamten Daten des Patienten gewährt, soll ein weiterentwickeltes System den Zugriff je Tagebuch freigeben. So kann der Benutzer genau bestimmen, welche seiner Ärzte auf welche Tagebuchdaten Zugriff erhalten. Außerdem ist es sinnvoll zumindest zwei administrative Rollen bereitzustellen:

Editor Erstellt und modifiziert FieldCollections und Tagebücher, erhält jedoch nie Zugriff zu Benutzerdaten.

MedicalUser Erhält zusätzlich zu den Rechten des Editors die Möglichkeit, Patienten ein Tagebuch zuzuweisen und Zugriffsrechte auf deren Daten zu beantragen.

Ein zusätzlicher Aspekt dieses Systems, bei dem mehrere Benutzer Zugriff auf die sensiblen Daten einer Person haben können, ist genau zu wissen, wer wann auf welche Daten zugegriffen hat. Auch für den Eigentümer des Kontos kann das unter Umständen von Interesse sein. Ein Zugriffs-Log könnte somit das Vertrauen des Benutzers in das System erhöhen.

Abgesehen von den Kommunikationsmöglichkeiten und den einhergehenden Sicherheitsüberlegungen, können zusätzliche Funktionen im System für eine noch bessere medizinische Betreuung sorgen. Wie in den Kapiteln 6.1.1

und 6.1.2 bereits ausführlich beschrieben, können einfache Entwicklungen bereits einen enormen Mehrwert bringen. Die Erinnerungsfunktion, die die Patienten durch E-Mails, Textnachrichten an ein Mobiltelefon oder andere Kommunikationskanäle an die Eingabe ihrer Daten in das aktuelle Tagebuch erinnert, kann zu einer verbesserten Qualität der Daten und somit zu einer besseren Diagnostik der Ärzte führen. Auf der anderen Seite könnten sich die Antwortzeiten der Arzt-Patienten-Kommunikation verkürzen, wenn der behandelnde Arzt über eine erfolgte Dateneingabe seines Patienten benachrichtigt werden würde. Weiters kann die Integration von verschiedenen Wissensdatenbanken nicht nur den Patienten beim Umgang mit seiner medizinischen Situation helfen, sondern auch die medizinischen Einrichtungen bei der Gestaltung und Auswertung von Tagebüchern unterstützen.

Durch die Wahl eines etablierten und mehrfach eingesetzten Standards ergeben sich noch viele weitere Möglichkeiten die Entwicklung solch eines Systems voranzutreiben. Neben kleineren Entwicklungsschritten, wie etwa die Erinnerungsfunktion, ist es aber vor allem die Kommunikation mit anderen Systemen und die Integration der Anwendung in Netzwerke zur Haltung und Verarbeitung medizinischer Daten, die die Applikation für Benutzer wertvoll gestaltet.

