HSNR HRV Quick Check

Projektdokumentation im Rahmen des Moduls:
BWI 50205 Web-Anwendungen
Prof. Dr. rer. nat. Claus Brell

Erstellt von:
Körber, Bastian, 846159
Witzel, André, 854555
Kriesten, Mirco, 835914

Versionsstand: 1.0 final

Mönchengladbach, den 16. Februar 2014

Inhaltsverzeichnis

1 Management Summary

Die vorliegende Projektarbeit entstand im Rahmen der Vorlesung Webanwendungen bei Prof. Dr. rer. nat. Claus Brell im 5. Semester Wirtschaftsinformatik an der Hochschule Niederrhein.

Ziel dieser Arbeit war es, in Form einer technischen Machbarkeitsstudie die Möglichkeit der Messung der Herzratenvariabilität[1] durch den Einsatz eines Polar Bluetooth-Gurtes in Verbindung mit einem mobilen Endgerät, sowie der Auswertung der Messdaten über eine Webseite, zu evaluieren.

Dafür wurde eine mobile Android Applikation zur schnellen Echtzeit-Analyse der Herzratenvariabilität entwickelt. Die grafische Auswertung aller Messergebnisse erfolgt über eine eigens dafür entwickelte Webseite.

Hierfür trägt die Messperson einen Brustgurt, der die Messdaten per Bluetooth an ein Android-Tablet oder Smartphone sendet. Dort werden die Daten in Echtzeit dargestellt und die RR-Intervalle in Form eines Poincaré-Plots[2] grafisch aufbereitet. Die Webseite gibt einen Überblick über alle vergangenen Messungen und bietet die Möglichkeit, sich diese grafisch aufbereitet (Poincaré-Plot, Herzfrequenzverlauf und Histogramm) anzeigen zu lassen.

Durch den Einsatz von weit verbreiteten und preiswerten Endgeräten, einer einfachen Bedienung und der geringen Größe, ist es, neben der privaten Nutzung, besonders für den Einsatz im Gesundheitsbereich, wie Fitnessstudios oder Gesundheitsmessen, geeignet.

Aufgrund des extrem hohen, besonders technischen Aufwandes, wurde das zeitliche Fenster geringfügig überschritten. Dafür ist der Prototyp vollumfänglich lauffähig und kann mit nur geringem Aufwand zu einem verkaufsfähigen Produkt weiterentwickelt werden.

Insgesamt wurde das Projekt äußerst zufriedenstellend und erfolgreich abgeschlossen.

2 Projektbeschreibung

2.1 Ausgangslage

Bereits seit vielen Jahren messen Sportler Ihre Herzfrequenz durch den Einsatz von speziellen Fitnessprodukten. Dies erlaubt den Sportlern nicht nur einen detaillierten Überblick über Ihre Leistungsfähigkeit, sondern gewährt auch eine eingeschränkte Beurteilung der Gesundheit. Eine genauere Analyse, z.B. durch die Analyse der Herzraten-variabilität, war jedoch zumeist nur Medizinern vorbehalten, die dafür spezielle Messinstrumente einsetzten.

Durch die konstante Weiterentwicklung der Technik werden die Messgeräte im Privatsektor immer genauer und sind mittlerweile in der Lage, die benötigen RR-Intervalle aufzuzeichnen.

[1] Die Herzratenvariabilität beschreibt die mehr oder weniger rhythmischen Schwankungen der Herzrate. (Freyer, 2012)

[2] Eine zweidimensionale Punktwolkendarstellungen, in welche jeder RR-Intervall mit dem folgenden RR-Intervall geplottet wird.

Dadurch eröffnet sich dem privaten Fitness- und Gesundheitssektor neue Möglichkeiten der Leistungs- und Gesundheitserfassung.

Trotz der Verfügbarkeit und weiten Verbreitung solcher Messgeräte herrscht aktuell ein Mangel an Systemen, die an die neuen Möglichkeiten anknüpfen und diese sinnvoll nutzen.

Mit diesem Projekt wird versucht diese Lücke zu schließen und ein System anzubieten, das sowohl einfach bedienbar, als auch preiswert ist und somit den Spalt zwischen Medizin- und Fitnesssektor signifikant verringert.

Der Verwendungszweck ist auf den Einsatz im Gesundheits- bzw. Fitnessbereich ausgelegt. So kann bei einer Erstberatung, einem Einführungstraining oder einer Vorführung auf Gesundheitsmessen durch Analyse der Herzratenvariabilität ein genauerer Überblick über den Gesundheitszustand der Person gewonnen werden.

2.2 Zielkatalog

In der folgenden Auflistung befinden sich die Anforderungen an das Projekt. Diese Anforderungen sind aus eigenen Ideen, sowie auch aus Anforderungen des Kunden entstanden.

Primärziele	1. Entwicklung eines funktionsfähigen Prototyps im Rahmen einer technischen Machbarkeitsstudie 2. Der Prototyp soll folgende Fähigkeiten besitzen: a. Herzfrequenz und RR - Intervall Aufzeichnung b. Verbindung mit Polar Wearlink per Bluetooth aufbauen c. Einfach zu bedienen sein 3. Über eine Internetseite zur Auswertung aller Ergebnisse verfügen
Sekundarziele	1. Zephyr Integration 2. Weitere Infos wie Nutzer, Alter etc. einbinden

Tabelle 1 Zielkatalog

Durch den Einsatz von Scrum als Vorgehensmodell, wurden alle Primärziele, sowie auch Punkt 2 der Sekundarziele im Product-Backlog beschrieben. Die Beschreibung im Product-Backlog umfasst eine Schätzung und eine Kurzbeschreibung der User Stories. Im Laufe des Projekts wurde die Schätzung regelmäßig angepasst. Sämtliche Anforderungen an das Produkt werden durch den Product-Backlog abgebildet.

2.3 Geplante Systemarchitektur

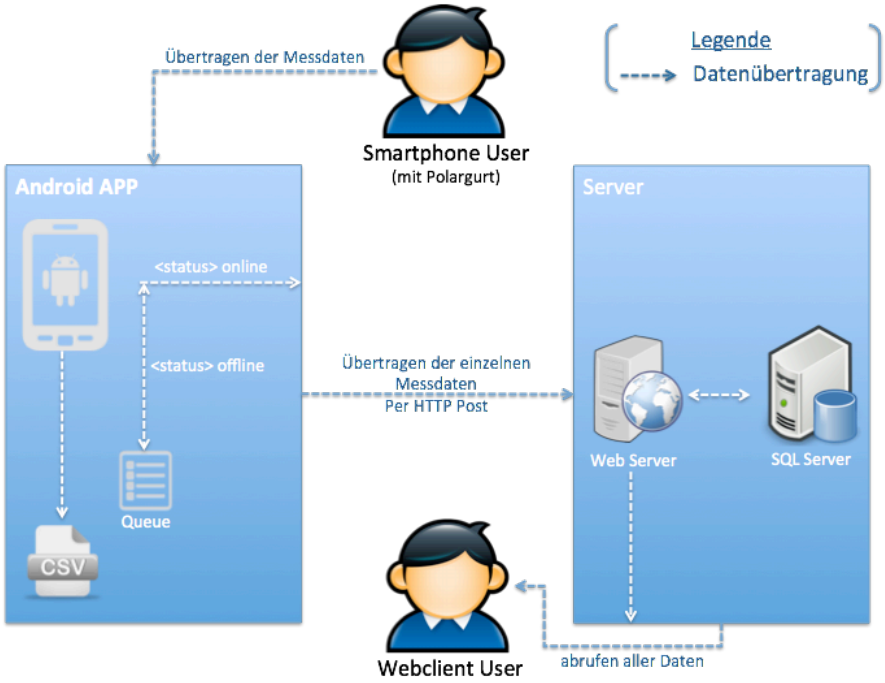


Abbildung 1 Systemarchitektur als Netzwerkdiagramm

Der Benutzer ist mit einem Polar Gurt und einem Android-Gerät ausgestattet, wobei die Kommunikation zwischen beiden Geräten über Bluetooth erfolgt. Die gesendeten Daten des Gurtes werden von dem Android-Gerät empfangen, verarbeitet und in eine lokale CSV Datei abgespeichert. Sofern eine Internetverbindung besteht, überträgt das Endgerät per HTTP-POST die Daten an den Webserver. Dieser schreibt die empfangenen Daten in eine vorgesehene SQL-Datenbank, welche zur späteren Detailanalyse als Grundlage dient. Zusätzlich können die Messdaten mit Benutzername, Alter und Geschlecht versehen werden.

