ulm university universität
uulm

Universität Ulm | 89069 Ulm | Germany

Fakultät für Ingenieurwissenschaften und Informatik
Institut für Datenbanken und Informationssysteme

Konzeption und Realisierung eines Sleep Monitoring Frameworks für iOS

Bachelorarbeit an der Universität Ulm

Vorgelegt von:
Andreas Reiter
andreas.reiter@uni-ulm.de

Gutachter:
Prof. Dr. Manfred Reichert

Betreuer:
Marc Schickler

2015

Fassung 22. September 2015

Satz: PDF-LaTeX 2$_\varepsilon$

Kurzfassung

Tinnitus ist eine häufig auftretende Gesundheitsstörung, deren Ursache immer noch unklar ist. Etwa 1% der Bevölkerung leidet unter diesem Problem. Mittels dieser Arbeit soll für die Tinnitusforschung ein Framework zur Verfügung gestellt werden, um Daten von Patienten während des Schlafens erheben zu können.
Dabei kommt ein iPhone zum Einsatz, mit dessen Sensoren Bewegungen und Geräusche aufgezeichnet werden. Im Anschluss daran werden die ermittelten Daten aufbereitet und analysiert. Die daraus gewonnen Erkenntnisse stehen dann der eingesetzten mobilen Anwendung zur Verfügung.
Im Zuge dieser Bachelorarbeit wird beispielhaft eine Anwendung umgesetzt, die den Einsatz des Frameworks ermöglicht und die Ergebnisse der analysierten Daten präsentiert. Die Anwendung und das Framework sind allgemein für iOS-basierte Plattformen entwickelt.

Inhaltsverzeichnis

1

Einleitung

In der industrialisierten Welt sind Schlafstörungen sehr weit verbreitet und führen zu einer Beeinträchtigung der Lebensqualität. Um dieses Krankheitsbild zu untersuchen hat sich die Methode der Polysomnografie (PSG) als Standard etabliert, um Schlaf zu bewerten und zu analysieren. Dieses Vorgehen setzt jedoch auf eine professionelle Schlafüberwachung in Schlaflaboren. Die PSG dauert mehrere Nächte und benötigt zudem eine teure Ausrüstung [Nat12].
Auch im Bereich der Tinnitusforschung kann diese Technik Anwendung finden, da eine mögliche Ursache dieses Leidens im Schlaf liegen könnte. Unter Tinnitus ist die Wahrnehmung eines Geräusches in Abwesenheit eines entsprechenden externen Stimulus zu verstehen. Dabei ist Tinniuts eine häufige Gesundheitsstörung die etwa 1% der Bevölkerung schwer beeinträchtigt. Derzeit kommen Fragebögen zum Einsatz, um den Schlaf subjektiv zu beurteilen. Diese Angaben könnten mittels einer Schlafüberwachung überprüft und validiert werden, um eine objektive Bewertung zu erhalten [Kre13] [SSPR15].

Der heutige Stand der Technik und die Verbreitung von Smartphones bieten die Möglichkeit, diverse Gesundheitsdaten von Nutzern zu erheben. So sind die meisten Geräte mit verschiedenen Sensoren ausgestattet, wie zum Beispiel einem Gyroskop, Beschleunigungssensor und Mikrofonen. Mit diesen Bauteilen lässt sich die Umgebung von Smartphones technisch erfassen und messen.
Ein alternatives Vorgehen zur PSG könnte demnach die Schlafüberwachung unter zu Hilfenahme eines Smartphones darstellen. Dazu soll ein Framework entwickelt werden mit dem es möglich ist Daten während des Schlafens zu erheben. Dieses Framework wird in einer mobilen Anwendung eingesetzt, um die Funktionsweise zu demonstrieren. Mit diesem Vorgehen kann eine günstige Alternative geschaffen werden, um Daten zum Schlafverhalten zu erhalten.

1.1 Ziel der Arbeit

Das Ziel dieser Arbeit stellt die Entwicklung eines Frameworks dar, mit dem es möglich ist die Sensorik eines Smartphones zu nutzen, um Bewegung und Geräusche aufzuzeichnen. Der verbaute Beschleunigungssensor ist für das Erkennen von Bewegungen zuständig, indem die jeweilige Beschleunigung der verschiedenen Achsen aufgezeichnet wird. Die Geräusche sollen über das Mikrofon, welches auch für Telefonie oder andere Spracheingaben eingesetzt wird, aufgenommen werden. In diesem Zusammenhang wird lediglich die Lautstärke berücksichtigt, um festzustellen, ob Geräusche während der Aufnahme auftreten. Smartphones haben den Vorteil, dass sie handlich und mobil einsetzbar sind, so kann beispielsweise ein iPhone direkt im Bett platziert werden, um die Bewegungen anhand der Veränderungen auf der Matratze zu erkennen, ohne zusätzliche Hardware einzusetzen. Anhand dieser Daten sollen aufgezeichnete Nächte beziehungsweise Schlafzeiten analysiert werden, um Aussagen über das Schlafverhalten treffen zu können. Dabei spielen unter anderem der Einschlafzeitpunkt und die Effizienz des Schlafes eine wichtige Rolle. Der Einsatz des Frameworks erfolgt im Zuge dieser Bachelorarbeit in einer Beispielanwendung.
Grundsätzlich ist das Framework so aufgebaut, dass alle Funktionen zu Schlafaufzeichnung auch für andere Anwendungen verfügbar sind. So ist es möglich zum Beispiel

für den Bereich der Tinnitusforschung eine Anwendung zu entwickeln, die die bereits erwähnten Fragebögen abbildet und zusätzlich zur Schlafüberwachung des Patienten verwendet werden kann, um subjektive als auch objektive Daten zu erhalten.

1.2 Aufbau der Arbeit

Zunächst wird auf die nötigen Grundlagen (Kapitel 2), wie unter anderem die verwendete Programmiersprache eingegangen. Darauf folgt eine Anforderungsanalyse (Kapitel 3) für die Anwendung und das Framework, die sowohl die funktionalen, als auch die nicht funktionalen Anforderungen festhält. Im nächsten Abschnitt wird auf den Entwurf (Kapitel 4), der als Grundlage der nachfolgenden Implementierungsarbeit (Kapitel 5) dient, näher eingegangen. Dabei werden die beiden Bestandteile getrennt von einander entwickelt und anschließend wird das Framework in die Anwendung eingebettet. Nach der Umsetzung folgt ein Abschnitt, indem näher auf Tests (Kapitel 6) der fertigen Anwendung eingegangen wird. Der Anforderungsabgleich (Kapitel 7) stellt den nächsten Teil der Arbeit dar. Hierbei wird beschrieben, in welchem Maße und Umfang die zuvor getroffenen Anforderungen umgesetzt werden konnten. Auch hier wird wiederum die Anwendung separat vom Framework betrachtet und ein Fazit gezogen. Abschließend erfolgt noch eine Zusammenfassung und ein Ausblick (Kapitel 8) des Projekts. Die nachfolgende Übersicht (siehe Abbildung 1.1) stellt den Aufbau dieser Arbeit nochmals grafisch dar.

Abbildung 1.1: Übersicht - Aufbau der Arbeit

2

Grundlagen

Dieses Kapitel befasst sich mit den Grundlagen, auf denen das Konzept und die Umsetzung des Frameworks und der dazugehörigen Anwendung beruhen. Zunächst werden plattformspezifische Besonderheiten angeführt, die die Wahl von iOS als Basis mit sich bringt. Darunter fallen sowohl die Programmiersprache als auch die dazugehörige Entwicklungsumgebung für die Überwachungsanwendung und das Framework. Darüber hinaus werden noch grundlegende Punkte zur Schlafüberwachung zum Vorgehen bei der Datenerhebung aufgegriffen .

2.1 Plattform

Wie bereits erwähnt wurde als Plattform iOS von Apple verwendet. Um für diese Plattform eine Anwendung beziehungsweise ein Framework erstellen zu können, ist ein

Entwickleraccount von Apple erforderlich. Zudem muss jedes Gerät mit seiner Seriennummer registriert werden, um dort eine Anwendung, die sich in der Entwicklung befindet, zu installieren.

2.2 Entwicklungsumgebung

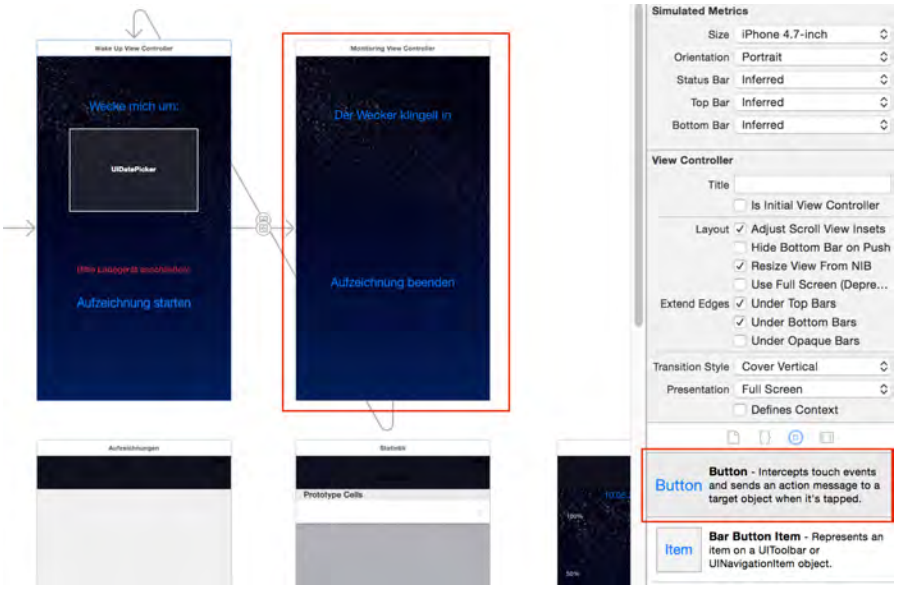

Abbildung 2.1: Xcode Interfacebuilder - Übersicht

Als Entwicklungsumgebung wurde Xcode (Version 6.4) von Apple verwendet [App15g]. Diese Umgebung ist die empfohlene Standardentwicklungsumgebung von Apple und ist daher optimal für die Anwendungs- und Frameworkentwicklung auf iOS-Basis geeignet. Eine Besonderheit von Xcode stellt der integrierte Interfacebuilder dar. Dieser bildet eine komplette Übersicht über jede Ansicht der Anwendung sowie deren Beziehungen untereinander grafisch ab. Darüber hinaus können Interface-Elemente mittels Drag and Drop mit der dazugehörigen Anwendungslogik verbunden werden.
Die Abbildung 2.1 zeigt einen Ausschnitt des erläuterten Interfacebuilders in Xcode. Mittig sind die einzelnen Ansichten und auch die Verbindungen zueinander ersichtlich.

Auf der rechten Seite werden Detailinformationen zu einer Ansicht aufgelistet und es sind Elemente (Beispiel: Button) zu erkennen, die für die Anwendung eingesetzt werden können.

2.3 Swift

Als Programmiersprache wurde die noch relativ neue Sprache Swift verwendet, um das Framework und die Anwendung zu implementieren. Swift ist eine von Apple eigens entwickelte Sprache zur Erstellung von Anwendungen für iOS basierte Geräte und Mac-Programme [App15f].
Auf Grund der Aktualität und der noch geringen Verbreitung von Swift, soll nachfolgend auf ein paar Besonderheiten dieser Programmiersprache eingegangen werden: Swift ist statisch typisiert, objektorientiert, imperativ und teilweise funktional. Darüber hinaus besteht die Möglichkeit Tupel zu verwenden. Dadurch können Funktionen zum Beispiel gleichzeitig mehrere Werte zurückliefern. Es existiert ein Datentyp „Any“ mit dem es möglich ist, alle anderen Datentypen aufzunehmen. Des weiteren werden Funktionen in Swift als Objekte angesehen, die sich auch als solche benutzen lassen. Dabei kann man Funktionen Variablen zuweisen und sie später über diese ansprechen [Chr14].

```
1 func saveRecordedDatatoFile(data: [(MotionData, AudioData)]){
2   if calcMeanValue {
3      var mean = meanRecordedData(data)
4      self.fileMannager?.writeToFile(...)
5   }
6   else {
7      for (m,a) in data {
8         ...
```

Listing 2.1: Swift - Codebeispiel

Der Ausschnitt aus dem Listing 2.1 zeigt Programmcode der eben angesprochenen Programmiersprache Swift. Hierbei ist die Verwendung von Tupeln in Zeile 1 zu erken-

nen. Die Variable„data“ wird als Array bestehend aus Tupeln vom Typ „MotionData“ und „AudioData“ deklariert. Anschließend wird in der 7. Zeile über diese Variable in einer for-Schleife iteriert. Dabei können die Elemente (hier: (m,a)) des Tupels einzeln angesprochen werden.