Anhang A

ADL Syntax des BMI

```
archetype (adl_version=1.4)
    openEHR-EHR-OBSERVATION.body_mass_index.v1

concept
    [at0000]    -- Body mass index
language
    original_language = <[ISO_639-1::en]>
description
    original_author = <
        ["name"] = <"Sam Heard">
        ["organisation"] = <"Ocean Informatics">
        ["date"] = <"22/03/2006">
        ["email"] = <"sam.heard@oceaninformatics.biz">
    >
    details = <
        ["en"] = <
            language = <[ISO_639-1::en]>
            purpose = <"To record the body mass index. This is index is
calculated by dividing the weight in kg by the height in metres squared.">
            use = <"To record BMI">
            keywords = <"obesity", "index">
            misuse = <"">
        >
    >
    lifecycle_state = <"Initial">
    other_contributors = <>

definition
    OBSERVATION[at0000] ∈ {    -- Body mass index
        data ∈ {
            HISTORY[at0001] ∈ {    -- history
```

```
                events cardinality ∈ {1..*; unordered} ∈ {
                    EVENT[at0002] occurrences ∈ {1..*} ∈ {    -- Any event
                        data ∈ {
                            ITEM_SINGLE[at0003] ∈ {    -- Single
                                item ∈ {
                                    ELEMENT[at0004] ∈ {    -- BMI
                                        value ∈ {
                                            C_DV_QUANTITY <
                                                property = <[openehr::349]>
                                                list = <
                                                    ["1"] = <
                                                        units = <"kg/m2">
                                                        magnitude = <|2.0..1000.0|>
                                                    >
                                                >
                                            >
                                        }
                                    }
                                }
                            }
                        }
                    }
                }
            }
            protocol ∈ {
                ITEM_LIST[at0005] ∈ {    -- List
                    items cardinality ∈ {0..*; ordered} ∈ {
                        ELEMENT[at0006] occurrences ∈ {0..1} ∈ {    -- Method
                            value ∈ {
                                DV_CODED_TEXT ∈ {
                                    defining_code ∈ {
                                        [local::
                                        at0007,     -- Automatic calculation
                                        at0008]    -- Direct entry
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }

ontology
    term_definitions = <
        ["en"] = <
            items = <
                ["at0000"] = <
                    description = <"The index indicating obesity">
```

```
                    text = <"Body mass index">
                >
                ["at0001"] = <
                    description = <"@ internal @">
                    text = <"history">
                >
                ["at0002"] = <
                    description = <"Any timed recording of BMI">
                    text = <"Any event">
                >
                ["at0003"] = <
                    description = <"@ internal @">
                    text = <"Single">
                >
                ["at0004"] = <
                    description = <"The index calculated from the mass
in kg divided by the square of the height in metres">
                    text = <"body mass index">
                >
                ["at0005"] = <
                    description = <"@ internal @">
                    text = <"List">
                >
                ["at0006"] = <
                    description = <"The method of calculating BMI">
                    text = <"Method">
                >
                ["at0007"] = <
                    description = <" Calculation from recorded height
and weight">
                    text = <"Automatic calculation">
                >
                ["at0008"] = <
                    description = <"Entered directly, calculated from
data not in the EHR">
                    text = <"Direct entry">
                >
            >
        >
    >
```

Anhang B

ERD/Tabellenbeschreibung

Abbildung B.1 zeigt das gesamte Datenbankmodell der Implementierung. Im Anschluss wird die Verwendung der einzelnen Tabellen kurz erläutert.

Categories Kategorien zur Strukturierung der Tagebücher. Verweis auf sich selbst, um beliebig viele Unterkategorien zu unterstützen.

CategoryDiaries Zuweisungen der Kategorien zu den Tagebüchern.

Diaries Definition der Tagebücher, die Patienten zugewiesen werden können. Die Standardfelder und das Protokoll werden als FieldCollections integriert. Werden Titel oder Beschreibung nicht angegeben, wird an dessen Stelle der gleichnamige Wert der Standard FieldCollection angezeigt.

DiaryAssignments Zuweisung eines Tagebuchs zu einem Patienten. Enthält Informationen über den zuweisenden User (StaffMember) sowie des Recurrence-Patterns. Wird die Zuweisung akzeptiert resultiert dies in einem Eintrag in PatientDataPermissions.

Evaluations Spezifikation einer grafischen oder tabellarischen Auswertung der Tagebuchangaben.

EvaluationFieldUnits Zuweisung der Einheiten eines Feldes zu einer Auswertung.

FieldCollections Zusammenschluss mehrerer Felder sowie weiterer FieldCollections (rekursiv über FieldCollectionLinks). Um eine FieldCollection einem Patienten zuweisen zu können, muss ein Tagebuch dafür erstellt werden.

FieldCollectionLinks Zuweisung von FieldCollections zu übergeordneten FieldCollections. Definiert zusätzlich die Reihenfolge.

FieldConstraints Einschränkungen von Werten in Feldern.

Fields Spezifikation der einzelnen Felder.

FieldTypes Mögliche (Daten-)Typen der Felder.

FieldUnits Einheiten der Felder.

IterationOccurrences Konkreter Tagebucheintrag für eine Iteration.

Iterations Iterationen (vergleiche mit Events in openEHR) eines Tagebuchs.

Messages Nachrichten, die mit den Tagebüchern beim Patienten angezeigt werden können.

Occurrences Vorkommnis eines Tagebuchs aufgrund der Recurrence.

OccurrenceValues Durch den Benutzer für ein Feld einer Iteration eines Tagebuchs eingegebener Wert.

PatientDataPermissions Angefragte, erteilte oder abgelehnte Zugriffserlaubnis auf einen Patienten durch ein StaffMember.

PatientDiaries Zuweisung von Tagebüchern an Patienten. Enthält zusätzlich einen Verweis auf die Recurrence.