Die Detailanalyse erfolgt über ein Webinterface. Hier werden detaillierte Informationen über die bisherigen Messeinheiten angezeigt. Folgende Diagrammtypen werden wiedergegeben:

- Poincaré-Plot
- Herzfrequenz – Histogramm[3]
- Herzfrequenz – Verlauf

[3] HF-Histogramm nach: http://cdn.imd-hrv.de/downloads/Fachseminar_IMD_HRV.pdf

2.4 Ressourcen

2.4.1 Personen

Name	Kompetenzen	
Bastian Körber	Fachlich	Fundierte Kenntniss in Java, Datenbanken (SQL), PHP, HTML, Objective C, Schnittstellen erfahren
	Methodisch	Projekterfahrung durch studentische Tätigkeit in einer Consulting Firma, Leitung verschiedener Projekte, Scrum erfahren
	Sozial	Teamfähigkeit, Rhetorikkurs, Fähigkeit auf Personen eingehen zu können
Mirco Kriesten	Fachlich	Fundierte Kenntniss in Java, Datenbanken (SQL), PHP, HTML, sehr Programmiererfahren
	Methodisch	Projekterfahrungen durch Ausbildung, Tätigkeit und studentische Tätigkeit in verschiedenen Software Häusern und in einer Consulting Firma. Leitung Repititutorium.
	Sozial	Teamfähigkeit, Rhetorikkurs, Fähigkeit auf Personen eingehen zu können, kreativ
André Witzel	Fachlich	Fundierte Kenntniss in Java, Datenbanken (SQL), PHP, HTML, sehr Programmiererfahren
	Methodisch	Projekterfahrung durch studentische Tätigkeit in einem Software Haus, Mitarbeit an verschiedenen Projekten, Leitung Tutorium.
	Sozial	Teamfähigkeit, Rhetorikkurs, Fähigkeit auf Personen eingehen zu können, kreativ

Tabelle 2 Personen und Kompetenzen

Rolle	Bastian Körber	Mirco Kriesten	André Witzel
Android		X	
Schnittstellenentwicklung		X	X
Webanwendung			X
Datenbanken			X
Scrum Master	X		
Product Owner	X		
Developer	X	X	X

Tabelle 3 Personen und Rollen

Weitere Rollen:
Kunde: Prof. Dr. rer. nat. Claus Brell

2.4.2 Material

Objekt	Anzahl	Bemerkung
Notebook	3	Windows, Mac
Mobile Geräte	4	Samsung Galaxy S3, Samsung Galaxy Tab 2, iPhone
Bluetooth-Gurt	1	Polar Wearlink Bluetooth
Webserver	1	Bereitgestellt durch André Witzel: • Debian • Kunagi • Apache (PHP, PHPmyadmin) • MySQL • FTP-Server
Verwendete Tools		• Kunagi (Scrum Management Tool) • Google Drive & Docs • Google Hangout • Facebook • Bubbl • Draw.io • Visio • GanttProject

Tabelle 4 Materialliste

3 Projektorganisation und –planung

3.1 Projektstrukturplan

Abbildung 2 Projektstrukturplan

3.2 Projektplan

3.2.1 Tabellarischer Projektplan

Tasks

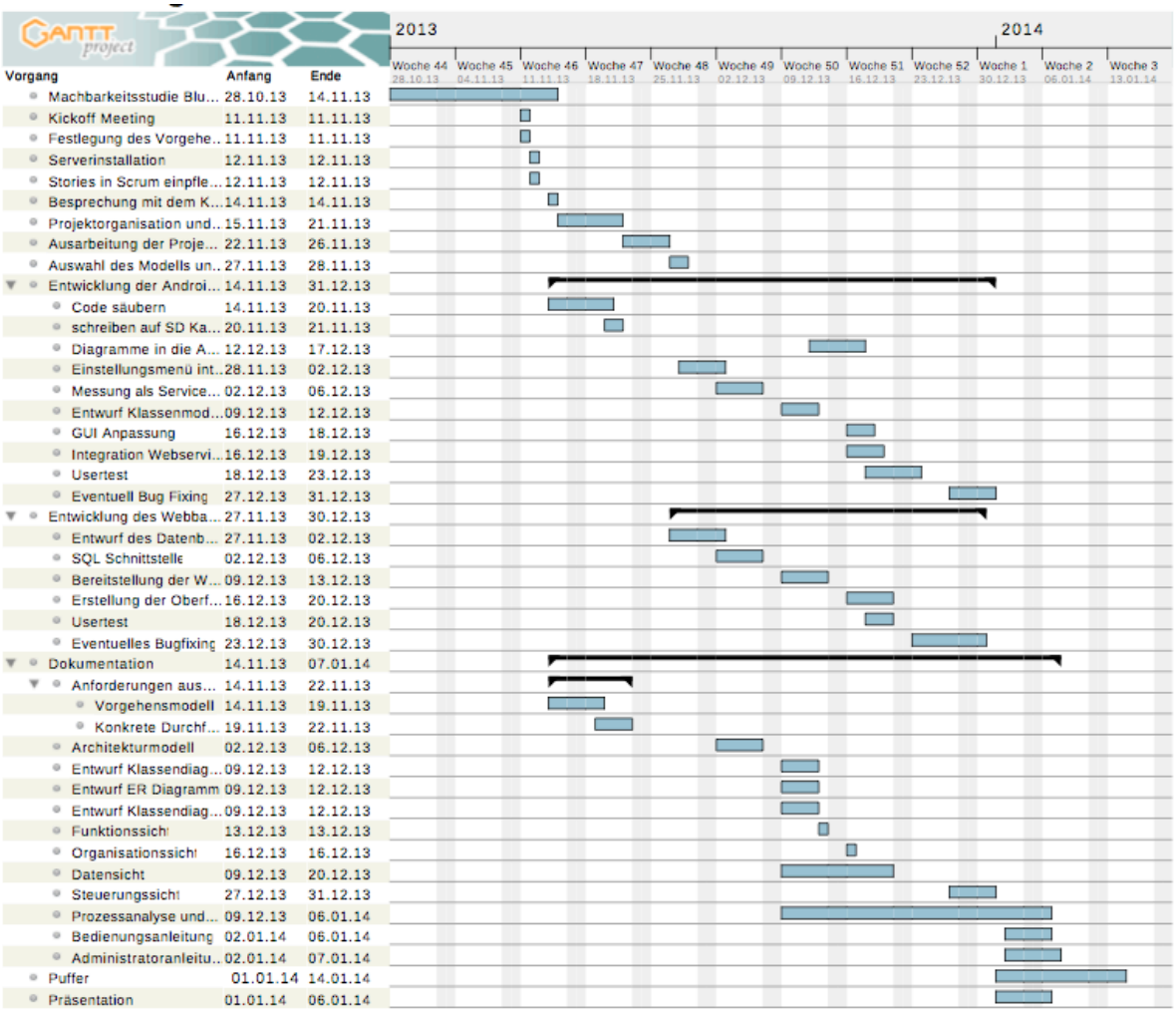
Vorgang	Anfang	Ende
Machbarkeitsstudie Bluetooth Integration	28.10.13	14.11.13
Kickoff Meeting	11.11.13	11.11.13
Festlegung des Vorgehensmodel	11.11.13	11.11.13
Serverinstallation	12.11.13	12.11.13
Stories in Scrum einpflegen	12.11.13	12.11.13
Besprechung mit dem Kunde (Festlegung der Ziele)	14.11.13	14.11.13
Projektorganisation und Planung	15.11.13	21.11.13
Ausarbeitung der Projektbeschreibung	22.11.13	26.11.13
Auswahl des Modells und der Notation	27.11.13	28.11.13
Entwicklung der Android APp	14.11.13	31.12.13
Code säubern	14.11.13	20.11.13
schreiben auf SD Karte	20.11.13	21.11.13
Diagramme in die Applikation einbetten	12.12.13	17.12.13
Einstellungsmenü integrieren	28.11.13	02.12.13
Messung als Service auslagen	02.12.13	06.12.13
Entwurf Klassenmodell	09.12.13	12.12.13
GUI Anpassung	16.12.13	18.12.13
Integration Webservices	16.12.13	19.12.13
Usertest	18.12.13	23.12.13
Eventuell Bug Fixing	27.12.13	31.12.13
Entwicklung des Webbackends	27.11.13	30.12.13
Entwurf des Datenbankkonzepts	27.11.13	02.12.13
SQL Schnittstelle	02.12.13	06.12.13
Bereitstellung der Webservices	09.12.13	13.12.13
Erstellung der Oberfläche	16.12.13	20.12.13
Usertest	18.12.13	20.12.13
Eventuelles Bugfixing	23.12.13	30.12.13