2.4 Schlafüberwachung

Dieser Abschnitt beschreibt die nötigen Grundlagen zur Schlafüberwachung, die für eine Überwachung mittels Smartphone relevant sind. Zunächst werden dazu die Möglichkeiten beschrieben, wie Messungen erfolgen können. Im Anschluss daran wird näher auf die Messwerte und deren Bedeutung eingegangen.
In dieser Arbeit soll eine für Smartphone angepasste Version der Aktigraphie verwendet werden. Das Prinzip der Aktigraphie besteht darin, mittels Sensoren Bewegungen zu messen. Somit sollen Rückschlüsse anhand der Häufigkeit und Intensität der Bewegungen auf den Schlaf ermöglicht werden. Dabei spielt auch die Einschlafzeit eine wichtige Rolle [Jut10].

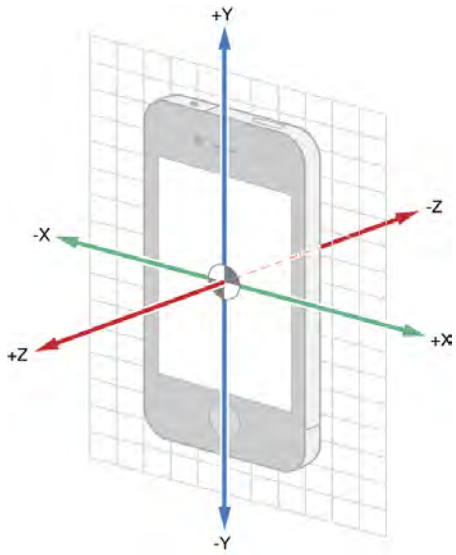

Abbildung 2.2: Beschleunigungssensor - Schema der jeweiligen Achsen [App15b]

Die Bewegungen werden mit dem Beschleunigungssensor erfasst. Dieser nimmt die Beschleunigung oder auch g-Kräfte, die auf das Gerät wirken, an allen drei Achsen auf. Die schematische Abbildung 2.2 soll nochmals veranschaulichen, wie die unterschiedlichen Achsen vom Beschleunigungssensor erfasst werden.
Mit diesem Sensor lassen sich Beschleunigungswerte von -2 g bis +2 g mit einer Genauigkeit von 0,018 g erkennen [Max15]. Zudem kann mit einer maximalen Frequenz von bis zu 100 Hz abgetastet werden. Für die Schlafüberwachung sind diese technischen Werte mehr als ausreichend. In diesem Projekt sind Messungen mit einer maximalen Frequenz von 1 Hz angestellt worden [App15b].
Für die Interpretation der gemessenen Werte ist nur die Gesamtbeschleunigung nötig. Diese lässt sich über folgende Formel ermitteln. Beim z-Wert muss noch 1 addiert werden, da bei einem liegenden iPhone mit dem Bildschirm nach oben die Erdbeschleunigung von -1 g in z-Richtung wirkt [Alv14].

$$\sqrt{x^2 + y^2 + (z+1)^2} \tag{2.1}$$

Anhand dieser Werte werden statistische Methoden und Berechnungen angestellt um den Bewegungsverlauf der Nacht zu beschreiben und Erkenntnisse über den Schlaf zu erhalten. Diese Verfahren sind im Kapitel 5 näher erläutert.
Die Aufzeichnung von Geräuschen ist nur informativ und soll die Bewegungsanalyse vervollständigen. Für diese Aufnahmen wird das interne Mikrofon des Gerätes ausgelesen und der so ermittelte Dezibel-Wert mit der Bewegung abgespeichert. Das Mikrofon kann einen Wertebereich von 0 dB bis -160 dB aufnehmen. Dabei bedeutet 0 dB die Maximallautstärke und -160 dB repräsentiert das Minimum (Stille) [App15a]. Wie bei der Bewegungsaufnahme wird die Lautstärke maximal mit einer Frequenz von 1 Hz abgetastet.

3

Anforderungsanalyse

Dieses Kapitel befasst sich mit der Analyse von funktionalen sowie nicht funktionalen Anforderungen. Dazu werden zunächst die Anforderungen der Schlafüberwachungsanwendung betrachtet und im Anschluss die des Frameworks. Zu den funktionalen Anforderungen zählen all jene Punkte, die wesentlich zum Funktionsumfang der beiden Teile beitragen. Im Gegensatz dazu sind im nicht funktionalen Bereich die Anforderungen zu finden, die die Programmeigenschaften betreffen.

3.1 Anwendung

Im folgenden Unterabschnitt wird näher auf die Anforderungsanalyse der eigentlichen Anwendung SleepMonitor eingegangen. Als ersten Schritt werden bereits bestehende

Produkte aufgegriffen und analysiert. Im Anschluss daran erfolgt die Definition der funktionalen sowie nicht funktionalen Anforderungen.

3.1.1 Analyse bestehender Anwendungen

Um einen Überblick und Anregungen für die zu entwickelnde Anwendung zu erhalten werden zunächst zwei bereits existierende Produkte betrachtet. Daraus lassen sich erste Ideen für Anforderungen ableiten. Beide Programme sind im AppStore für das iPhone erhältlich und können von dort heruntergeladen und installiert werden. Leider geben die Hersteller keine genaueren internen oder technischen Details über ihre mobilen Anwendungen bekannt.

WakeApp

Abbildung 3.1: Screenshots - WakeApp

Zunächst wurde die Anwendung WakeApp betrachtet [App15h]. Die Anwendung verspricht ein besseres Aufwachen, da versucht wird den Weckzeitpunkt in eine Schlafphase

mit leichtem Schlaf zu legen. Dabei werden Bewegungen und auch Geräusche aufgenommen. Die Abbildung 3.1 zeigt zwei Screenshots der Anwendung. Dabei fällt zunächst die Statistik mit den Durchschnittswerten auf. Dadurch kann der Benutzer seinen Schlaf über eine Nacht hinaus betrachten und zu allgemeineren Erkenntnissen gelangen. Der rechte Bildausschnitt zeigt eine einzelne Analyse. Hier wird die Bewegungs- und die Geräuschaufnahme separat dargestellt. Darüber hinaus wird die Schlafqualität analysiert.

Sleep Cycle

Abbildung 3.2: Screenshots - Sleep Cycle

Die mobile Anwendung Sleep Cycle ist die zweite analysierte Anwendung [Nor15] . Nach der Installation und Verwendung für mehrere Nächte sind nachfolgende Eigenschaften für die Analyse erwähnenswert. Wie in Abbildung 3.2 dargestellt, ist die Anwendung sehr übersichtlich gestaltet und auch wie in der zuvor beschriebenen Anwendung wird eine horizontale Tabbar eingesetzt. Es werden statistische Angaben zur einer einzelnen Aufnahme angezeigt. Darüber hinaus erfolgen auch Angaben für die Nacht mit dem Besten beziehungsweise Schlechtesten Schlaf. Gerade für die erstmalige Verwendung

bietet die Anleitung eine wichtige Hilfestellung. Dadurch lässt sich die Verwendung der Anwendung und ihre Funktionen sowie auch die richtige Positionierung des Gerätes schnell erlernen. Die Anleitung wird im rechten Teil der Abbildung 3.2 dargestellt.

Zusammenfasssung

Der folgende Abschnitt soll nochmals anhand einer Tabelle 3.1 die wichtigsten Eigenschaften der beiden mobilen Anwendungen zusammengefasst werden.

Eigenschaft	WakeApp	Sleep Cycle
Übersichtlichkeit	Es sind teilweise zu viele Datenwerte in einer Ansicht. Auch die häufige Verwendung von Diagrammen mindert die Übersichtlichkeit.	Die Statistik ist etwas zu überladen mit Daten. Für erweiterte Informationen muss das iPhone gedreht werden.
Design	Es werden viele Farben verwendet, um einzelne Elemente zu gestalten. Das restliche Design ist in schlichten Grautönen gehalten.	Einheitliches und passendes Design, welches für die nächtliche Umgebung ausgelegt ist.
Datenerhebung	Geräusche und Bewegungen werden aufgenommen.	Geräusche und Bewegungen werden aufgenommen.
Statistik	Viele Details und sehr ausführliche Statistiken. Es können Zeiträume für die Auswertung festgelegt werden.	Ausführliche Grafiken und Analysen über mehrere Tage hinweg.
Hilfestellung	Eine ausreichende Anleitung beziehungsweise Hilfe wird angeboten.	Gut beschriebene und ausführliche Anleitung mit grafischen Elementen zur Unterstützung. Auch eine Testfunktion wird dort angeboten.

Tabelle 3.1: Übersicht - Vergleich von WakeApp und Sleep Cycle

3.1.2 Funktionale Anforderungen

Dieser Unterabschnitt befasst sich mit der Darlegung der nötigen funktionalen Anforderungen, die an die Schlafüberwachungsanwendung gestellt werden. In der nachfolgenden Tabelle 3.2 sind die jeweiligen Anforderungen aufgeführt.

Nr.	Anforderung	Beschreibung
I	Weckerfunktion	Auf Grund der Überwachung des Schlafes, soll es möglich sein, dass der Benutzer eine Weckzeit angeben kann zu der die Aufnahme und auch der Schlaf enden soll
II	Weckerfunktion - Gerät lautlos oder im Standby-Modus	Werden über einen längeren Zeitraum keine Eingaben getätigt wechseln Smartphones normalerweise in den Standby-Modus. Auf Grund dieser Tatsache soll auch in diesem Zustand der Wecker funktionieren. Für den Fall, dass das iPhone auf lautlos gestellt ist, soll der Alarm trotzdem gespielt werden, um eine zuverlässiges Wecken zu gewährleisten.
III	Starten der Aufnahme	Mit Bestimmen eines Weckzeitpunktes kann die Aufnahme gestartet werden. Auch hier gilt, die Aufnahme soll während des gesamten Zeitraumes aktiv bleiben, auch wenn sich das Gerät im Standby-Modus befindet. Zusätzlich soll eine Prüfung stattfinden, ob das Gerät mit einer Stromquelle verbunden ist, um die Aufnahme sicher gewährleisten zu können.
IV	Beenden der Aufnahme	Sobald eine Aufnahme gestartet wird, muss es auch wieder möglich sein diese zu beenden. Das Beenden soll sowohl manuell möglich sein, als auch automatisch nach Ablauf des Überwachungszeitraumes statt finden. Nach Ablauf der Überwachungsdauer muss die Weckfunktion aktiv werden, bei manuellem Beenden ist dies nicht nötig.

V	Übersicht aller Aufnahmen	Der Anwender soll seine bisherigen Aufnahmen einsehen können. Dabei müssen die einzelnen Aufzeichnungen aussagekräftig beschriftet sein. Darüber hinaus sollte eine chronologische Reihenfolge eingehalten werden.
VI	Details einer Aufnahme	Einen Kernpunkt stellt die Detailansicht einer gewählten Aufzeichnung dar. Dort soll ersichtlich sein, wie der Schlafverlauf während des Zeitraumes von statten gegangen ist. Hier soll auch eine grafische Übersicht des Schlafverlaufes sowie nähere Werte der Aufzeichnung dargestellt werden.
VII	Statistiken	Als zusätzlichen Punkt soll eine Gesamtstatistik über alle Aufzeichnung vorhanden sein. Somit kann vom Anwender ein Trend über seine bisherigen Aktivitäten eingesehen werden. Hierbei können Mittelwerte, längste und kürzeste Nacht und auch beste beziehungsweise schlechteste Nacht (Schlafeffizienz) aufgeführt sein.

Tabelle 3.2: Funktionale Anforderungen der Anwendung

3.1.3 Nicht funktionale Anforderungen

Neben den funktionalen Anforderungen, ist es auch wichtig die nicht funktionalen zu analysieren. Dieser Abschnitt stellt alle Punkte dar, die Eigenschaften der SleepMonitor-Anwendung beschreiben. Die Anforderungen sind in Tabelle 3.3 aufgelistet.

Nr.	**Anforderung**	**Beschreibung**
I	Robustheit	Unter Robustheit sollen gerade die unterschiedlichen Zustände der verwendeten Geräte beachtet werden. Da beispielsweise ein iPhone wenn es sich im Ruhezustand befindet, keine Töne abspielt und auch die Aufnahme unterbricht.

II	Zuverlässigkeit	Einen wichtigen Punkt stellt die Zuverlässigkeit dar. In Bezug auf die Aufzeichnung und der Weckerfunktion muss sichergestellt sein, dass diese reibungslos funktionieren. Gerade die Konsequenzen eines funktionslosen Weckers wären fatal für einen Anwender und somit auch für die Anwendung selbst.
III	Individualisierbarkeit	Eine Anwendung soll Anpassungsmöglichkeiten an die Bedürfnisse des Anwenders bereit stellen, um den Komfort und somit die Benutzung zu verbessern. Gerade der Weckton ist ein sensibler Bereich, der unbedingt an die Vorlieben angepasst werden sollte. Auch die Lautstärke dieses Tons sollte individuell einstellbar sein.
IV	Hilfestellung	Im Normalfall sollte eine mobile Anwendung möglichst intuitiv verwendbar sein, da aber bei der Schlafaufzeichnung die Positionierung des Gerätes im Bett eine wichtige Rolle spielt, sollte dies für den Anwender beschrieben sein. Damit kann eine optimale Verwendung der Anwendung erfolgen.