Patients Die Patienten.

Recurrences Wiederholungsmuster der Tagebücher.

StaffMembers Das medizinische Personal.

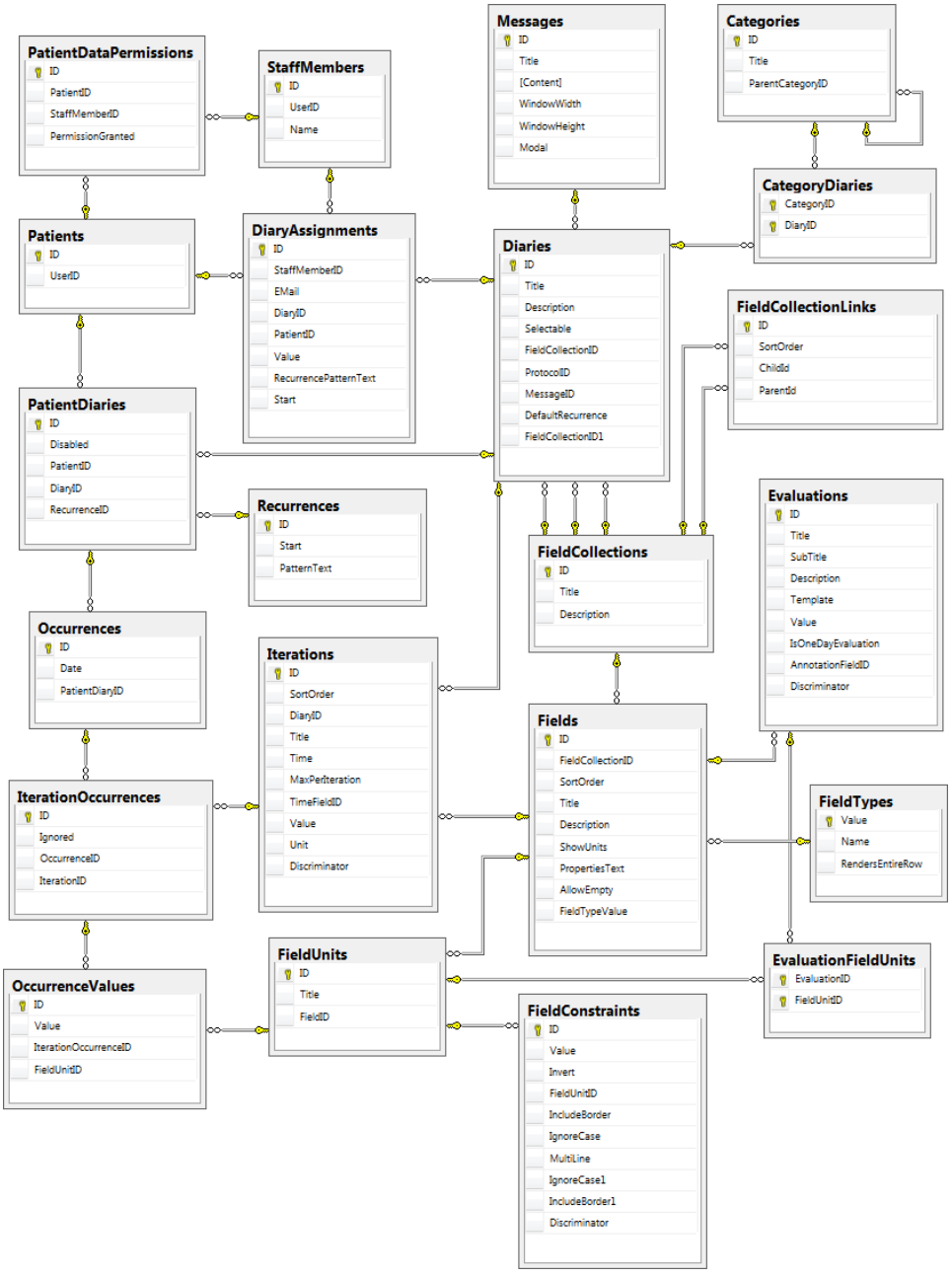


Abbildung B.1: Vollständiges ERD des verwendeten Datenmodells.