Abbildung 3 tabellarischer Projektplan

Vorgang	Anfang	Ende
Dokumentation	14.11.13	07.01.14
Anforderungen aus Kundensicht	14.11.13	22.11.13
Vorgehensmodell	14.11.13	19.11.13
Konkrete Durchführung	19.11.13	22.11.13
Architekturmodell	02.12.13	06.12.13
Entwurf Klassendiagramm App	09.12.13	12.12.13
Entwurf ER Diagramm	09.12.13	12.12.13
Entwurf Klassendiagramm WEb	09.12.13	12.12.13
Funktionssicht	13.12.13	13.12.13
Organisationssicht	16.12.13	16.12.13
Datensicht	09.12.13	20.12.13
Steuerungssicht	27.12.13	31.12.13
Prozessanalyse und Modellierung	09.12.13	06.01.14
Bedienungsanleitung	02.01.14	06.01.14
Administratoranleitung	02.01.14	07.01.14
Puffer	01.01.14	14.01.14
Präsentation	01.01.14	06.01.14

Abbildung 4 tabellarischer Projektplan

3.2.2 Gantt-Diagramm



Vorgang	Anfang	Ende
Machbarkeitsstudie Blu...	28.10.13	14.11.13
Kickoff Meeting	11.11.13	11.11.13
Festlegung des Vorgehe..	11.11.13	11.11.13
Serverinstallation	12.11.13	12.11.13
Stories in Scrum einpfle...	12.11.13	12.11.13
Besprechung mit dem K...	14.11.13	14.11.13
Projektorganisation und..	15.11.13	21.11.13
Ausarbeitung der Proje...	22.11.13	26.11.13
Auswahl des Modells un..	27.11.13	28.11.13
Entwicklung der Androi...	14.11.13	31.12.13
Code säubern	14.11.13	20.11.13
schreiben auf SD Ka...	20.11.13	21.11.13
Diagramme in die A...	12.12.13	17.12.13
Einstellungsmenü int..	28.11.13	02.12.13
Messung als Service...	02.12.13	06.12.13
Entwurf Klassenmod...	09.12.13	12.12.13
GUI Anpassung	16.12.13	18.12.13
Integration Webservi...	16.12.13	19.12.13
Usertest	18.12.13	23.12.13
Eventuell Bug Fixing	27.12.13	31.12.13
Entwicklung des Webba...	27.11.13	30.12.13
Entwurf des Datenb...	27.11.13	02.12.13
SQL Schnittstelle	02.12.13	06.12.13
Bereitstellung der W...	09.12.13	13.12.13
Erstellung der Oberf...	16.12.13	20.12.13
Usertest	18.12.13	20.12.13
Eventuelles Bugfixing	23.12.13	30.12.13
Dokumentation	14.11.13	07.01.14
Anforderungen aus...	14.11.13	22.11.13
Vorgehensmodell	14.11.13	19.11.13
Konkrete Durchf...	19.11.13	22.11.13
Architekturmodell	02.12.13	06.12.13
Entwurf Klassendiag...	09.12.13	12.12.13
Entwurf ER Diagramm	09.12.13	12.12.13
Entwurf Klassendiag...	09.12.13	12.12.13
Funktionssicht	13.12.13	13.12.13
Organisationssicht	16.12.13	16.12.13
Datensicht	09.12.13	20.12.13
Steuerungssicht	27.12.13	31.12.13
Prozessanalyse und...	09.12.13	06.01.14
Bedienungsanleitung	02.01.14	06.01.14
Administratoranleitu..	02.01.14	07.01.14
Puffer	01.01.14	14.01.14
Präsentation	01.01.14	06.01.14

Abbildung 5 Gantt-Diagramm

3.3 Projektrisiken

Ein großes Risiko könnte die fehlende API von Polar sein. Falls die eingesetzten Entwickler keine Möglichkeit finden den Polar Bluetooth-Gurt mit dem Android-Gerät zu koppeln, ist es unmöglich das Projekt funktionsfähig abzugeben. Zuzüglich besteht die Gefahr, dass durch gesetzliche Änderungen oder Herausgaben des Herstellers, der Bluetooth-Gurt nicht mehr in dieser Form verwendet werden darf.

Des Weiteren kann es vorkommen, dass ein Entwickler eine Anforderung in einem falschen Kontext versteht. Um diesem Problem zu entgehen, ist es wichtig eine für alle Verantwortlichen verständnisvolle Anforderungsanalyse zu erstellen.

Ein häufiges Problem ist die zu geringe Aufwandsschätzung. Oft schätzen Entwickler den Aufwand zu gering, weil die Lösungsfindungsphase und ebenso die Testphase komplett vernachlässigt werden. Hier sollten auf alle Schätzungen ein drittel Mehraufwand berechnet werden.

Meist ändern sich die Anforderungen des Kunden im Laufe des Projektes, darauf sollten die Projektverantwortlichen eingerichtet sein und nötigen Puffer einberechnen.

Eine besondere Ausgangslage dieses Projektes ist, dass alle drei Projektbeteiligten nicht nur studieren und den üblichen Studienaufwand bewerkstelligen müssen, sondern nebenbei eine Tätigkeit in anderen Unternehmen ausüben. Dies sollte ebenfalls berücksichtigt werden, da es vorkommen kann, dass ein Projektmitglied plötzlich aufgrund von Mehraufwand bei der Arbeit nicht kontinuierlich weiterentwickeln kann. Das muss genauso berücksichtigt werden, wie krankheitsbedingte Ausfälle.

4 Vorgehensmodell

4.1.1 Auswahl des Vorgehensmodells

Dieses Projekt wird mit dem Vorgehensmodell Scrum gestaltet. Scrum bietet sich in diesem Projekt an, weil das Ziel und die Möglichkeit des Umsetzen eines fertigen Produktes noch sehr wage ist. Da die Möglichkeit der Umsetzung evaluiert werden muss und später erst die genauen Anforderungen bestimmt werden können, ist es sinnvoll eine agile Managementmethode zu benutzen. Ein weiterer Nutzen ist, dass Scrum an die besondere Position der drei Projektbeteiligten angepasst werden kann. Scrum ist ein selbstlernendes Framework und übernimmt eine mit der Laufzeit genauer werdende Einschätzung der Stories, welche in einem Sprint erledigt werden können. Somit wird die Planung für den Product-Owner um einiges erleichtert.

Die komplette Umsetzung des Projektes fand im Scrum-Tool Kunagi statt. Dort wurde der Sprint-Backlog / Product-Backlog, die Schätzung der Stories, die Impediments und die Verteilung der Tasks erledigt. Lediglich die Daily Scrums und Sprint Reviews wurden über Google Hangout abgehalten, welches jedoch in Kombination mit Kunagi agierte.

Ein weiterer Nutzen der Methodik ist die effektive Rollenverteilung. Jeder Beteiligte kann sich ein Projektgebiet aussuchen, in dem er besonders stark ist und muss nicht Aufgaben erledigen, die ihm nicht liegen. Der Auftraggeber wird durch den Einsatz des Backlogs in den Vordergrund gestellt und kann so jede Änderungsanforderung zeitnah an das Team herantragen. Kunagi wird im unteren Abschnitt noch genauer erläutert.