Tabelle 3.3: Nicht funktionale Anforderungen der Anwendung

3.2 Framework

Dieser Teilabschnitt umfasst ähnlich zu Abschnitt 3.1 die Anforderungen, die an das Framework zur Schlafüberwachung gestellt werden. Hierbei wird wieder zuerst auf die funktionalen Anforderungen gefolgt von den nicht funktionalen, eingegangen.

3.2.1 Funktionale Anforderungen

In diesem Teilabschnitt werden nun alle Anforderungen, die das Framework zur Schlafüberwachung aufweisen muss, definiert.

Tabelle 3.4 gibt eine Übersicht über alle nötigen funktionalen Anforderungen des Frameworks.

Nr.	Anforderung	Beschreibung
I	Datenerhebung	Eine wesentlichen Bestandteil des Frameworks soll die Datenerhebung ausmachen, da diese essenziell für jegliche Aufzeichnungen sind. Dabei sollen zwei verschiedene Datenarten erhoben werden. Zum einen Bewegungsdaten, die über den Beschleunigungssensor ermittelt werden, und zum anderen Audiodaten, die vom integrierten Mikrofon stammen. Außerdem muss die Möglichkeit bestehen, festzulegen in welchen Intervallen die Daten abgegriffen werden.
II	Aufbereitung der Rohdaten	Die erhobenen Daten stammen direkt von den Sensoren und sollen für ihre weitere Verwendung (Speicherung, Analyse) aufbereitet werden. Der Beschleunigungssensor liefert zum Beispiel Daten für jede Achse des Gerätes, die zusammengefasst werden sollten.
III	Speicherung der Daten	Das Speichern von Daten ist ein weiterer Punkt, der unverzichtbar für das Aufzeichnungsvorhaben ist. Eine geeignete Datenstruktur und Dateiformat muss dazu gewählt werden. Auch die Festlegung eines Speicherorts muss in diesem Zuge erfolgen.
IV	Datenanalyse	Nachdem die nötigen Daten erhoben wurden und aus Sicht des Frameworks richtig abgelegt sind, stellt die Datenanalyse den nächsten Schritt, dar. Dabei muss auf die Daten zugegriffen werden können und für die jeweiligen Analysefunktionen zur Verfügung stehen. Ausgewertet sollte unter anderem Schlafdauer, Einschlafzeitpunkt und die ermittelte Schlafeffizienz.

Tabelle 3.4: Funktionale Anforderungen des Framework

3.2.2 Nicht funktionale Anforderungen

Den letzten Teilbereich der Anforderungsanalyse stellen die nicht funktionalen Anforderungen des Frameworks, dar. Dabei sollen wie auch schon in Abschnitt 3.1, die nötigen Eigenschaften festgelegt werden. Die nachfolgende Tabelle 3.5 enthält alle nicht funktionalen Anforderungen.

Nr.	Anforderung	Beschreibung
I	Wiederverwendbarkeit	Dieser Punkt soll zum einen eine gute Struktur des Quellcodes, sowie eine unnötige Verwendung von redundanten Code verlangen. Da dieses Framework nicht nur für die Anwendung die im Zuge dieses Projekts erstellt wird geeignet sein sollte, muss der Quellcode dementsprechend strukturiert werden.
II	Datenkapselung	Hierbei sollten Programmteile des Frameworks, die nicht von einer Anwendung aus zugreifbar sind, entsprechend gekapselt werden. Gerade kritische interne Methoden und Attribute sollten vor einem Zugriff von außen geschützt sein.
III	Aussagekräftige Methodennamen	Da das Framework im Anwendungskontext eingesetzt wird, erfolgen Zugriffe auf dessen Funktionalitäten. Daher sollen die Methodennamen so gewählt werden, damit Entwickler deren Zweck gut erkennen. Dies soll den Umgang mit dem Framework einfacher machen und den Anwendungscode leserlich halten.
IV	Kommentare zur Steigerung der Leserlichkeit	Ein wichtiger Zusatz zum eigentlichen Quellcode sind Kommentare, gerade wenn andere Entwickler mit dem Framework arbeiten. Auch für den Fall von Anpassungen oder Erweiterungen sind Kommentare äußerst hilfreich und erleichtern diese Tätigkeiten deutlich.

Tabelle 3.5: Nicht funktionale Anforderungen des Framework

4

Entwurf

Dieses Kapitel beschreibt den Entwurf der Anwendung und auch des Frameworks näher. Im Entwurf sollen die ersten Konzepte und Ideen für die Umsetzung des Projekts festgehalten werden. Im Bereich der Anwendung kommen Mockups zum Einsatz, die das grobe Aussehen und das Bedienkonzept beschreiben. Für das Framework ist ein Klassendiagramm sinnvoll, da keine Benutzeroberfläche existiert und nur Klassen und deren Beziehungen im Fokus stehen. Im Entwurf muss auf die zuvor definierten Anforderungen Rücksicht genommen werden.

4.1 Anwendung

Auch in diesem Abschnitt wird die Anwendung getrennt vom Framework betrachtet, um auf die einzelnen Teile unabhängig von einander betrachten zu können. Es wird

wiederum mit den Entwufsüberlegungen für die Anwendung begonnen und anschließend auf das Framework eingegangen.

4.1.1 Allgemeine Überlegungen

Als Anhaltspunkte für den Entwurf dienen unter anderem auch die zuvor betrachteten Anwendungen aus Abschnitt 3.1.1. Um eine übersichtliche und einfache Navigation durch die mobile Anwendung zu ermöglichen, bietet eine Tabbar (siehe Abbildung 4.1) eine sehr gute Möglichkeit. Diese Navigationsmethode wird auch in den analysierten Anwendungen verwendet. Auch die Tatsache, dass es nur wenige Einträge gibt, spricht für die Wahl einer solchen Navigation. Des weiteren wurde die grafische Darstellung des Schlafverlaufes aufgegriffen, da so ein visueller Eindruck entsteht, mit dem auf den ersten Blick alles erkennbar ist. Eine Hilfe beziehungsweise Anleitung wurde bereits in den Anforderungen definiert und fand auch in der Sleep Cycle App ihre Anwendung [Nor15].

Abbildung 4.1: Tabbar als Navigationsstruktur

Zunächst wird in der folgenden Abbildung 4.2 ein Überblick über die Struktur der Anwendung gegeben. Dabei stützt sich die komplette Navigation auf die bereits erwähnte Tabbar. Die Pfeile kennzeichnen die jeweiligen Interaktionsmöglichkeiten. Die Ansicht der laufenden Aufnahme sowie die Detailansicht einer Aufnahme sind nicht direkt über das Menü erreichbar sondern nur über die entsprechenden Views. Es ist jedoch möglich aus den eben genannten Ansichten zu allen anderen Menüpunkten zu gelangen, da die Tabbar dauerhaft präsent ist.

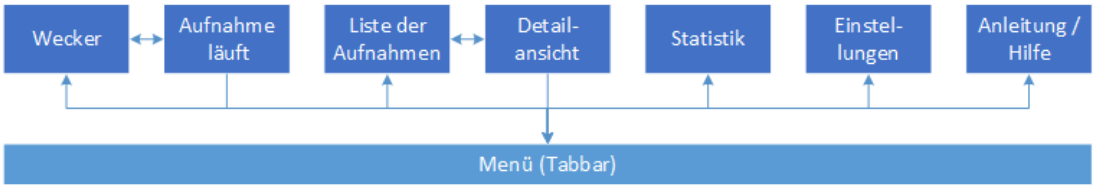

Abbildung 4.2: Struktur der Schlafüberwachungsanwendung

4.1.2 Beispielszenario - Bedienung

In diesem Abschnitt soll näher auf die Bedienung der Anwendung eingegangen werden. Da für diese Anwendung keine komplexen Eingaben erforderlich sind, wird das Bedienkonzept anhand eines Standardszenarios beispielhaft durchgeführt.
Das Szenario umfasst das Stellen des Weckers, sowie das Starten der Aufnahme. Nach Ablauf der Aufnahmezeit wird dann das Ergebnis betrachtet, in dem die entsprechende Aufnahme ausgewählt und die nötigen Details dazu aufgerufen werden. Um dieses Vorgehen zu veranschaulichen, werden die entsprechenden Mockups aus 4.1.3 verwendet.
Der Wecker wird durch drehen an der Stunden- beziehungsweise Minutenauswahl eingestellt. Dabei kann durch stärkere Gesten schneller gedreht werden. Ist die gewünschte Zeit eingestellt, genügt das Drücken des Startbuttons (vgl. Abb. 4.3) um mit der Aufnahme zu beginnen. Nach Beendigung der Aufnahme muss in der Tabbar auf die Statistik gewechselt werden. Dort befindet sich eine Liste aller vorhandener Aufzeichnungen. Durch das Auswählen der obersten Zeile (letzte Aufnahme) gelangt man automatisch zur Detailansicht. (vgl. Abb. 4.4) Diese Ansicht kann über die Tabbar oder durch die Verwendung der Zurück-Schaltfläche oben links wieder verlassen werden.

4.1.3 Detailentwurf

Nachfolgend werden die Entwürfe anhand entsprechender Mockups vorgestellt. Diese wurden mit dem Storyboard-Tool aus Xcode erstellt [App15g].

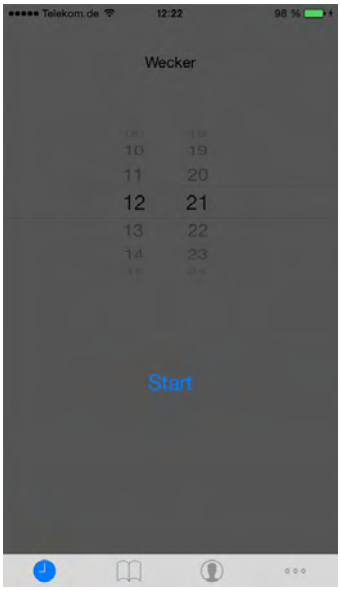

Abbildung 4.3: Detailentwurf - Mockup für die Weckeroberfläche

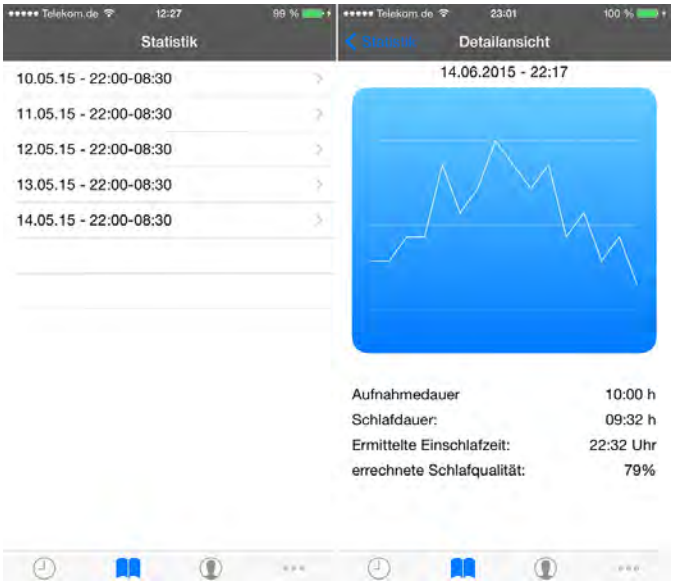

Abbildung 4.4: Detailentwurf - Mockups für Übersicht der Aufnahmen und Detailansicht einer Aufnahme

Die erste Abbildung 4.3 stellt den Entwurf der Weckeroberfläche dar. Diese Oberfläche dient dem Anwender dazu, seine gewünschte Weckzeit (Stunden und Minuten) einzustellen und den Aufnahmevorgang zu starten.
Die obenstehende Abbildung 4.4 zeigt links eine tabellarische Übersicht der bereits vorhandenen Aufnahmen. Dabei wird das Aufnahmedatum und die dazugehörige Uhrzeit des jeweiligen Eintrages angegeben. Auf der rechten Seite ist eine Detailansicht einer solchen Aufnahme aus der zuvor erläuterten Tabelle zu sehen. Hier ist auch die grafische Umsetzung des Schlafverlaufes abgebildet. Darüber hinaus kommen noch weitere Angaben zur gewählten Aufnahme hinzu, die textuell unterhalb der Grafik angefügt sind.
In der letzten Abbildung 4.5 ist auf der linken Seite die Ansicht der Einstellungen dargestellt. Abgebildet sind dort eine Auswahlmöglichkeit für Wecktöne und eine für die Lautstärke des Alarmtons. Nebenstehend ist ein Entwurf für die eingangs erwähnte Hilfe beziehungsweise Anleitung zu finden.