Abkürzungsverzeichnis

ADL	Archtype Definition Language
AIM	Advanced Informatics in Medicine
AM	Archetype Model
API	Application Programming Interface
BMI	Body-Mass-Index
CCOW	Clinical Context Object Workgroup
CCR	Continuity of Care Record
CSS	Cascading Stylsheet
DI	Dependency Injection
DSG	Datenschutzgesetz
EF4	Entity Framework 4
EHR	Electronic Health Record
ELGA	Elektronische Gesundheitsakte
EMR	Electronic Medical Record
ERD	Entity Relationship Diagram
FDA	US Food and Drug Administration
GDA	Gesundheitsdiensteanbieter
GEHR	Good European Health Records

GTelG	Gesundheitstelematikgesetz
HL7	Health Level 7
HL7 CDA	Health Level 7 - Clinical Document Architecture
HTML	Hypertext Markup Language
IoC	Inversion of Control
KIS	Krankenhausinformationssystem
MPG	Medzinproduktegesetz
MVC	Model View Controller
MW	Medical Workstation
NHIN	US Nationwide Health Information Network
OR-mapping	object-relational mapping
PHR	Personal Health Record
POCO	Plain Old CLR Object
PU	Patients Unit
RIA	Rich Internet Application
RIM	Reference Information Model
RM	Reference Model
SNOMED-CT	Systematized Nomenclature of Human and Veterinary Medicine - Clinical Terms
TDD	Test Driven Development
UI	User Interface
XML	Extensible Markup Language

Literaturverzeichnis

[1] Peter Haas. *Medizinische Informationssysteme und Elektronische Krankenakten*. Springer, 2005. 1.1, 2.1.1, 2.1, 2.1.2, 2.4, 2.6

[2] J. J. Cimino. Linking patient information systems to bibliographic resources. *Methods Inf Med*, 35(2):122–126, Jun 1996. 1.1

[3] Gunther Eysenbach and Christian Köhler. Health-related searches on the internet. *JAMA*, 291(24):2946, Jun 2004. 1.1

[4] Hans Oh, Carlos Rizo, Murray Enkin, and Alejandro Jadad. What is ehealth (3): a systematic review of published definitions. *J Med Internet Res*, 7(1):e1, 2005. 1.1

[5] Tom H Van De Belt, Lucien J L P G Engelen, Sivera A A Berben, and Lisette Schoonhoven. Definition of health 2.0 and medicine 2.0: a systematic review. *J Med Internet Res*, 12(2):e18, 2010. 1.1

[6] Paul C. Tang, Joan S. Ash, David W. Bates, J. Marc Overhafe, and Daniel Z. Sands. Personal health records: Definitions, benefits, and strategies for overcoming barriers to adoption. *Journal of the American Medical Informatics Association*, 13(2):121–126, 2006. 1.3, 2.2, 2.4, 6

[7] Indivo research. http://indivohealth.org/research. 1.3, 2.6.2, 4.3

[8] Marco Eichelberg, Thomas Aden, Jörg Riesmeier, Asuman Dogac, and Gokce B. Laleci. A survey and analysis of electronic healthcare record standards. *ACM Computing Surveys*, 37:277–315, Dezember 2005. 2.1, 2.5, 2.5.1, 2.5.2

[9] Ilias Iakovidis. Towards personal health record: current situation, obstacles and trends in implementation of electronic healthcare record in europe. *International Journal of Medical Informatics*, 52(1-3):105–115, 1998. 2.1

[10] F. Leiner. *Medizinische Dokumentation: Grundlagen einer qualitätsgesicherten integrierten Krankenversorgung.* Schattauer, 2003. 2.1.1

[11] Margaret K Amatayakul and Steven S Lazarus. *Electronic health records: transforming your medical practice.* Medical Group Management Association, 2005. 2.1.2

[12] Dipak Kalra and David Ingram. Electronic health records. In Krzysztof Zielinski, Mariusz Duplaga, and David Ingram, editors, *Information Technology Solutions for Healthcare*, Health Informatics, pages 135–181. Springer London, 2006. 10.1007/1-84628-141-5_7. 2.1.2

[13] The Offce of Health and the Information Highway. Toward electronic health records. Health Canada, 2001. 2.1.2

[14] C. P. Waegemann. Current status of epr development in the us. In *Toward An Electronic Health Record Europe*, 1999. 2.1.2