4.1.2 Konkrete Durchführung

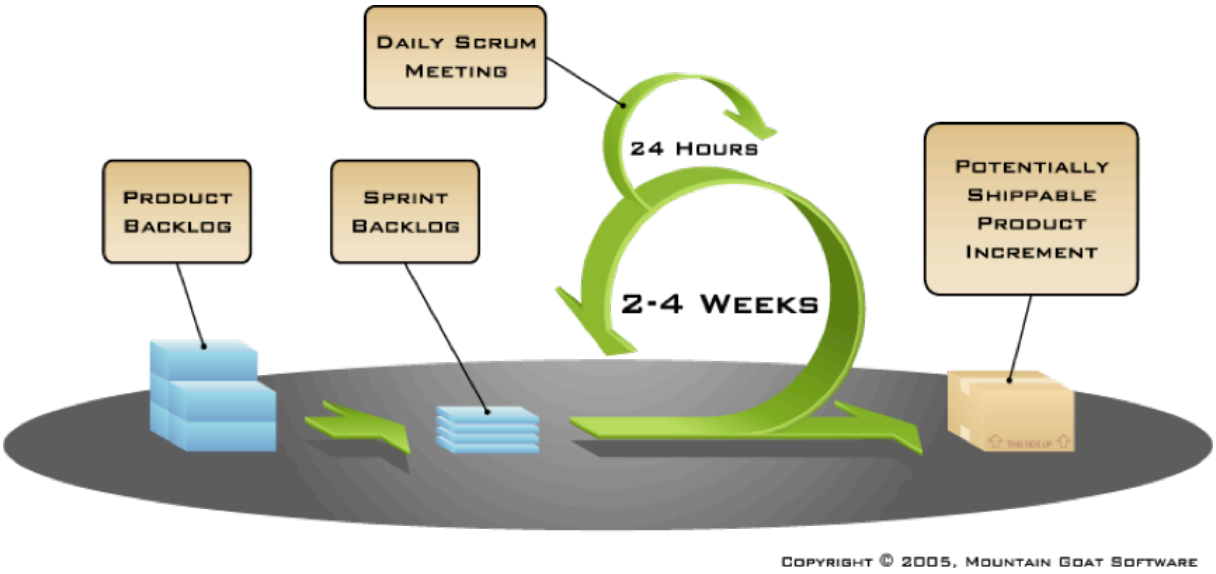


Abbildung 6 Scrum

Aufgrund der verhältnismäßig geringen Projektzeit, wurde die Sprintdauer auf eine Zeit von einer Woche festgelegt. Dies ermöglichte ein flexibles Vorgehen und eine schnelle Reaktion auf Veränderungen im Projekt. Zusätzlich konnten die Projektmitglieder agil auf ihr Studium reagieren.

Ansonsten wurde der Rahmen des Scrum Frameworks mit Hilfe von Kunagi verwirklicht.

4.1.3 Scrum-Tool Kungai

Kunagi ist ein webbasiertes Open-Source Produkt. Es bietet die komplette Planung eines Projekts nach dem Scrum-Frameworks. Das Tool ist intuitiv zu bedienen und für Leute, welche schon Erfahrung mit Scrum haben einfach zu benutzen. Alle Schritte des Scrum Frameworks werden abgedeckt und kollaboratives Arbeiten unterstützt.

Anbei werden einige Ausschnitte des Tools anhand von Screenshots gezeigt und kurz erklärt:

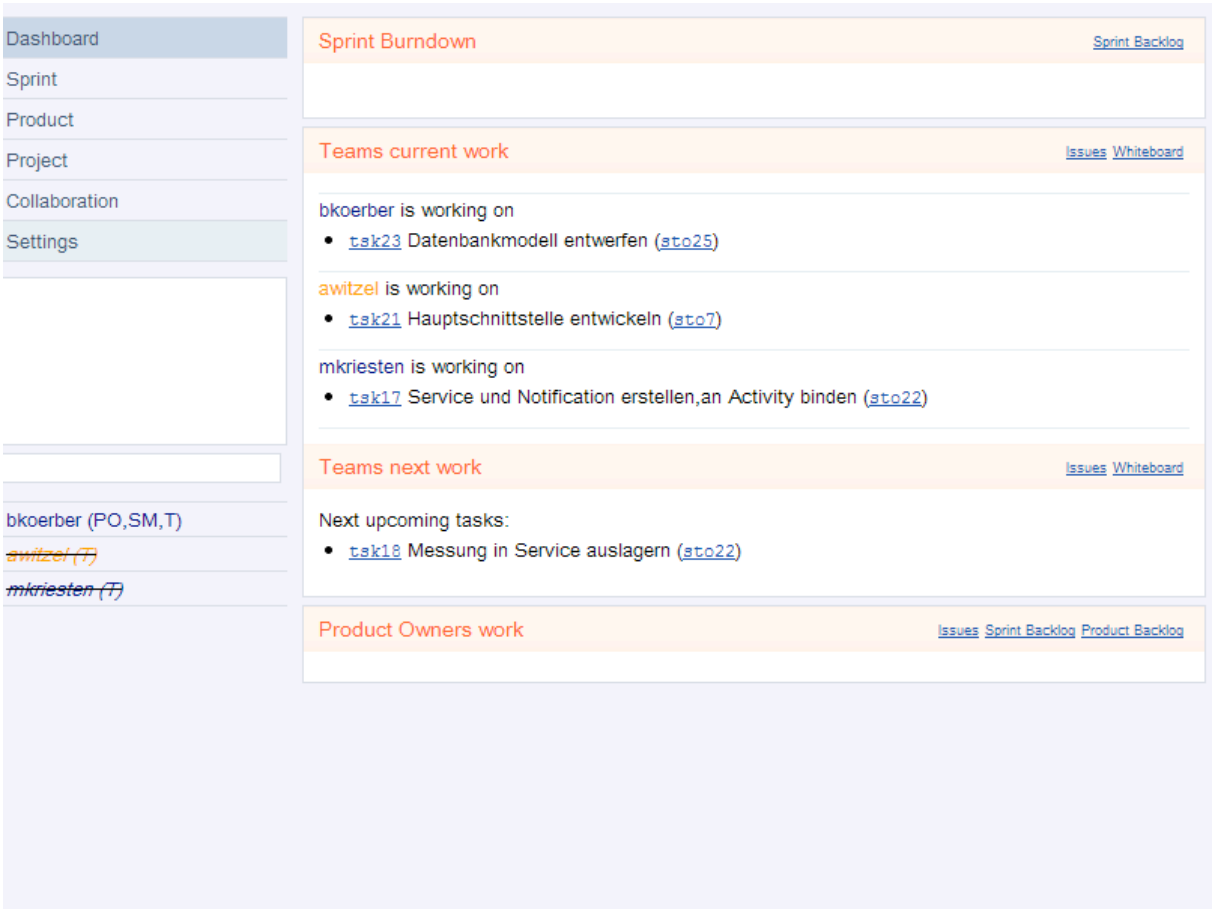


Abbildung 7 Übersicht

Dies ist die Hauptansicht bei Kunagi. Auf den ersten Blick ist zu erkennen, welches Projektmitglied, welche Aufgabe hat. Des Weiteren kann durch das Burndown Diagramm ein direkter Eindruck verschafft werden, wie der aktuelle Sprint gerade verläuft.