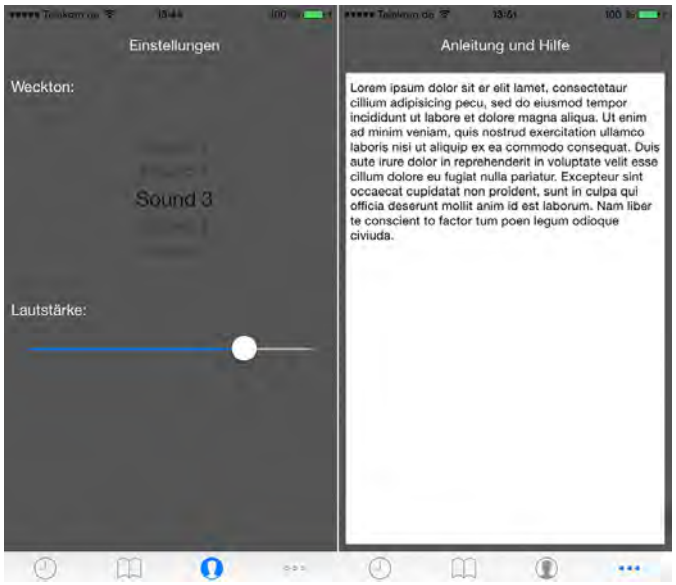

Abbildung 4.5: Detailentwurf - Mockups für Einstellungs- und Hilfeoberflächen

4.2 Framework

Dieser Abschnitt behandelt den Entwurf des Frameworks. Hierbei wird auf die Gestaltung der einzelnen Klassen und deren Verbindungen untereinander eingegangen. Des weiteren muss der Zugriff auf die Sensorik beachtet und in die Entwurfsüberlegungen mit einbezogen werden.

4.2.1 Allgemeine Überlegungen

Um eine Übersicht über die geplante Struktur zu bekommen, dient Abbildung 4.6. Eine zentrale Rolle dabei spielt die Klasse *SleepMonitor*. Diese Klasse soll die aufgenommenen Daten der *AudioRecorder*- und *MotionRecorder*-Klasse empfangen und über den *FileManager* ins Filesystem der Anwendung ablegen.
Für den Zugriff auf die Sensorik sind die beiden bereits erwähnten Recorder-Klassen verantwortlich. Dabei soll die Klasse *AudioRecorder* auf das Mikrofon des Gerätes zugreifen und die Lautstärke während der Aufnahme erfassen. Die *MotionRecorder*-Klasse hingegen empfängt die Daten des Beschleunigungssensors, der im Gerät verbaut ist. Das genaue Format der zu erhebenden Daten wird im nachfolgenden Klassenkonzept näher beschrieben.
Die Klasse *FileManager* beinhaltet alle nötigen Methoden, um Daten in das Filesystem zu schreiben und sie auch wieder auszulesen. Eine vom *SleepMonitor* unabhängige Klasse stellt der *RecordMangager* dar. Diese Klasse beschäftigt sich mit den bereits aufgenommenen Daten zum Schlafverlauf. Zum Beispiel sollen alle Daten zu einer Aufnahme für die weitere Verarbeitung zur Verfügung gestellt werden. Auch diese Klasse greift über den *FileManager* auf das Filesystem zu.
Die Klasse *StatistikManager* ist für die Auswertung aller bestehender Aufnahmen zuständig. Hierfür wird eine separate Statistikdatei im Dateisystem erzeugt. Diese zusätzliche Datei soll unnötige Berechnungen und auch Dateizugriffe einsparen. Der *StatistikManager* übernimmt den Umgang mit den jeweiligen Datensätzen, die Berechnungen sind in der Klasse *SleepEvaluationHelper* ausgelagert um einen besseren Überblick und Struktur zu erhalten.

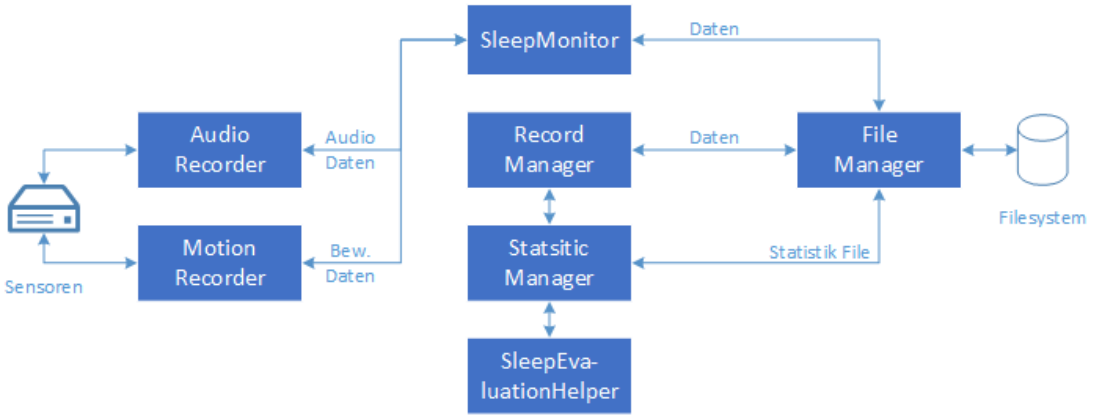

Abbildung 4.6: Strukur des Frameworks zur Schlafüberwachung

4.2.2 Klassenkonzept

Anhand der zuvor erläuterten Struktur des Frameworks, soll nun auf das Klassenkonzept näher eingegangen werden. Dabei wurden die jeweiligen Klassen aus der oben definierten Struktur in ein Klassendiagramm (siehe Abbildung 4.7) überführt und die einzelnen Klassen um Attribute und Methoden erweitert. Aus Gründen der Übersichtlichkeit wurden nur relevante Methoden und Attribute in das Klassendiagramm aufgenommen und zum Beispiel Setter- und Getter-Methoden vernachlässigt.
Im Klassendiagramm wurden aus Gründen der Verständlichkeit, die Zusätze von Schnittstellen zur Anwendung, Hardware (Mikrofon und Beschleunigungssensor) und zum Dateisystem, mit aufgenommen. Im Vergleich zur vorher angegeben Struktur sind noch die Klassen *AudioData* und *MotionData* hinzugekommen. Diese Klassen dienen nur dazu, Datenobjekte für die beiden unterschiedlichen Größen, abzuleiten.
Ein Objekt der Klasse *AudioData* kann somit den ermittelten Dezibel-Wert des Mikrofons entgegen nehmen. Die *MotionData*-Klasse ist für die Daten des Beschleunigungssensors zuständig. Dieser Sensor ermittelt immer drei Werte, je einen für die x-,y-, und z-Richtung [App15d]. Da für die Bewegung nur eine Gesamtbeschleunigung nötig ist, werden die Beträge der Werte zusammengefasst.
Laut Anforderungen soll auch das Prinzip der Datenkapselung umgesetzt werden. Für

den Zugriffsschutz von Klassen, die nur innerhalb des Frameworks verwendet werden, ist dahin gehend gesorgt, dass nur Klassen die explizit als öffentlich gekennzeichnet sind, außerhalb des Frameworks ableitbar sind. Um Methoden, die nur innerhalb einer Klasse aufrufbar sein sollen, zu schützen, erfolgt eine *private*-Kennzeichnung.

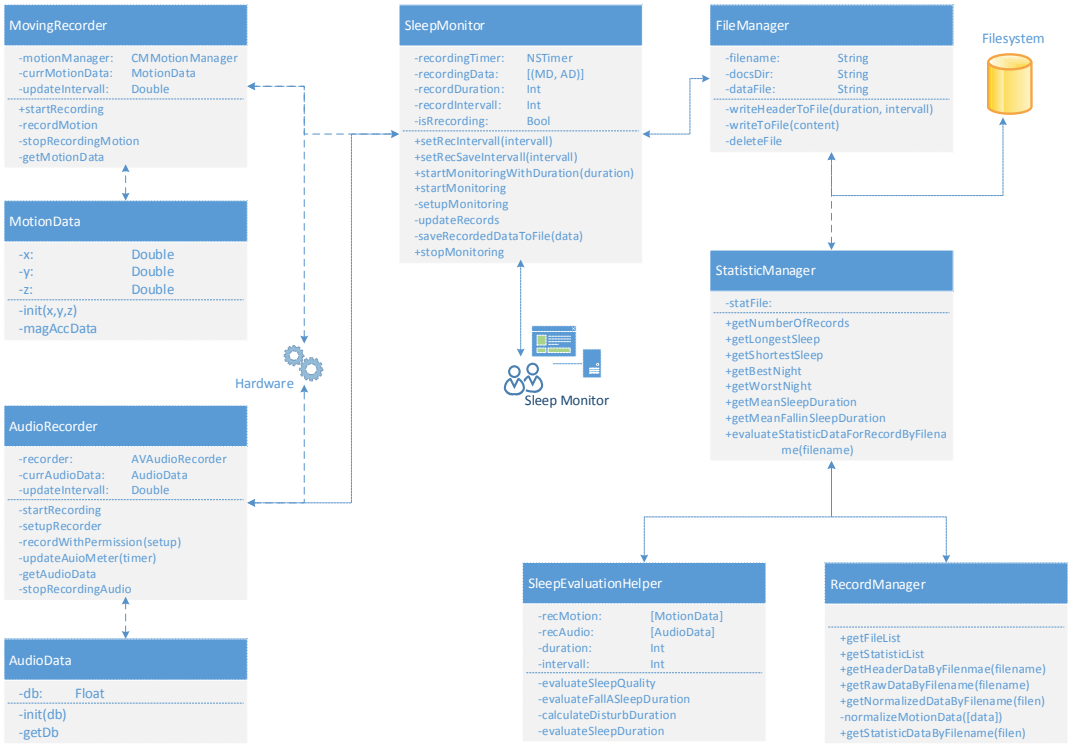

Abbildung 4.7: Klassendiagramm - Framework

4.3 Zusammenhang

Dieser letzte Teilabschnitt des Entwurfskapitels soll nun die beiden zuvor getrennt betrachteten Bestandteile *Anwendung* und *Framework* nochmals aufgreifen und deren Zusammenhang dargestellt werden. Da beide Teile im Rahmen dieses Projektes in Beziehung zu einander stehen, ist es wichtig ihre Interaktionen darzustellen. Nachfolgende Abbildung 4.8 stellt diese Zusammenhänge dar. Hierbei ist anzumerken, dass nur aus

dem Aufnahmedialog heraus aktiv in das Framework eingegriffen wird. Alle anderen Interaktionen stellen nur Datenübermittlungen seitens des Frameworks dar. Somit erfolgt der steuernde Zugriff auf das Framework nur an dieser einen Stelle.

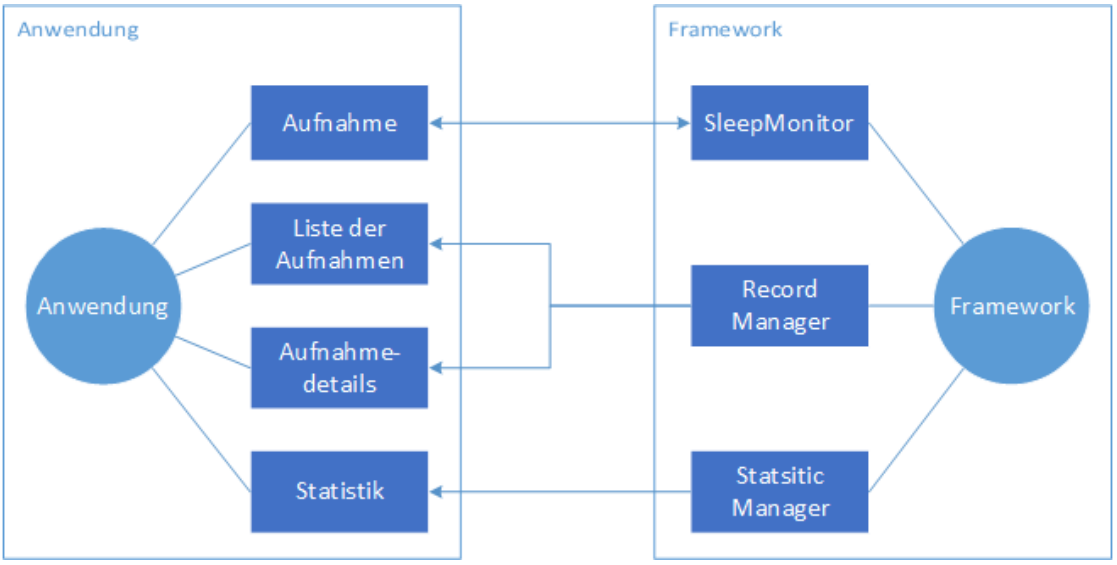

Abbildung 4.8: Beziehung - Anwendung und Framework

5

Implementierung

Nach Abschluss der Entwürfe folgt die eigentliche Realisierung der Anwendung und des Frameworks. Dieses Kapitel befasst sich mit der Implementierung dieser beiden Teile. Sowohl das Framework als auch die Beispielanwendung sind mit der bereits in 2.3 vorgestellten Programmiersprache Swift umgesetzt. Xcode (siehe 2.2) dient als Entwicklungsumgebung für dieses Projekt. Da es sich um eine native Anwendung handelt, kann auf die Standardbibliotheken seitens Apple für den Zugriff auf die Sensorik, zurückgegriffen werden [SSP+13].