[15] A Rind, T Wang, W Aigner, S Miksch, K Wongsuphasawat, C Plaisant, and B Shneiderman. Interactive information visualization for exploring and querying electronic health records: A systematic review. Technical report, Human-Computer Interaction Lab, University of Maryland, http://cgis.cs.umd.edu/localphp/hcil/tech-reports-search.php?number=2010-19, September 2010. 2.1.2

[16] L.L. Weed. *Das problemorientierte Krankenblatt.* Schattauer, 1978. 2.1.2

[17] Taowei David Wang, Catherine Plaisant, Alexander J. Quinn, Roman Stanchak, Shawn Murphy, and Ben Shneiderman. Aligning temporal data by sentinel events: discovering patterns in electronic health records. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 457–466, New York, NY, USA, 2008. ACM. 2.1.2

[18] E. A. Balas, S. M. Austin, J. A. Mitchell, B. G. Ewigman, K. D. Bopp, and G. D. Brown. The clinical value of computerized information services. a review of 98 randomized clinical trials. *Arch Fam Med*, 5(5):271–278, May 1996. 2.1.2, 6.1.1

[19] P.C. Dykes. *Critical Pathways- Interdisziplinäre Versorgungspfade: DRG-Managementinstrumente*. Hans Huber Programmbereich Pflege. Huber, 2002. 2.1.2

[20] IBM Österreich GmbH im Auftrag der Bundesgesundheitsagentur. Machbarkeitsstudie betreffend einführung der elektronischen gesundheitsakte (elga) im österreichischen gesundheitswesen, Nov. 2006. 2.1.3, 2.6.1

[21] Datenschutzrat der Republik Österreich. Stellungnahme zum elektronische gesundheitsakte-gesetz (elga-g), März 2011. 2.1.3

[22] Julia Starzer. Änderung des medizinproduktegesetzes, Februar 2010. http://reloaded.wko.at/wk/format_detail.wk?AngID=1&StID=533129&DstID=3914. 2.1.3

[23] Fred Schulte and Emma Schwartz. Fda considers regulating safety of electronic health systems, Februar 2010. http://www.huffingtonpost.com/2010/02/23/fda-considers-regulating_n_474137.html. 2.1.3

[24] Henning Düwert. *Haftungsrechtliche Aspekte der Einführung elektronischer Gesundheitskarten in Deutschland und Österreich*. PhD thesis, Universität Wien, 2010. 2.1.3

[25] Christian Mauro, Ali Sunyaev, Sebastian Dünnebeil, Jan Marco Leimeister, and Helmut Krcmar. Mobile anwendungen im kontext des medizinproduktegesetzes. In Stefan Fischer, Erik Maehle, and Rüdiger Reischuk, editors, *GI Jahrestagung*, volume 154 of *LNI*, pages 1170–1182. GI, 2009. 2.1.3

[26] Dixie A. Jones, Jean P. Shipman, Daphne A. Plaut, and Catherine R. Selden. Characteristics of personal health records: findings of the medical library association/national library of medicine joint electronic

personal health record task force. *Journal of the Medical Library Association*, 98(3):243–249, 2010. 2.2, 4.5, 6.1.2

[27] Robert L. Coffield, Gerald E. DeLoss, and Gray Plant Mooty. The rise of the personal health record: Panacea or pitfall for health information. *Health Lawyers News*, 12:8–13, 10 2008. 2.2

[28] David Kaelber and Eric C Pan. The value of personal health record (phr) systems. *AMIA Annual Symposium Proceedings*, pages 343–347, 2008. 2.2, 2.4, 6

[29] Upmc fast facts, 2010. http://www.upmc.com/aboutupmc/fast-facts/Pages/default.aspx. 2.2, 12

[30] Claudia Borchard-Tuch. Piepton erinnert an medikamente. *Pharmazeutische Zeitung*, 48, 2007. 2.3, 4.4.1.1

[31] Eli O Meltzer, Norma Kelley, and Melbourne F Hovell. Randomized, cross-over evaluation of mobile phone vs paper diary in subjects with mild to moderate persistent asthma. *The Open Respiratory Medicine Journal*, 2:72–79, 2008. 2.3

[32] Florian Daniel, Fabio Casati, Patricia Silveira, Monica Verga, and Marco Nalin. Improving health, not just tracking it: A personal health and lifestyle platform. *IEEE Internet Computing*, 99(PrePrints), 2011. 2.4