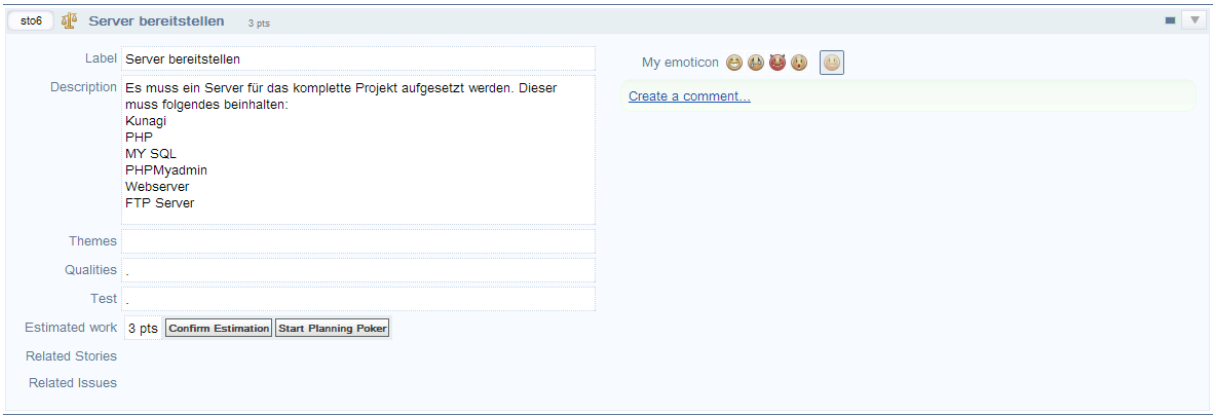


Abbildung 8 Erstellung einer Story

Um eine Story zu erstellen, müssen folgende Angaben gemacht werden:

- Ein Titel
- Eine Beschreibung
- Geschätzte Zeit

Die geschätzte Zeit kann ebenfalls durch ein Scrum Planning Poker geschätzt werden. Jeder Teilnehmer loggt sich bei Kunagi ein und geht auf die zu schätzende Story, nun bedient er das Feld „Start Planning Poker“ und gibt die geschätzte Storypunktzahl als Karte ab. Nachdem jeder User dies getan hat, wird verglichen, ob alle dieselbe Zahl haben. Falls nicht müssen alle nochmal abstimmen. Dieser Zyklus wird solange wiederholt, bis alle dieselbe Storypunktzahl geschätzt haben.

Abbildung 9 Ausschnitt aus dem Product-Backlog

Wie in der Abbildung 8 zu sehen ist, zeigt Kunagi nicht nur den gesamten Product-Backlog an, sondern gibt eine ungefähre Schätzung ab, wann welche Stories erledigt werden können. Diese Stories wurden vorher vom Product-Owner priorisiert. Die Schätzung durch Kunagi erfolgt durch ein Mittelwert der vorher erreichten Storypunkte in einem Sprint. Erfahrungen haben gezeigt, dass umso länger ein Projekt läuft, desto genauer werden die Vorhersagen und Schätzungen durch Kunagi.

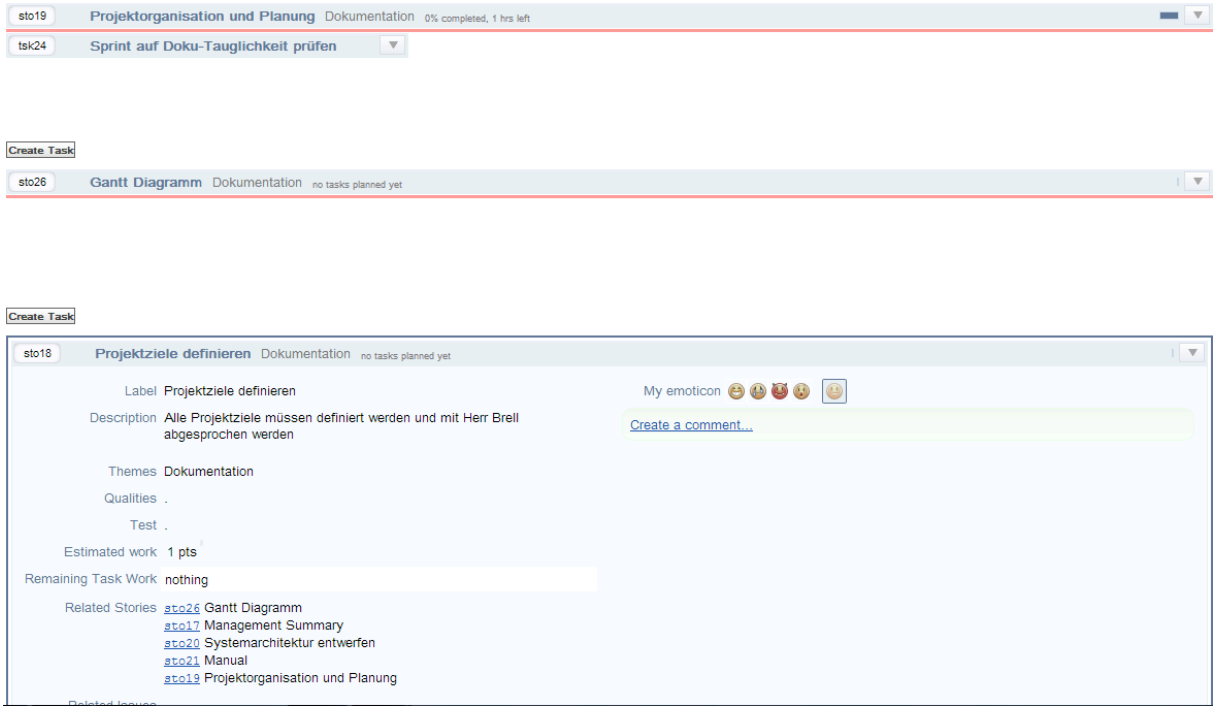


Abbildung 10 Sprint-Backlog

Jedem Sprint werden einzelne Stories zugeordnet. Da eine Story nur ein Überbegriff für verschiedene Tasks ist, werden im Sprint-Planning die einzelnen Stories in Tasks untergliedert. Diese Tasks können sich die Entwickler im Laufe des Sprints selbstständig zuteilen. Somit werden keinem Entwickler Aufgaben aufgezwungen, sondern jeder Entwickler wählt seine Aufgaben aus dem Sprint-Backlog aus.

Sobald ein Entwickler eine Task ausgewählt hat, erhält sie den Status „Claimed Task“. Sobald die Aufgabe abgeschlossen ist, erhält sie den Status „Completed Task“. Wenn alle Tasks einer Story erledigt sind, wird die Story geschlossen.

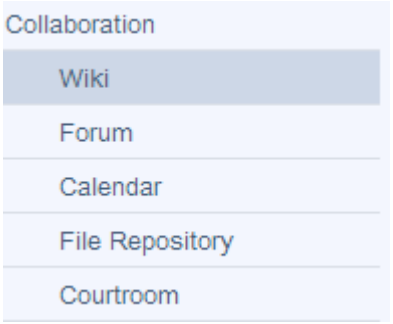


Abbildung 11 Zusammenarbeit unter Kunagi

Kunagi bietet zusätzlich zu den Scrum-Methoden weitere sinnvolle Features, um die Zusammenarbeit in einem Projekt zu unterstützen:

So verfügt Kunagi zum Beispiel über ein Wiki. In ihm kann das erlangte Projektwissen eingepflegt werden. Dadurch wird eine Projektwissensdatenbank erstellt, welche für spätere Recherchearbeiten hilfreich sein kann. Für Diskussionen bietet Kunagi ein Forum, somit bleibt alles, was in diesem Projekt besprochen wurde in einem System bzw. an einem Ort. Um diesen Effekt vollkommen auszunutzen, können alle projektrelevanten Dokumente in einem File Repository gespeichert werden.

Ebenfalls bietet Kunagi einen internen Kalender, dieser ist für komplexere Sachverhalte nicht empfehlenswert. Hier sollte weiterhin auf andere Produkte, wie beispielsweise Outlook, zurückgegriffen werden.

Um die Projektdisziplin auf eine lustige Weise zu verbessern, kann bei jedem Verstoß, z.B. gegen die Scrum-Methodik oder einer Verspätung bei den Daily Scrums, ein Strafbetrag eingeführt werden. Diese Strafen werden im „Courtroom“ festgehalten, sobald genügend Geld in der Kasse ist, kann z. B. eine Projektfeier veranstaltet werden.