Zunächst wird näher auf das Framework eingegangen. Dabei wird zuerst die Umsetzung der Datenerfassung erläutert. Im Anschluss daran folgt die Analyse und Interpretation der zuvor ermittelten Daten. Hier werden die in Kapitel 2 angesprochenen Methoden angewandt. Ergänzend kommt das Dateihandling und die Datenstruktur zu diesem Abschnitt hinzu.

Im zweiten Abschnitt dieses Kapitels soll auf die Beispielanwendung genauer eingegan-

gen werden. Hierfür wird das Design der Benutzeroberflächen, sowie der verwendete Interfacebuilder von Xcode näher beschrieben. Im Anschluss daran folgt eine Beschreibung zur Einbindung eines Frameworks in eine mobile Anwendung auf iOS-Basis. Die Visualisierung der vom Framework zur Verfügung gestellten Daten stellt den abschließenden Part der Anwendungsentwicklung dar. Zusätzlich sind noch Besonderheiten und Probleme angeführt, die diese Entwicklungsphase mit sich bringt.

5.1 Realiserung des Frameworks

Dieser Abschnitt greift unter anderem den Aufbau des Frameworks, die Datenerfassung und die Analyse dieser Daten auf. Zusätzlich wird noch auf das Dateihandling und die Datenstruktur eingegangen, um die erhobenen Werte zu speichern.

5.1.1 Architektur

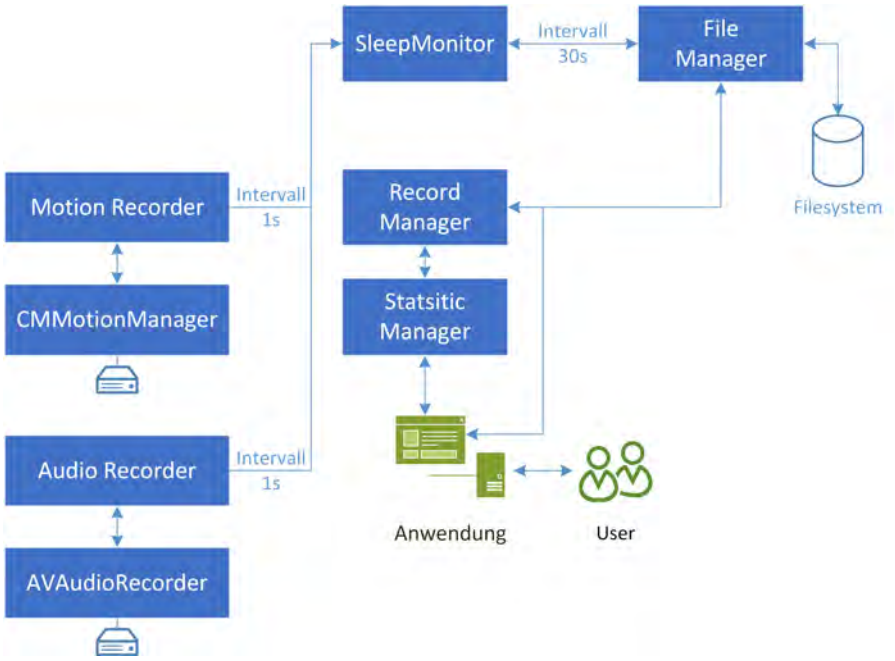

Abbildung 5.1: Übersicht - Architektur des Frameworks

Die Grafik 5.1 stellt den grundlegenden Aufbau und das Konzept des Frameworks in Bezug auf den Datenfluss genauer da. Für die Aufnahme von Bewegung und Geräuschen sind jeweils Zyklen von 1 Sekunde gewählt. Zudem wird nicht jeder Datensatz sofort mittels des *FileMangagers* gespeichert, sondern nur alle 30 Sekunden werden die angesammelten Werte abgelegt. Um Daten an die Anwendung weiter zu geben sind der *StatisticManager* oder auch der *RecordManager* zuständig.
In den beiden nachfolgenden Abschnitten wird die Bewegungserfassung unter der Verwendung des *CMMotionManagers* beziehungsweise die Geräuschaufnahme mit dem *AVAudioRecorder* genauer dargestellt.

5.1.2 Bewegungserfassung

Für die Bewegungserfassung existiert der *CMMotionManager*, auf den zurückgegriffen werden kann. Dieser Manager ist Teil des *CoreMotion-Frameworks* und für die Verwendung des Beschleunigungssensors zuständig [App15c] [SPSR15].

```
1  var motionManager: CMMotionManager
2  if motionManager.accelerometerAvailable{
3      motionManager.accelerometerUpdateInterval = intervall
4      motionManager.startAccelerometerUpdatesToQueue(
5      NSOperationQueue.currentQueue(),
6      withHandler: {(
7          data: CMAccelerometerData!, error: NSError!) in
8              var x = data.acceleration.x
9                  ...})}
10   else {
11   ...
12   motionManager.stopAccelerometerUpdates()
13   ...
```

Listing 5.1: CMMotionManager - Codebeispiel

Der Codeauszug 5.1 soll die Benutzung des Managers und des Beschleunigungssensors verdeutlichen. Zunächst wird geprüft, ob der Sensor vorhanden ist. Anschließend wird festgelegt mit welchem Intervall die Werte abgegriffen werden. Nach dem Start lassen sich die Werte mittels eines Handlers auslesen. Beispielhaft ist das Auslesen des zurückgelieferten x-Wertes im Code angegeben. Um die Aufzeichnung der Werte zu beenden, wird die Funktion *stopAccelerometerUpdates()* aufgerufen.

5.1.3 Audioerfassung

Für die Aufnahme von Ton ist die *AVAudioRecorder*-Klasse geeignet. Diese Klasse ist Bestandteil des *AVFoundation*-Frameworks, welches im Standard von Xcode enthalten ist und lediglich eingebunden werden muss. Das *AVFoundation*-Framework stellt eine Schnittstelle zur Verfügung um audiovisuelle Medien in iOS- und OSX-Anwendungen zu verwalten und abzuspielen [App15a].
Auf Grund der Tatsache, dass nur die Lautstärke der aufgezeichneten Geräusche und nicht das Geräusch an sich benötigt wird, sind die Aufnahmeeinstellungen wie im Codeauszug 5.2 gewählt. Diese Konfiguration verwendet möglichst minimale Einstellungen um die Aufzeichnung durchzuführen. Dabei wird jede Sekunde die maximale Lautstärke mit der *updateMeters*-Funktion aufgezeichnet.

```
1 var recordSettings = [
2    AVFormatIDKey: kAudioFormatMPEG4AAC,
3    AVEncoderAudioQualityKey : AVAudioQuality.Min.rawValue,
4    AVEncoderBitRateKey : 8000,
5    AVNumberOfChannelsKey: 1,
6    AVSampleRateKey : 22050
7 ]
```

Listing 5.2: AVAudioRecorder Einstellungen - Codebeispiel

5.1.4 Analyse und Interpretation der Daten

Um die erhobenen Rohdaten zu verarbeiten erfolgt zunächst eine Normalisierung diese Werte. Die Bewegungsdaten werden anhand der nachfolgenden Formel, und die Audiodaten mittels einer Meter-Tabelle normalisiert [App15e]. Alle weiteren Schritte erfolgen mit den normalisierten Datenwerten [Muh13].

$$\textit{norm[i]} = \frac{a[i] - Min}{Max - Min} \tag{5.1}$$

Die Einteilung in Schlaf- beziehungsweise Wachphasen erfolgt über einen statistischen Ansatz. Dabei wird ein Schwellenwert ermittelt, anhand dessen sich die Bewegungsdaten einteilen lassen. Hierbei werden immer Zeitintervalle von 4 Minuten betrachtet und gezählt, wie oft dieser Schwellenwert pro Zeitraum überschritten wird. Sind mehr als 10% der Bewegungsdaten über diesem Wert, so wird dieser Zeitraum als Wachphase eingestuft. Sind entsprechend weniger verzeichnet, so kann dieses Intervall als Schlafphase eingeteilt werden [Alv14].
Der Schwellenwert wird durch nachfolgende Gleichung berechnet. Diese Gleichung ist eine bekannte statistische Methode um Schwellenwerte zu ermitteln [Alv14][T. 10].

$$\textit{Schwelllenwert} = \frac{Mittelwert + Standardabweichung}{2} \tag{5.2}$$

Dieses Vorgehen liefert ein binäres Array, welches die beiden Zustände schlafend oder wach, abbildet. Diese Phasen stehen für die Anwendung bereit und können für eine visuelle Umsetzung dienen.
Für die Werte Einschlafzeit, Schlafdauer und Schlafeffizienz werden wiederum die normalisierten Bewegungsdaten für die Interpretation herangezogen. Die Analyse der Einschlafzeit lässt sich über die Iteration der Bewegungsdaten bestimmen. Hierbei wird nach dem ersten Zeitraum von 10 Minuten gesucht in dem keine nennenswerten Bewegungen statt finden [Nat12]. Mit der Einschlafzeit lässt sich in Folge auch die Schlafdauer berechnen.

Die Analyse der Schlafeffizienz wird mittels folgender Gleichung durchgeführt [Muh13]:

$$\textit{Schlafeffizienz} = \frac{(Schlafdauergesamt) - (Zeitdauerwach)}{(Schlafdauergesamt)} \tag{5.3}$$

Dieses Vorgehen erlaubt eine einfache Berechnung der Effizienz als Prozentwert und wird wiederum anhand der Bewegungsdaten, die auch in den vorhergehenden Berechnungen zur Anwendung kommen, durchgeführt.
Nach Abschluss der Datenanalyse stehen die Werte für die Anwendung zur Verfügung und können jeder Zeit mit den entsprechenden Methoden dort hin zurückgegeben werden.

5.1.5 Dateihandling und Datenstruktur

Für das Dateihandling ist die Klasse *FileManager* zuständig. Diese Klasse legt die Files im Dokumenten-Verzeichnis der Anwendung auf dem iPhone ab. Dabei wird der Dateiname zur Identifizierung verwendet. Auf Grund der Benutzung eines Zeitstempels ist diese Benennung immer eindeutig. Der *FileManager* kann neue Dateien anlegen, die hinzugekommenen Datenwerte anfügen oder auch die gesamte Datei wieder entfernen. Außerdem existiert eine Methode um die Kopfdaten zu einem File hinzuzufügen.
Als Dateiformat ist aus Gründen der Einfachheit das CSV-Format in Verwendung.

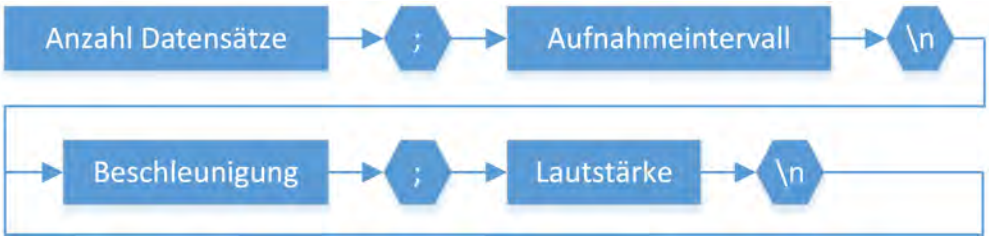

Abbildung 5.2: Schema - Aufbau einer Aufnahmedatei

Die Abbildung 5.2 illustriert den grundlegenden Aufbau der Aufnahmedateien. In der ersten Zeile jeder Datei stehen die Kopfinformationen, die Aufnahmeintervall und die

Anzahl der Datensätze beinhaltet. Im Anschluss folgen n-Datensätze, die jeweils Beschleunigung und Lautstärke durch Semikolon getrennt, beinhalten. Dabei entspricht eine Zeile genau einem Datenpaar und ein Zeilenumbruch symbolisiert den nächsten Datensatz.

5.2 Realiserung der Anwendung

In diesem Abschnitt soll die Umsetzung der Beispielanwendung näher beschrieben werden. Zunächst wird das umgesetzte Oberflächendesign vorgestellt und die Anbindung des Frameworks an die Anwendung beschrieben. Im Anschluss daran folgt die Visualisierung der Schlafdaten, die vom Framework zur Verfügung gestellt werden.