[33] Andreas Triantafyllidis, Vassilis Koutkias, Ioanna Chouvarda, and Nicos Maglaveras. Mobile personal health systems for patient self-management: On pervasive information logging and sharing within social networks. In Paulo Novais, Davy Preuveneers, and Juan Corchado, editors, *Ambient Intelligence - Software and Applications*, volume 92 of *Advances in Intelligent and Soft Computing*, pages 141–148. Springer Berlin / Heidelberg, 2011. 10.1007/978-3-642-19937-0_18. 2.4

[34] David Ingram. The origins of openehr, Oktober 2002. http://www.openehr.org/about/origins.html. 2.5.1, 2.6

[35] Heather Leslie. openehr - the world's record. *Pulse+IT Magazine [online]*, 6:50–55, November 2007. 2.5.1

[36] T Beale and S Heard, editors. *openEHR Architecture Overview*. 1.0.2 edition, Nov 2008. 2.5.1, 2.3, 2.5.1, 6.1.2

[37] Thomas Beale. Archetypes: Constraint-based domain models for future-proof information systems. *OOPSLA 2002 workshop on behavioural semantics*, 2002. 2.5.1

[38] R. H. Dolin, L. Alschuler, C. Beebe, P. V. Biron, S. L. Boyer, D. Essin, E. Kimber, T. Lincoln, and J. E. Mattison. The hl7 clinical document architecture. *J Am Med Inform Assoc*, 8(6):552–569, 2001. 2.5.2

[39] Arbeitsgemeinschaft Elektronische Gesundheitsakte. Elga kernanwendungen - cda dokumente für das österreichische gesundheitswesen: Implementierungsleitfaden, Juli 2009. 2.5.2, 2.4, 2.6.1

[40] ASTM International. E2369 - 05e1 standard specification for continuity of care record (ccr), 2005. 2.5.3

[41] Sean Nolan. Again with the standards thing, Juli 2008. http://blogs.msdn.com/b/familyhealthguy/archive/2008/07/13/again-with-the-standards-thing.aspx. 2.5.3

[42] Google health data api ccr reference. http://code.google.com/intl/de-DE/apis/health/ccrg_reference.html. 2.5.3, 2.5

[43] David C Kibbe. Untangling the electronic health data exchange, Juni 2008. http://e-caremanagement.com/untangling-the-electronic-health-data-exchange/. 2.5.3

[44] George W Bush. Executive order (eo) 13335: Incentives for the use of health information technology and establishing the position of the national health information technology coordinator. *Federal Register*, 69(84):24057–24061, April 2004. 2.6.1

[45] Kenneth D Mandl, William W Simons, William C R Crawford, and Jonathan M Abbett. Indivo: a personally controlled health record for health information exchange and communication. *BMC Medical Informatics and Decision Making*, 7:25, 2007. 2.6.2, 2.6, 2.7, 4.3

[46] Daniel Haas. Indivo x: The open-source personally controlled health record platform, 2011.

http://www.oscon.com/oscon2011/public/schedule/detail/19713. 2.6.2

[47] Robert Steinbrook. Personally controlled online health data – the next big thing in medical care? *The New England Journal of Medicine*, 358(16):1653–1656, April 2008. 2.6.2, 2.6.3

[48] E. J. Gómez, M. E. Hernando, A. García, F. Del Pozo, J. Cermeño, R. Corcoy, E. Brugués, and A. De Leiva. Telemedicine as a tool for intensive management of diabetes: the diabtel experience. *Computer Methods and Programs in Biomedicine*, 69:163–177, 2002. 2.6.3

[49] Rachel Hess, Cindy L. Bryce, Kathleen McTigue, Katharine Fitzgerald, Susan Zickmund, Ellen Olshansky, and Gary Fischer. The diabetes patient portal: Patient perspectives on structure and delivery. *Diabetes Spectrum*, 19(2):106–110, 2006. 2.6.3

[50] Paul E Ruskin, Judy Can Der Wende, Cynthia R Clark, Joanne Fenton, Janie Deveau, Ramesh Thapar, Manish Prasad, and Bruce A Kehr. Feasibility of using the med-emonitor system in the treatment of schizophrenia: A pilot study. *Drug Information Journal*, 37(3):283–291, 2003. 2.6.3