5 Systemarchitektur

5.1 Architekturmodell

Für die Fertigstellungen dieses Projektes wurde das ARIS Framework benutzt. Dieses Framework wird meist für kleinere Projekte benutzt, da die Anwendung und Betrachtung der Geschäftsprozesse komplett und dennoch sehr einfach gehalten sind.

Auswahl des Modells und der Notationen

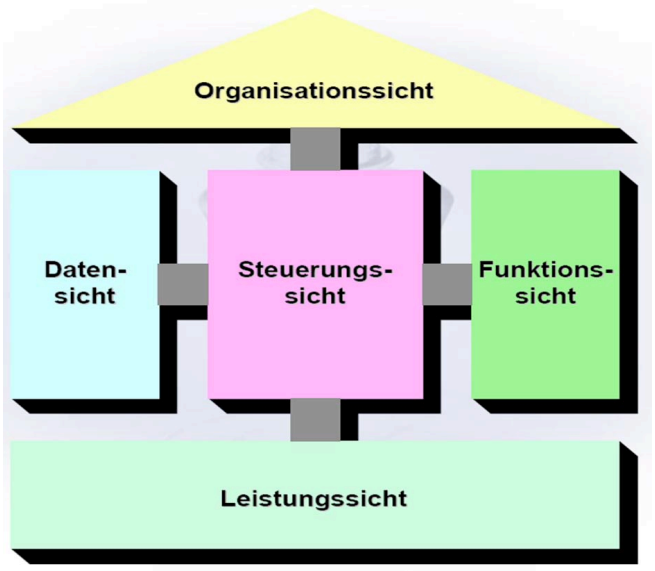


Abbildung 12 ARIS Haus,

„ARIS steht für die Architektur integrierter Informationssysteme. Das ARIS-Konzept ist ein allgemeiner Bezugsrahmen für die Geschäftsprozessmodellierung und stellt ebenen- und sichtenspezifische Modellierungs- und Implementierungsmethoden bereit.“ (Gadatsch, 2010)

Das ARIS Haus ist dabei die Strukturierung, um die einzelnen Modellierungen voneinander zu trennen, aber auch deren Verbindung zueinander zu zeigen. Normalerweise liegt der Fokus von ARIS auf betriebswirtschaftlichen Prozessen, welche aus Tätigkeiten bestehen, die eine Leistung als Endziel besitzen. Dies ist in unserem Projekt nicht der Fall, somit fällt die Leistungssicht raus.

5.2 Modellierung

Im Folgenden findet sich die Modellierung der einzelnen ARIS-Sichten. Auf eine detaillierte Modellierung der Klassen (mobile Applikation und Webservice) wurde bewusst verzichtet, da der Fokus auf der Beschreibung der Prozessabläufe lag.

Im Quelltext der Android Applikation und des Webclients werden alle Klassen in Kommentaren ausreichend kurz erläutert, ebenfalls sind alle wichtigen Quelltext-Teile kommentiert.

5.2.1 Organisationssicht

Da im Rahmen dieses Projektes kein Kunde vorhanden ist, entfällt die grafische Darstellung der Organisationssicht.

5.2.2 Datensicht

5.2.2.1 Mobile Applikation

Die Android Anwendung verfügt über keine Datenbank, zum einen rechtfertigt die geringe Anzahl an Daten keine Datenbank und zum anderen soll auf die Messdaten in sehr einfacher Form zugegriffen werden können. Des Weiteren muss die App nicht auf die Daten alter Messungen zurückgreifen.

Aus diesem Grund erfolgt die Speicherung der aktuellen Messdaten auf der SD-Karte des Endgerätes in Form einer einfachen CSV-Datei. Der Aufbau ist dabei folgender:

Fortlaufende Nr. ; RR-Intervall in Millisekunden ; Uhrzeit (HH:MM:SS)

Sämtliche Einstellungen der App (z.B. Auswahl des Bluetooth-Gurtes) werden von dem Android SDK automatisch gespeichert und geladen.

5.2.2.2 Webserver

Folgendes ER-Diagramm zeigt das serverseitige Datenbankmodell:

Abbildung 13 serverseitiges ER-Diagramm

Jede Messung verfügt über eine eindeutige Mess-ID, eine Start- und Endzeit und einen zugeordneten Benutzer. Falls kein User dem Server übergeben wird, verwendet er einen vorher erstellten User namens „Unknown“. Diese Angaben helfen für eine spätere Analyse der Daten, sowie auch einer Zuordnung der einzelnen Messdaten. Einer Messung (measures) können beliebig viele Messdaten (rates) zugeordnet werden.

Die Messdaten besitzen als Primärschlüssel die Sequenznummer und die Mess-ID, welche ebenfalls der Primärschlüssel der zugeordneten Messung ist. Zuzüglich besitzen die Messdaten jeweils ein RR Intervall und einen Zeitstempel.

5.2.3 Funktionssicht

Die Funktionssicht wird durch den folgenden Funktionsbaum dargestellt:

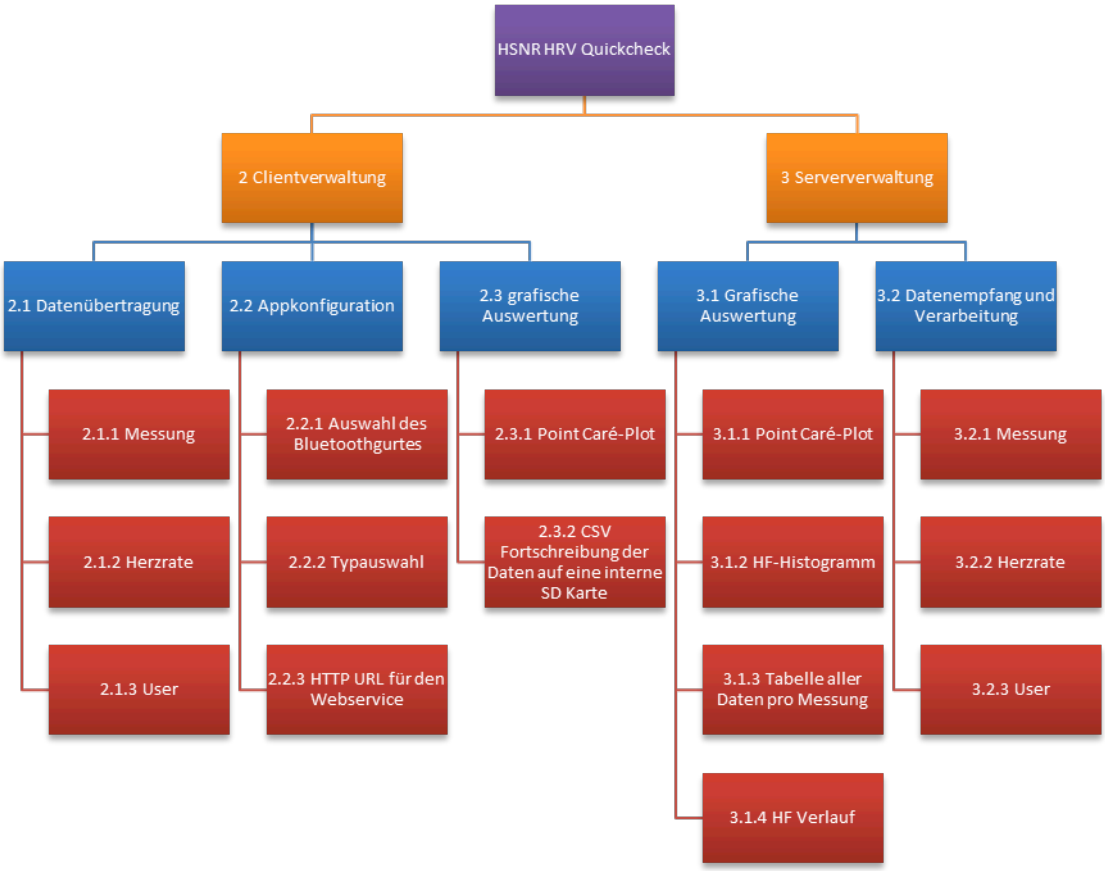


Abbildung 14 Funktionsbaum

Die Hauptfunktionen werden in kleinere Unterfunktionen zerlegt. Es werden Beziehungen zwischen den Funktionen verdeutlicht und die einzelnen Stufen auf unterschiedlichen Aggregationsstufen abgebildet.