5.2.1 Benutzeroberflächendesign

Da es sich um eine Anwendung zur Schlafüberwachung handelt, wird sie häufig nachts vor dem Schlafen und am Morgen benutzt. Aus diesem Grund ist das Design in einem dunklen Blau gehalten und der Hintergrund stellt einen Nachthimmel dar. Durch den Verzicht auf grelle Farben und ein schlichtes Design soll die Bedienung in dunkler Umgebung leichter fallen.
In Kapitel 4.1 wird die Verwendung einer Tabbar bereits angesprochen. Diese Menüführung hat somit Einfluss auf jede Benutzeroberfläche der Anwendung.
Die Abbildung 5.3 zeigt die umgesetzten Benutzeroberflächen. Die Linke Ansicht stellt die Weckeroberfläche dar, mit der die Weckzeit festgelegt und die Aufnahme gestartet wird. Auf der rechten Seite ist die Liste der vorhandenen Aufzeichnungen abgebildet. Durch die Auswahl eines Eintrages werden Detailinformationen aufgeführt. Um Einträge aus dieser Liste zu entfernen wird durch eine Wischgeste nach links ein Löschbutton eingeblendet mit dem das Entfernen einer Aufzeichnung möglich ist.

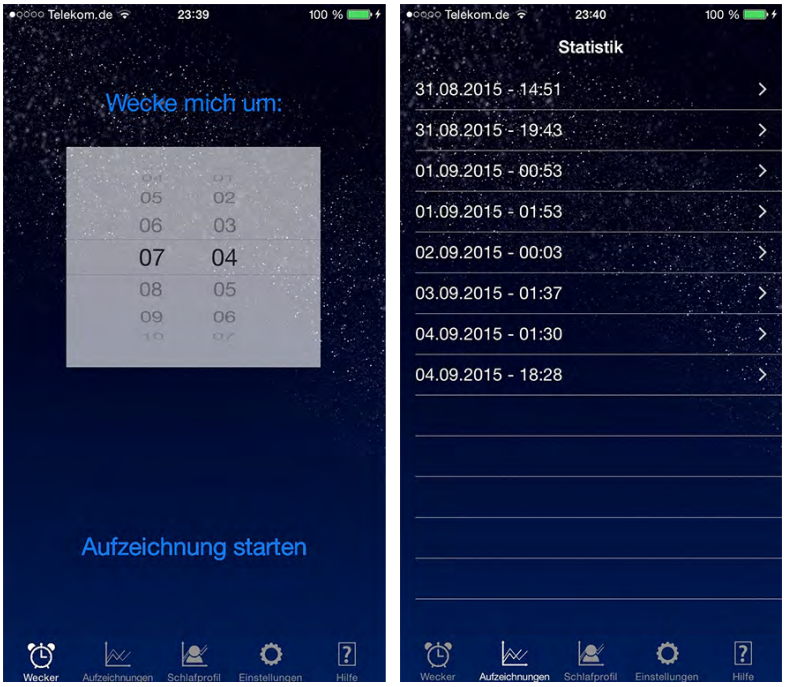

Abbildung 5.3: Benutzeroberfläche - Screenshot von Weckeroberfläche und Aufzeichnungsübersicht

5.2.2 Einbinden eines iOS-basierten Frameworks

Um das Framework in der Beispielanwendung verwenden zu können, muss dieses zuvor in das Projekt für die Anwendung eingebunden werden. Zunächst muss das Framework über die nötigen Klassen verfügen. Dabei ist zu beachten, dass nur öffentliche Klassen außerhalb des Frameworks zugreifbar sind. Anschließend wird das Framework kompiliert und Xcode erstellt ein entsprechendes Framework-Produkt. Dies kann dann über den Finder ausgewählt und per Drag und Drop in die Anwendung gezogen werden. Das Framework erscheint nun in den Projekteinstellungen. Dabei kann es zu doppelten Einträgen kommen, wovon einer wieder herausgelöscht werden muss. Das Framework ist nun sowohl im *Link Binary With Libaries* als auch im *Embed Frameworks* Abschnitt der Projektverwaltung anzutreffen. Nach Abschluss dieses Vorganges ist das Framework an den benötigten Stellen im Programm verwendbar. Dazu ist jedoch eine einmalige

Importanweisung pro Klasse nötig, um Objekte aus den Framework-Klassen ableiten zu können.

5.2.3 Visualisierung der Schlafdaten

Den letzten Abschnitt für die Realisierung der Anwendung stellt die geeignete Darstellung der vom Framework zurückgelieferten Schlafdaten dar. Die Aussagen über den gesamten Schlafverlauf, wie zum Beispiel der Einschlafzeitpunkt können einfach textuell dargestellt werden. Bewegungs- und Audiodaten hingegen lassen sich anhand der Datenmenge nur grafisch abbilden.

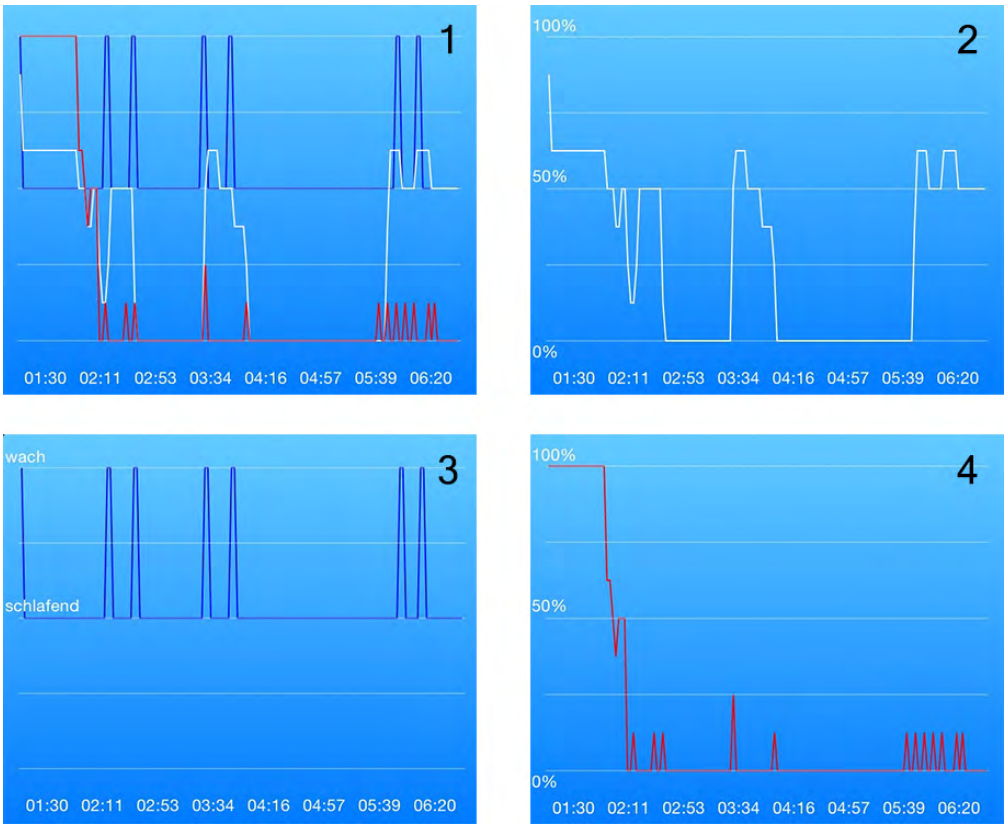

Abbildung 5.4: Grafische Übersicht - Verschiedene Ansichtsmöglichkeiten der Schlafdaten

Hierfür wird eine separate View-Klasse in Xcode erstellt und anschließend in die Detailansicht der Anwendung eingebettet. Auf Grund es beschränkten Platzes ist die maximale Anzahl von Datenpunkte auf dem Grafen mit 150 Punkten beschränkt. Da weit aus mehr Daten pro Nacht existieren, müssen diese zusammengefasst werden. Je nach Anzahl der Datenwerte führt dies zur Bildung von gemittelten Punkten, um die Datenwerte darzustellen.
Die Abbildung 5.4 zeigt die unterschiedlichen Ansichten der jeweiligen Datenarten. Diese können sowohl einzeln als auch zusammengefasst in einer kombinierten Ansicht angezeigt werden. Diese Übersicht stellt die verschiedenen Ansichten des grafischen Verlaufes noch einmal beispielhaft dar. Ansicht 1 zeigt alle Datenwerte. In Bild 2 sind nur die Bewegungen und in 4 nur Geräusche abgebildet. Die Übersicht 3 enthält nur die beiden unterschiedlichen Schlafstadien.

5.3 Besonderheiten und Probleme

In diesem Teilbereich soll auf Besonderheiten beziehungsweise Probleme eingegangen werden, die während der Implementierung aufgetreten sind.
Bei der Entwicklung von Anwendungen für iOS kann man normalerweise auf Simulatoren, die von Xcode bereit gestellt werden, zurückgreifen. Leider musste aus zweierlei Gründen auf diese Möglichkeit verzichtet werden. Zum einen wird die Sensorik durch das Framework verwendet, was dazu führt, dass sich nur die Anwendung simulieren lässt. Den zweiten Punkt stellt das kompilieren dar, denn das Framework benötigt für den Simulator ein separat kompiliertes Projekt. Somit musste die Anwendung jedes mal auf das Gerät installiert werden, um Fortschritte zu testen. Diese Methode ist deutlich zeitintensiver als die Benutzung eines Simulators.
Als ein weiteres Problem kam die Größe und Anzahl der Aufzeichnungsdateien hinzu. Dieses Problem trat erst nach mehreren Aufnahmen auf, die über eine ganze Nacht lang liefen. Mit der Umsetzung eines Schlafprofils, indem der Benutzer Werte über mehrere Nächte zusammengefasst sehen kann, wurde anfangs jede Aufzeichnung einzeln betrachtet und die entsprechenden Werte erhoben. Dieser Sachverhalt führte zu Verzögerungen und somit zu einer schlechten Performanz. Um dieses Problem zu

beheben, wurde eine separate Datei eingeführt, in der nur die ermittelten Werte gehalten werden. In Abbildung 4.6 ist dieser Ansatz bereits eingefügt, um diese Problematik zu umgehen. Es erfolgt eine Aktualisierung dieser Werte, sobald eine Aufzeichnung hinzukommt oder gelöscht wird. Somit kann für das Schlafprofil immer auf diese Statistik-Datei zurückgegriffen werden und eine Vielzahl an Berechnungen lässt sich dadurch einsparen.

6

Tests

In diesem Kapitel soll die implementierte Anwendung und das Framework getestet werden. Zunächst wird die eigentliche Funktionalität untersucht. Dabei geht es darum, dass die Anwendung fehlerfrei arbeitet und die Schlafdaten korrekt aufgezeichnet, gespeichert und ausgewertet werden. Anschließend wird die Anwendung unter realen Bedingungen mehrere Nächte lang getestet und die Ergebnisse dargestellt.

6.1 Funktionstest

Dieser Abschnitt befasst sich mit dem Test der implementierten Funktionalitäten der Anwendung und somit der damit verbundenen Funktionen aus dem Framework. Zunächst wird das Vorgehen näher beschrieben, indem ein Testplan angefertigt wird. Dort sind

alle relevanten Punkte aufgeführt, die zu testen sind. Im Anschluss daran wird das Testergebnis dieses Funktionstests beschrieben.

6.1.1 Testplan

Tabelle 6.1 beschreibt alle Punkte, die getestet wurden. Zudem ist das Resultat des jeweiligen Testvorganges mit angeführt.

Nr.	Funktion	Ergebnis
I	Weckfunktion - normal	Funktion bei allen Testaufzeichnungen gegeben
II	Weckfunktion - Gerät lautlos	Zunächst fehlerhaft, da keine normalen Töne abgespielt werden können. Behoben durch Anpassung der Benachrichtigungseinstellungen in der Anwendung
III	Weckfunktion - Anwendung im Hintergrund	Der Alarmton wird auch dann abgespielt, wenn die Anwendung im Hintergrund liegt. Um den Ton abzuschalten, muss die Anwendung geöffnet werden.
IV	Aufzeichnen - Stromversorgung	Während das Gerät mit dem Stromnetz verbunden ist, wird der Akku nicht belastet und die Aufzeichnung funktioniert. Wird während der Aufzeichnung das Gerät vom Stromnetz getrennt wird die Aufzeichnung abgebrochen.
V	Aufzeichnen - ohne Stromversorgung	Die Problematik des Abschalten des Gerätes während der Aufzeichnung auf Grund einer nicht ausreichenden Stromversorgung ist durchaus gegeben. Eine Einstellung verhindert Aufnahmen ohne Anschluss an das Stromnetz. Wird diese Funktion deaktiviert, muss der Anwender das Risiko in Kauf nehmen.