[51] B. Siegmund, W. Mondorf, R. Klamroth, M. Westfeld, A. Galler, and H. Pollmann. Telemedizin in der hämophilie: Haemoassist – ein elektronisches patiententagebuch zur therapieoptimierung. *Telemedizinführer Deutschland*, pages 170–173, 2009. 2.6.3

[52] Brian Dolan. Google health redesigns, adds mobile app partners, September 2010. http://mobihealthnews.com/8904/google-health-redesigns-adds-mobile-app-partners/. 2.8

[53] Ariel Ortiz Ramirez. Three-tier architecture. *Linux Journal*, 75, 2000. 4.2.1

[54] Piero Fraternali, Gustavo Rossi, and Fernando Sanchez-Figueroa. Rich internet applications. *IEEE Internet Computing*, 14:9–12, 2010. 4.4

[55] Dennis C Neale and John M Carroll. The role of metaphors in user interface design. *Handbook of Human-Computer Interaction*, pages 441–462, 1997. 4.4.1.1

[56] Christopher Ferris and Joel Farrell. What are web services? *Communications of the ACM*, 46(6):31, Juni 2003. 4.5.1

[57] Wenchu Cen, Lin Wang, Jin Zhao, Seshu Zheng, and Yi Zeng. Method and system for providing internet services. *US Patent Application Publication*, (US 2009/0328174 A1). 4.5.2

[58] Alex Mackey. *Introducing .NET 4.0 With Visual Studio 2010.* Apress, 2010. 5.1

[59] Eilon Lipton. Asp.net mvc design philosophy, Dezember 2007. http://weblogs.asp.net/leftslipper/archive/2007/12/10/asp-net-mvc-design-philosophy.aspx. 5.1

[60] Martin Fowler. Inversion of control containers and the dependency injection pattern, Jänner 2004. http://www.martinfowler.com/articles/injection.html. 5.2

[61] Brad Wilson. Asp.net mvc 3 service location, part 5: Idependencyresolver, Oktober 2010. http://bradwilson.typepad.com/blog/2010/10/service-location-pt5-idependencyresolver.html. 5.2.2

[62] Scott Guthrie. Code-first development with entity framework 4, Juli 2010. http://weblogs.asp.net/scottgu/archive/2010/07/16/code-first-development-with-entity-framework-4.aspx. 5.3

[63] Scott Guthrie. Introducing "razor- a new view engine for asp.net, Juli 2010. http://weblogs.asp.net/scottgu/archive/2010/07/02/introducing-razor.aspx. 5.4

[64] Atanas Korchev. Cutting edge meets razor, August 2010. http://blogs.telerik.com/blogs/posts/10-08-04/cutting_edge_meets_razor.aspx. 5.5.1

[65] Rfc 5545 – internet calendaring and scheduling core object specification (icalendar), September 2009. http://tools.ietf.org/html/rfc5545. 5.6

[66] Kevin J. Leonard, Mark Casselman, and David Wiljer. Who will demand access to their personal health record? *Healthcare Quarterly*, 11(1):92–97, 2008. 6

[67] Deloitte Center for Health Solutions. 2008 survey of health care consumers: Executive summary. 6

[68] Frank Ueckert, Michael Goerz, Maximilian Ataian, Sven Tessmann, and Hans-Ulrich Prokosch. Empowerment of patients and communication with health care professionals through an electronic health record. *International Journal of Medical Informatics*, 70(2–3):99–108, 2003. MIE 2002 Special Issue. 6.1.1

[69] Deborah Beranek Lafky, Bengisu Tulu, and Thomas A Horan. Information systems and health care x: A user-driven approach to personal health records. *Communications of the Association for Information Systems*, 17, 2006. Article 46. 6.1.2

[70] DiabLink medical services. Telemedizinische Überwachung von diabetikern kann viel geld sparen, März 2007. 6.2

[71] Melinda Beeuwkes Buntin, Cheryl Damberg, Amelia Haviland, Kanika Kapur, Nicole Lurie, Roland McDevitt, and M. Susan Marquis. Consumer-directed health care: early evidence about effects on cost and quality. *Health Affairs (Millwood)*, 25(6):w516–w530, 2006. 6.2

[72] Brendan Borrell. The fairness of health insurance incentives, Jänner 2011. http://articles.latimes.com/2011/jan/03/health/la-he-health-incentives-20110103. 6.2