5.2.4 Leistungssicht

Da keine materiellen Güter als Ergebnis dieses Projektes erwartet werden, ist es nicht möglich die Leistungssicht zu modellieren. Die Leistungssicht wird daher für das vorliegende Projekt nicht ausgefüllt.

5.2.5 Steuerungssicht

5.2.5.1 Auswahl der Modellierungssprache

Aufgrund der teilweise komplexen Prozesse wurden unterschiedliche Modellierungssprachen ausgewählt, um die Prozesse in der besten und verständnisreichsten Form darzustellen. Der Hauptprozess wurde in BPMN dargestellt und gibt einen Überblick über den Ablauf einer Messung bzw. die Abfrage der Daten mittels des Web-Clients.

Anschließend wird der Ablauf einer Messung aus Sicht der mobilen Applikation und der Prozessablauf einer Serveranfrage bzw. die Anzeige der Daten im Detail erläutert und grafisch dargestellt.

5.2.5.2 Modellierung

5.2.5.2.1 Hauptprozess

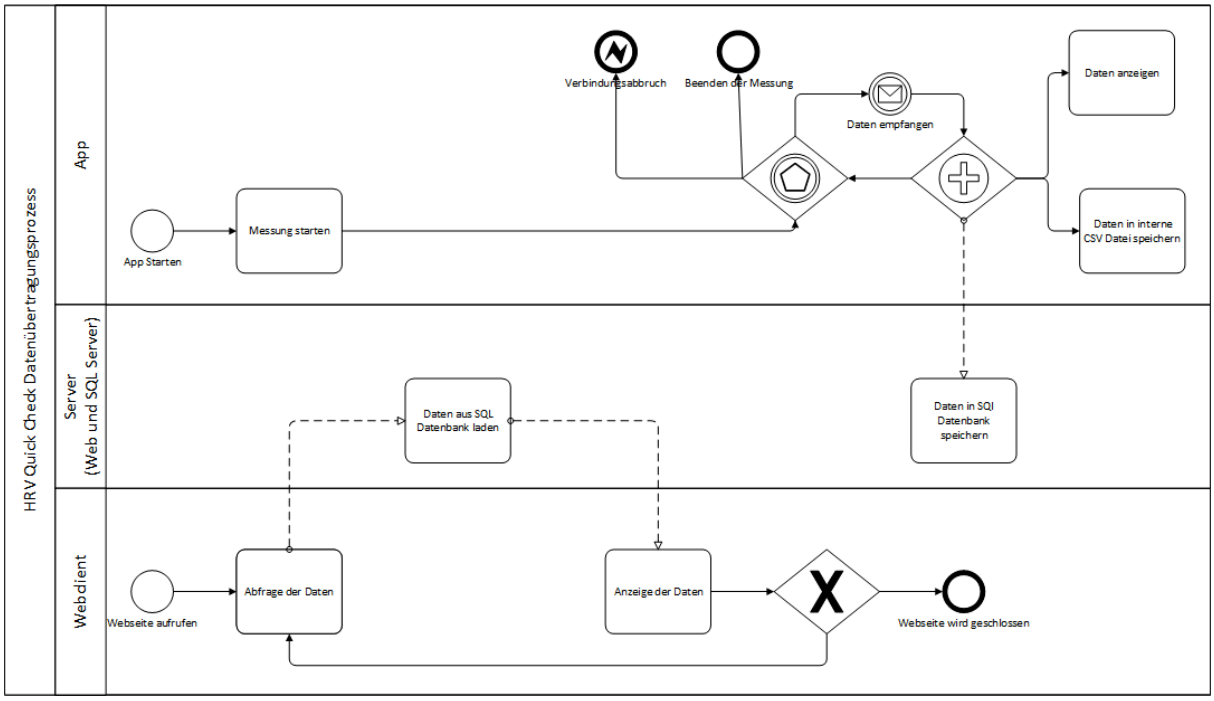


Abbildung 15 BPMN Datenübertragungsprozess

Die Messung startet sobald der Anwender, nach dem Start der App, den entsprechenden Start-Button drückt. Während die Messung läuft, können drei Ereignisse auftreten: Die Verbindung bricht aufgrund eines Verbindungsfehlers zwischen App und Brustgurt ab, der Nutzer beendet die Messung durch einen Klick auf „Stopp“ oder neue Daten wurden aus dem Brustgurt ausgelesen. Neue Daten werden dem Nutzer angezeigt, intern auf der SD-Karte gespeichert und an den Webserver geschickt. Der detaillierte Ablauf einer Messung aus Sicht der mobilen Applikation ist unter Punkt 5.2.5.2.2 dargestellt.

Empfängt der Server Daten von der mobilen Applikation, speichert dieser die Daten in eine MySQL Datenbank. Der allgemeine Ablauf des Servers wird unter Punkt 5.2.5.2.3 aufgezeigt.

Ein weiterer Prozessstart ist der Aufruf der Webseite. Es werden die Daten von der Datenbank abgerufen und zur Anzeige gebracht. Dieser Prozess wird im Punkt 5.2.5.2.4 beschrieben.

5.2.5.2.2 Ablauf Messung (App-Sicht)

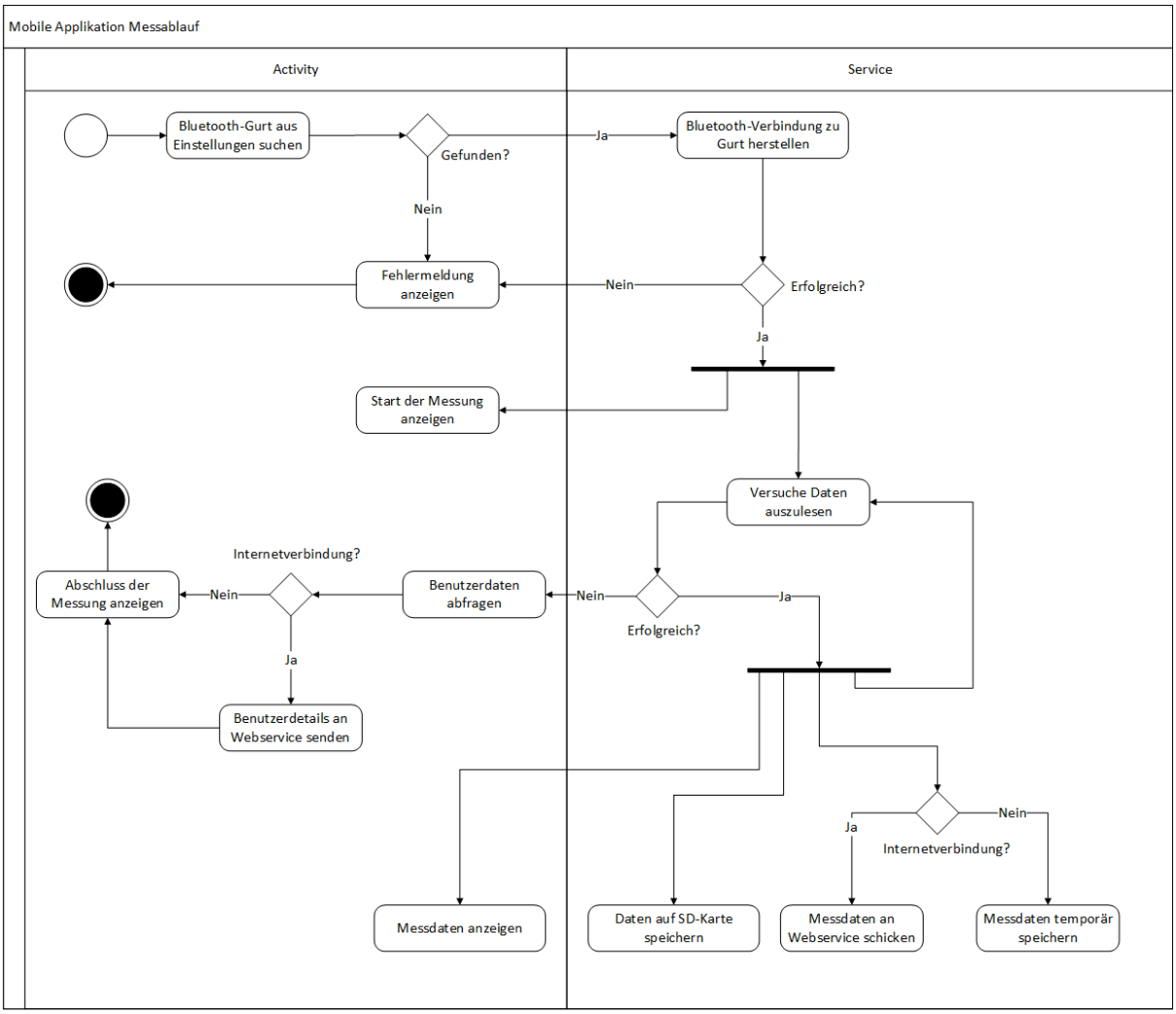


Abbildung 16 Messablauf (App-Sicht)