VI	Aufzeichnen - Anwendung im Hintergrund	Solange die Anwendung nicht komplett geschlossen wird, sondern lediglich durch die Betätigung des Home-Buttons in den Hintergrund gelangt, wird die Aufzeichnung nicht unterbrochen. Es erscheint ein Banner in dem auf die laufende Anwendung hingewiesen wird. Durch Auswahl dieses Hinweises gelangt man wieder zurück zur Anwendung.
VII	Aufnahme - Löschen	Durch das Wischen über einen Tabelleneintrag erscheint ein Lösch-Button. Nach dessen Betätigung wird der Eintrag aus der Tabelle und auch aus dem Dateisystem entfernt. Die Statistikdatei wird beim nächsten Aufruf entsprechend geändert.
VIII	Aufnahme - Details	Wird eine Aufzeichnung aus der Tabelle ausgewählt, so kommt man direkt zur dazugehörigen Detailansicht. Es wurden keine Abweichungen festgestellt.
IX	Schlafprofil	Das Schlafprofil wird jeweils korrekt und aktuell angezeigt, da beim Aufruf die Statistikdatei überprüft wird, ob Änderungen (Löschungen oder neue Aufzeichnungen) vorhanden sind. Ist dies der Fall wird die Datei aktualisiert.
X	Einstellungen	Alle Einstellungen werden beim ersten Start der Anwendung vorbelegt und bei Änderungen werden die neuen Einstellungen abgespeichert.

Tabelle 6.1: Übersicht - Testplan

6.1.2 Ergebnisse

Es wurden alle wichtigen Funktionen der Anwendung und somit auch des Frameworks getestet. Die Tests verliefen durchaus positiv, bis auf die Problematik mit der Weckton-wiedergabe, die aber behoben werden konnte. Das Risiko bei einer Schlafaufzeichnung

ohne Energiezufuhr lässt sich von der Anwendungsseite nur dahin gehend beseitigen, indem man diese Möglichkeit standardmäßig deaktiviert. Von einer generellen Pflicht eines Netzanschlusses wird jedoch abgesehen, da beispielsweise mit voll geladenem Akku und einer kurzen Aufnahmezeit eine Aufzeichnung möglich sein sollte, ohne das Gerät anzuschließen.

6.2 Betriebstest

Im Folgenden wird die Anwendung unter realen Bedingungen eingesetzt und das Verhalten getestet. Hierfür wird vorab das Vorgehen zur Durchführung dieses Tests näher erläutert. Abschließend wird wiederum das Ergebnis des Tests erläutert.

6.2.1 Vorgehen

Für diesen Test wird das Gerät an das Netzteil angeschlossen und wie in der Hilfe der Anwendung dargestellt, im Bett positioniert. Dabei ist sichergestellt, dass das iPhone nicht direkt an der Kante liegt und nicht von einem Kissen oder ähnlichem verdeckt wird. Um Störungen wie beispielsweise ankommende Nachrichten, E-Mails oder sonstige Benachrichtigungen zu vermeiden, wird das Gerät in den Flugzeugmodus versetzt. Diese Benachrichtigungen lösen oftmals Vibrationen am Gerät aus, die zur Verfälschung der Aufzeichnung führen könnten.

6.2.2 Ergebnisse und Fazit

Folgende Übersicht 6.2 stellt die Daten von fünf verschiedenen Aufzeichnungen dar. Es wurden durchschnittlich pro Aufnahmenacht 20899 Messungen angestellt. Zudem sind jeweils die Startzeiten der Aufnahme so wie die ermittelte Einschlafzeit aufgeführt. Anhand dieser Zeiten ergeben sich auch Aufnahme- und Schlafdauer. Die Schlafeffizienz bewegt sich auch jeweils in realistischen Bereichen von 30% bis 71%.

Nr.	Aufnahme-zeit	Mess-punkte	Aufnahme-dauer	Schlaf-dauer	Einschlaf-zeit	Effizienz
1	01:24	24059	06:40	06:03	02:02	71%
2	02:40	15839	04:23	03:56	03:08	52%
3	01:47	24219	06:43	05:11	03:19	51%
4	01:35	19559	05:25	05:15	01:45	30%
5	01:23	20819	05:46	05:32	01:37	42%

Tabelle 6.2: Übersicht - Testergebnisse Betriebstest

Die Effizienz bei Test Nummer 4 ist auffällig niedrig und liegt nur bei 30%. Um zu verdeutlichen, wieso dieser niedrige Wert zu Stande kommt ist die nachfolgende Abbildung 6.1 angeführt. In dieser Abbildung sind die deutlichen Bewegungsphasen (weiße Linie) zu erkennen. Diese deuten auf unruhigen Schlaf hin und mindern somit die Effizienz dessen. Die rote Linie gibt Aufschluss über Geräuschaktivitäten während des Aufnahmezeitraumes, diese werden aber bei der Berechnung der Effizienz nicht beachtet und dienen somit nur der Vollständigkeit.

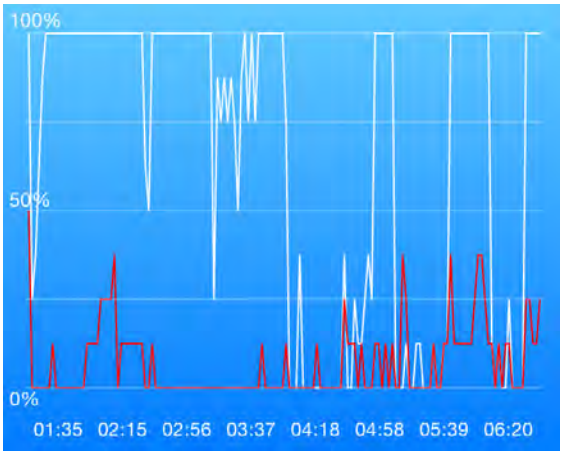

Abbildung 6.1: Analyse - Testergebnis Nummer 4

Die Tests lieferten durchaus interessante Werte. Die subjektiv betrachtet realistisch sind. Vor allem die Einschlafzeit ist sehr unterschiedlich ausgefallen, wobei es Nächte gibt an denen es schwerer fällt einzuschlafen und andere bei denen die Müdigkeit überwiegt. Durch das iPhone im Bett ist es zunächst gewöhnungsbedürftig einzuschlafen, aber nach ein paar Nächten spielt dies keine Rolle mehr.

7

Anforderungsabgleich

In diesem Kapitel werden nach Abschluss der Implementierung und des Testens, die zuvor festgelegten Anforderungen an die Anwendung und an das Framework abgeglichen. Mittels dieses Abgleiches lässt sich feststellen, wie und in welchem Maße die unterschiedlichen Anforderungen umgesetzt sind. Dabei werden wieder beide Teilbereiche getrennt von einander betrachtet. Es erfolgt eine Untersuchung der einzelnen Anforderungen in Bezug auf deren Erfüllung. Dazu sind jeweils Bemerkungen zur Realisierung jeder Teilanforderung mit angegeben. Den Abschluss dieses Kapitels bilden eine Zusammenfassung des Ergebnisses in Verbindung mit einem Fazit.

7.1 Anwendung

Dieser Abschnitt befasst sich mit dem Abgleich der Anforderungen die die Anwendung betreffen. Zuerst werden die funktionalen Anforderungen, gefolgt von den nicht funktionalen Anforderungen betrachtet. Die Betrachtung erfolgt jeweils in tabellarischer Form.

7.1.1 Funktionale Anforderungen

Die Tabelle 7.1 gibt Aufschluss über die funktionalen Anforderungen der Anwendung.

Nr.	Anforderung	Status	Beschreibung
I	Weckerfunktion	erfüllt	Für die Weckfunktion wurde eine eigene Ansicht erstellt und auch durch Tests (siehe Tabelle 6.1) überprüft. Die gewünschte Weckzeit lässt sich mit einem Datepicker, der für Uhrzeiten modifiziert wurde, auswählen.
II	Weckerfunktion - Gerät lautlos oder im Standby-Modus	erfüllt	Auch diese Anforderung an den Wecker ist umgesetzt und mit Tests geprüft, um diese wichtige Funktion sicherzustellen.
II	Starten der Aufnahme	erfüllt	Sobald eine Weckzeit eingestellt ist und das Gerät mit dem Stromnetz verbunden ist (Durch Einstellung auch ohne Stromzufuhr möglich) kann die Aufnahme gestartet werden. Während der Aufzeichnung erscheint eine Ansicht, in der die verbleibende Zeit angezeigt wird.

III	Beenden der Aufnahme	erfüllt	Wurde eine Aufnahme gestartet, kann diese per Betätigung des entsprechenden Buttons manuell beendet werden. Die Aufzeichnung endet automatisch nach Ablauf der eingestellten Zeit.
IV	Übersicht aller Aufnahmen	erfüllt	Eine Liste aller Aufnahmen wird in tabellarischer Form angeboten. Jeder Eintrag ist mit Datum und Startzeitpunkt beschriftet und nach zeitlicher Reihenfolge sortiert.
V	Details einer Aufnahme	erfüllt	Durch Auswahl eines Tabelleneintrages in der Übersicht wird dieser in einer neuen Ansicht im Detail wiedergegeben. Diese Anzeige enthält alle wichtigen Informationen über den Aufnahmezeitraum.
VI	Statistiken	erfüllt	Es wurde eine eigene Ansicht erstellt, in der zusammengefasste Werte der einzelnen Aufnahmen aufgeführt sind. Es werden alle geforderten Werte aufgeführt. Darüber hinaus wird die durchschnittliche Schlafdauer, Schlafeffizienz und Einschlafdauer mit angegeben.

Tabelle 7.1: Abgleich - funktionaler Anforderungen der Anwendung

7.1.2 Nicht funktionale Anforderungen

In diesem Abschnitt werden die nicht funktionalen Anforderungen der Anwendung abgeglichen. Diese Ausführung stellt nachfolgende Tabelle 7.2 dar.

Nr.	Anforderung	Status	Beschreibung
I	Robustheit	erfüllt	Die geforderten Kriterien wurden alle hinreichend getestet (siehe 6.1).

II	Zuverlässigkeit	erfüllt	Die Weckfunktion ist auch anhand von Tests überprüft worden (siehe 6.1).
III	Individualisierbarkeit	erfüllt	Um dieser Anforderung nachzukommen ist einen eigenständiger Menüpunkt mit Einstellungen eingeführt worden. Hier kann der Benutzer unter verschiedenen Wecktönen wählen. Außerdem kann die Lautstärke angepasst werden. Zusätzlich ist es möglich den benötigten Anschluss an eine Stromquelle für Aufzeichnungen zu deaktivieren beziehungsweise erneut einzuschalten.
IV	Hilfestellung	erfüllt	Eine Ansicht in der die wichtigsten Funktionen samt Schaubild für die Positionierung des Gerätes ist vorhanden.

Tabelle 7.2: Abgleich - nicht funktionaler Anforderungen der Anwendung

7.2 Framework

Wie zu Beginn des Kapitels bereits erwähnt, folgen in diesem Abschnitt die Abgleiche der Anforderungen des Frameworks. Auch hier werden wiederum zunächst die funktionalen Anforderungen abgeglichen. Die nicht funktionalen Anforderungen folgen im zweiten Teilabschnitt.

7.2.1 Funktionale Anforderungen

Nachfolgend erfolgt der Abgleich der funktionalen Anforderungen des Frameworks. Dieser Abgleich ist in der nachstehenden Tabelle 7.3 dargestellt.

Nr.	Anforderung	Status	Beschreibung
I	Datenerhebung	erfüllt	Der Zugriff auf die entsprechenden Sensoren wurde umgesetzt und die nötigen Datenwerte können in entsprechend eingestellten Intervallen abgegriffen werden.
II	Aufbereitung der Rohdaten	erfüllt	Die Beträge der Datenwerte des Beschleunigungssensors werden für jede Achse (siehe Formel 2.1 um eine Gesamtbeschleunigung zu erhalten. Diese Rohdaten der gesamten Beschleunigung und der Audiodaten werden anschließend noch normalisiert (siehe 5.1.4).
III	Speicherung der Daten	erfüllt	Die erhobenen Daten werden im Dokumentenverzeichnis der Anwendung abgelegt. Als Datenformat wurde das CSV-Format ausgewählt, da sich die Handhabung einfach gestaltet.
IV	Datenanalyse	erfüllt	Die Funktionen zum Speichern der Daten wurden erweitert um die Daten wieder auszulesen. Dadurch stehen diese für weitere Zwecke zur Verfügung. Es wurde eine zusätzliche Klassen implementiert, die Methoden bietet, um die Daten zu analysieren. Um die Schlafeffizienz ermitteln zu können, werden die Zeiten in denen ein erhöhtes Bewegungsaufkommen stattfand, aufaddiert.

Tabelle 7.3: Abgleich - funktionaler Anforderungen des Framework

7.2.2 Nicht funktionale Anforderungen

Der letzte Abgleich erfolgt an den nicht funktionalen Anforderungen des Frameworks. In Tabelle 7.4 wird dieser Abgleich aufgeführt.

Nr.	Anforderung	Status	Beschreibung
I	Wiederverwendbarkeit	erfüllt	Diese Anforderung wurde erfüllt, indem sich wiederholende Funktionen in Methoden ausgelagert wurden und dementsprechend redundante Abschnitte vermieden werden. Zudem sind alle Klassen und Methoden einheitlich aufgebaut.
II	Datenkapselung	erfüllt	Bereiche von die außerhalb der Frameworkumgebung zugreifbar sein sollen, haben eine dementsprechende öffentliche Deklaration. Es wurde bereits beschrieben, dass nur öffentliche Klassen, Methoden und Attribute von außen erreichbar sind.
III	Aussagekräftige Methodennamen	erfüllt	Alle Methoden folgen einer einheitlichen Namensgebung. Zunächst wird mittels einem Verb die Tätigkeit beschrieben, gefolgt von der eigentlichen Funktion. Ein Beispiel hierfür wäre, „saveRecordedDatatoFile“. Die Namensgebung ist in englisch gehalten.
IV	Kommentare zur Steigerung der Leserlichkeit	erfüllt	Jede Methode einer Klasse wurde mit einem Kommentar über ihre Funktion versehen. Alle Kommentare sind gleichermaßen aufgebaut und wiederum in englisch verfasst. An wichtigen Stellen sind zusätzliche Anmerkungen im Code eingefügt um sich besser zurecht finden zu können.

Tabelle 7.4: Abgleich - nicht funktionaler Anforderungen des Framework

7.3 Ergebnis und Fazit

In diesem abschließenden Abschnitt wird die Umsetzung der geforderten Punkte noch einmal zusammengefasst.
Es konnten soweit alle gestellten Anforderungen sowohl für die Anwendung als auch für das Framework umgesetzt werden. Dabei sind die unterschiedlichen Zustände des iPhones zu beachten, da diese einer genauen Behandlung und Betrachtung unterliegen, um die Schlafaufzeichnung richtig umsetzen zu können. Der Teil der nicht funktionalen Anforderungen scheint zunächst einfach handhabbar, vor allem im Bereich des Frameworks bedeuten diese Punkte einen deutlichen Aufwand. Gerade das Kommentieren der Methoden und wichtigen Stellen im Programmcode bringen eine zeitlich nicht zu unterschätzende Arbeit mit sich.
Zusammenfassend ist das Umsetzen der Anforderungen eine machbare Herausforderung, wobei gerade einfach wirkende Punkte nicht in ihrem zeitlichen Aufwand zu unterschätzen sind. Es wurden zudem keine Studien durchgeführt, um die erhobenen und interpretierten Datenwerte zu evaluieren, da die Schlafüberwachungsanwendung nur umgesetzt wurde, um die Funktionen des Frameworks zu testen und zu benutzen.

8

Zusammenfassung und Ausblick

Nach Abschluss aller Arbeiten kann für den Bereich der Tinnitusforschung ein iOS-basiertes Framework zur Verfügung gestellt werden, das für weitere Forschungsvorhaben in Bezug auf Schlafüberwachung seine Verwendung findet. Auch die Anwendung selbst, könnte als Grundlage für weitere mobile Anwendungen dienen. Auf Grund der Struktur und Lesbarkeit des Programmcodes beider Teile, sollte es gut möglich sein auf diese Arbeit aufzubauen und gegebenenfalls Erweiterungen oder Änderungen durchzuführen.
Zunächst erscheint die Aufzeichnung von Bewegungen und Geräuschen als nicht all zu schwere Herausforderung. Bis jedoch diese Werte in geeigneter Form vorliegen, um sie adäquat verarbeiten, interpretieren und abspeichern zu können, ist ein nicht zu unterschätzender Aufwand in Form von Zeit und Überlegungen nötig. Als weitere Herausforderung ergab sich die Anzeige der vom Framework bereit gestellten Daten in der Anwendung. Da auf Grund der Displaygröße nur ein geringes Platzangebot zur

Verfügung steht, müssen die Daten so aufbereitet und zusammengefasst dargestellt werden, um diesen Platz bestmöglich zu nutzen. Als Problematisch stellten sich Updates seitens Apple im Projektverlauf heraus. Gerade Änderungen an Xcode oder an Swift selbst führten zu Syntaxfehlern, die durch teils aufwändiges Debugging behoben werden mussten.

Die Entwürfe und eine ausführliche Planung der Anwendung und des Frameworks konnten zu einer zielführenden Umsetzung beitragen. Da die Aufnahme auch in den unterschiedlichen Zuständen des Gerätes (zum Beispiel: lautlos oder im Standby-Modus) funktionieren muss, ergaben sich einige Probleme die erst durch weitreichende Recherchen und Tests gelöst werden konnten. Auch die Tatsache, dass Swift eine noch relativ neue Programmiersprache darstellt, die nur im Umfeld von iOS beziehungsweise OSX verwendet werden kann, bedeutet eine geringere Verbreitung als andere Sprachen für die Entwicklung von mobilen Anwendungen.

Um dieses Framework zu erweitern oder die Aufnahmen zu verbessern könnte die Verwendung einer Apple Watch als weitere Aufzeichnungsquelle für zukünftige Arbeiten dienen. Am 09.09.2015 ist das neue Betriebssystem *watchOS 2* für die Uhren bekannt gegeben worden, mit dem es möglich ist native Anwendungen von Drittanbietern auf diesen Geräten zu installieren [Apl15]. Somit könnte die Apple Watch genutzt werden, um entweder eine zweite Aufzeichnungsmöglichkeit zu erhalten, oder diese Uhr anstatt des iPhones als Aufnahmegerät zu verwenden. Zusätzlich könnte auch die Verwendung von mehren Geräten, die während oder nach der Aufnahme ihre Ergebnisse vergleichen und synchronisieren, angedacht werden. Gerade bei zwei Personen in einem Bett könnte diese Erweiterung ihre Anwendung finden. Damit ließe sich auch eine plattformübergreifende Schnittstelle schaffen, damit auch unterschiedliche Geräte für Aufzeichnungszwecke genutzt werden können.

Diese Erweiterungen beziehungsweise Verbesserungen würden zu einem noch robusteren und flexibleren Framework führen. Auch die Ergebnisse der Aufzeichnung könnten dadurch an Genauigkeit und Qualität gewinnen.

Am Ende dieser Bachelorarbeit kann ein gut geeignetes Framework für die Schlafüberwachung zur Verfügung gestellt werden. Die angedachten Erweiterungsmöglichkeiten könnten das Framework zusätzlich bereichern und weiter verbessern.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[Alv14] ALVIKA GAUTAM,VINAYAK S. NAIK, ARCHIE GUPTA, S. K. SHARMA: *An Smartphone-based Algorithm to Measure and Model Quantity of Sleep*. IIITD-TR-2014-005, 2014

[Apl15] APLLE INC.: *Mehr Zifferblätter und Features für ein noch persönlicheres Erlebnis*. http://www.apple.com/de/watchos-2/?afid=p238%7CsjHBO8r0L-dc_mtid_20925oze42631_pcrid_90053731121_%26cid%3Dwwa-de-kwg-watch-slid-, 2015. – Abgerufen am 13.09.2015

[App15a] APPLE INC.: *AVFoundation Framework Reference - AVAudioRecorder Class Reference*. https://developer.apple.com/library/ios/documentation/AVFoundation/Reference/AVAudioRecorder_ClassReference/, 2015. – Abgerufen am 03.09.2015

[App15b] APPLE INC.: *Capturing Device Movement with Core Motion*. https://developer.apple.com/library/prerelease/ios/documentation/EventHandling/Conceptual/EventHandlingiPhoneOS/motion_event_basics/motion_event_basics.html#//apple_ref/doc/uid/TP40009541-CH6-SW14, 2015. – Abgerufen am 03.09.2015

[App15c] APPLE INC.: *CoreMotion Framework Reference - CMMotionManager Class Reference*. https://developer.apple.com/library/ios/documentation/CoreMotion/Reference/CMMotionManager_

Class/#//apple_ref/doc/uid/TP40009670-CH1-SW14, 2015. – Abgerufen am 07.09.2015

[App15d] APPLE INC.: *Handling Accelerometer Events Using Core Motion*. https://developer.apple.com/library/prerelease/ios/documentation/EventHandling/Conceptual/EventHandlingiPhoneOS/motion_event_basics/motion_event_basics.html, 2015. – Abgerufen am 12.08.2015

[App15e] APPLE INC.: *Meter Table*. https://developer.apple.com/library/ios/samplecode/avTouch/Listings/MeterTable_h.html, 2015. – Abgerufen am 19.08.2015

[App15f] APPLE INC.: *Swift. Eine neue Sprache, mit der jeder fantastische Apps entwickeln kann.* https://www.apple.com/de/swift/, 2015. – Abgerufen am 25.07.2015

[App15g] APPLE INC.: *Xcode*. https://developer.apple.com/xcode/, 2015. – Abgerufen am 25.07.2015

[App15h] APPZOO GMBH: *WakeApp*. http://thewakeapp.com, 2015. – Abgerufen am 04.08.2015

[Chr14] CHRISTIAN SCHMITZ: *Macoun 2014: iOS 8, Yosemite und Swift*. http://www.heise.de/mac-and-i/artikel/Macoun-2014-iOS-8-Yosemite-und-Swift-2405151.html?artikelseite=4, 2014. – Abgerufen am 16.08.2015

[Jut10] JUTTA STRAUCH: *Erfassung des Schlaf-Wachverhaltens*. http://www.diss.fu-berlin.de/diss/servlets/MCRFileNodeServlet/FUDISS_derivate_000000006625/Final3.pdf, 2010

[Kre13] KREUZER PM, VIELSMEIER V, LANGGUTH B: Chronic tinnitus: an interdisciplinary challenge. In: *Dtsch Arztebl Int 2013, 100(16): 278-284. DOI: 10.32.38/arztebl.2013.0278*. Deutsches Ärzteblatt International, 2013

[Max15] MAXIMILIAN SCHIRMER, HAGEN HÖPFNER: *Smartphone Hardware Sensors*. https://www.uni-weimar.de/medien/wiki/images/

Zeitmaschinen-smartphonesensors.pdf, 2015. – Abgerufen am 03.09.2015

[Muh13] MUHAMMAD FAHIM, LE BA VUI, IRAM FATIMA, SUNGYOUNG LEE, YONGIK YOON: A Sleep Monitoring Application for u-lifecare Using Accelerometer Sensor of Smartphone. In: *G. Urzaiz et al. (Eds.): UCAmI 2013, LNCS 8276*. Springer International Publishing Switzerland, 2013, S. 151–158

[Nat12] NATALE V, DREJAK M, ERBACCIA A, TONETTI L, FABBRI M, MARTONI M: Monitoring sleep with a smartphone accelerometer. Japanese Society of Sleep Research, 2012

[Nor15] NORTHCUBE AB: *Sleep Cycle*. http://www.sleepcycle.com, 2015. – Abgerufen am 04.08.2015

[SPSR15] SCHICKLER, Marc ; PRYSS, Rüdiger ; SCHOBEL, Johannes ; REICHERT, Manfred: An Engine Enabling Location-based Mobile Augmented Reality Applications. In: *Web Information Systems and Technologies - 10th International Conference, WEBIST 2014, Barcelona, Spain, April 3-5, 2014, Revised Selected Papers*. Springer, 2015 (LNBIP)

[SSP+13] SCHOBEL, Johannes ; SCHICKLER, Marc ; PRYSS, Rüdiger ; NIENHAUS, Hans ; REICHERT, Manfred: Using Vital Sensors in Mobile Healthcare Business Applications: Challenges, Examples, Lessons Learned. In: *9th Int'l Conference on Web Information Systems and Technologies (WEBIST 2013), Special Session on Business Apps*. 2013, S. 509–518

[SSPR15] SCHOBEL, Johannes ; SCHICKLER, Marc ; PRYSS, Rüdiger ; REICHERT, Manfred: Process-Driven Data Collection with Smart Mobile Devices. In: *Web Information Systems and Technologies - 10th International Conference, WEBIST 2014, Barcelona, Spain, Revised Selected Papers*. Springer, 2015 (LNBIP)

[T. 10] T. L. ALVES, C. YPMA, AND J. VISSER: Deriving metric thresholds from benchmark data. In: *26th IEEE International Conference on Software Main-*

tenance in Timisoara, Romania DOI: 10.1109/ICSM.2010.5609747. IEEE Xplore, 2010

Name: Andreas Reiter Matrikelnummer: 690730

Erklärung

Ich erkläre, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den .....

Andreas Reiter