Das dargestellte Diagramm zeigt den Ablauf einer Messung aus Sicht der mobilen Applikation. Da die Applikation intern in zwei Komponenten aufgeteilt ist, werden die Aktivitäten in den entsprechenden Swimlanes dargestellt, um kenntlich zu machen, welcher Teil der Applikation die Aktivität ausführt.

Die Messung beginnt, wenn der Anwender den Start-Button in der App betätigt. Ein automatischer Start (z.B. durch Erkennung neuer Herzschläge) wurde in der ausgelieferten Version des Prototyps nicht realisiert. Im ersten Schritt wird der in den Einstellungen hinterlegte Brustgurt im Android-Gerät gesucht und ein Verbindungsaufbau wird gestartet. Wird der Gurt nicht gefunden bzw. kann die Verbindung nicht hergestellt werden, wird dem Anwender eine Fehlermeldung angezeigt und die Messung abgebrochen. Ist der Verbindungsaufbau erfolgreich, wird in einer Schleife versucht, Daten aus dem Byte-Strom des Bluetooth Gurts auszulesen. Wurden die Daten erfolgreich ausgelesen und verarbeitet (Verarbeitung siehe Punkt 6.2.2.1), werden die Daten an die Activity zur Anzeige gesendet und auf der internen SD-Karte in einer CSV-Datei gespeichert. Des Weiteren prüft die Anwendung, ob eine Verbindung zum Internet besteht. Ist eine Verbindung vorhanden, werden die neuen Messdaten an den Webservice per HTTP-POST gesendet. Besteht keine Internetverbindung werden die neuen Messdaten temporär in einer Variablen gespeichert und an den Webservice gesendet, sobald eine Internetverbindung vorhanden ist.

Konnten keine neuen Daten aus dem Bluetooth-Gurt ausgelesen werden, kann dies zwei Ursachen haben, die vom Ablauf her allerdings identisch sind und deswegen im Diagramm nicht unterschieden werden. Entweder trat ein Verbindungsfehler zum Bluetooth-Gurt auf (z.B. zu große Entfernung) oder der Benutzer hat die Messung durch einen Klick auf „Stopp“ beendet, wodurch die Verbindung zum Gurt geschlossen wird. In beiden Fällen können keine neuen Messdaten ausgelesen werden. Der Anwender wird über das Ende informiert und erhält die Möglichkeit, einen Namen, Alter und Geschlecht anzugeben. Daraufhin sendet die mobile Anwendung einen HTTP-POST an den Webservice, der die Endzeit der Messung und optional die Benutzerdaten enthält.

5.2.5.2.3 Prozess Server Ablauf

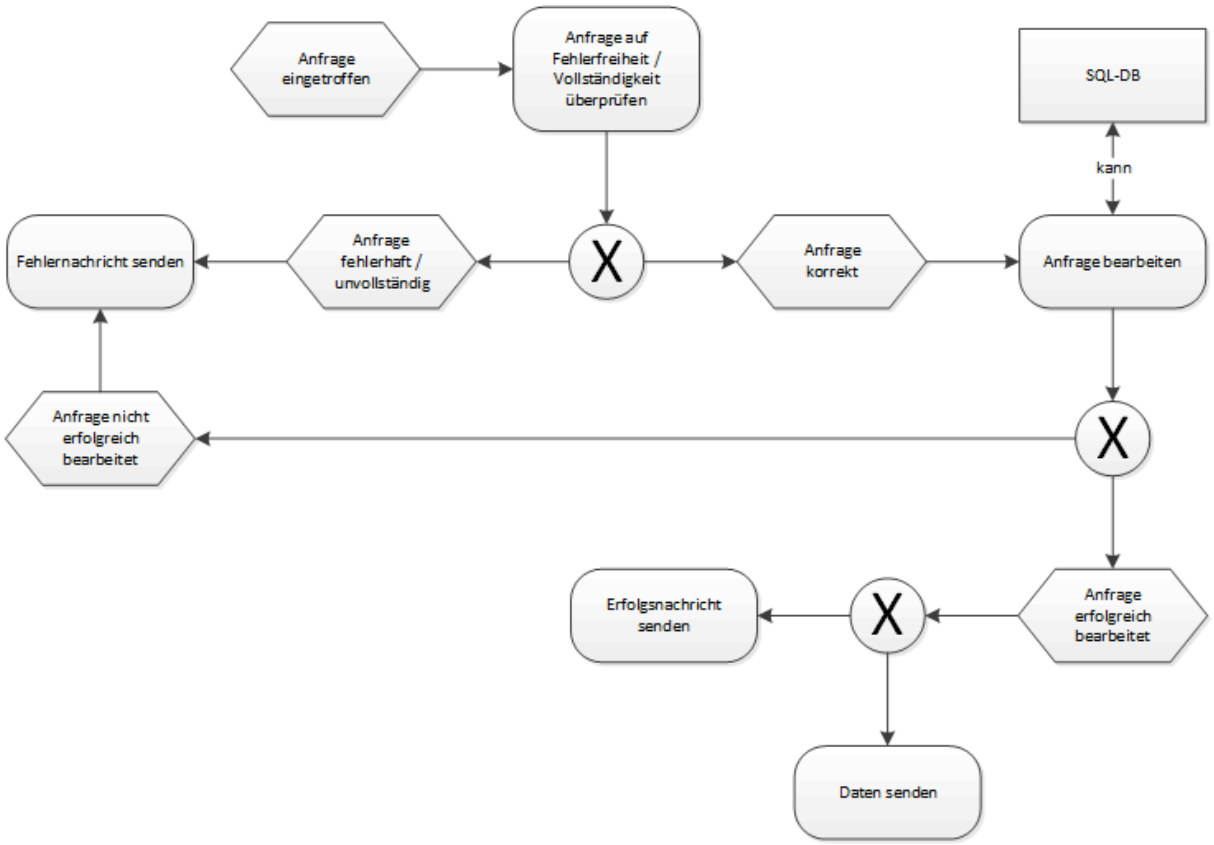


Abbildung 17 EPK Prozess Abfrage der Daten (Server)

Der Server als Webserver ist zustandslos und durchläuft somit für jede eingehende Anfrage den o. a. Prozess.

In dem Diagramm erkennt man, dass der Server mit Eintreffen einer Anfragen beginnt zu arbeiten. Diese Anfrage wird auf bekannte Fehler und vorhandenen Inhalt geprüft. Sollte ein Fehler aufgetreten sein oder für die Anfrage die passenden Inhalte fehlen, so sendet der Server eine Fehlernachricht an den anfragenden Client zurück. Anderenfalls versucht der Server die Anfrage zu bearbeiten. Je nach Anfrage greift der Webserver lesend und / oder schreibend auf den SQL-Server zu. Wenn die Bearbeitung der Anfrage fehlschlägt, sendet der Server wiederrum eine Fehlernachricht. Ansonsten sendet der Server - entsprechend der Anfrage - eine Erfolgsnachricht oder die gewünschten Daten als String an den Anfragenden.

5.2.5.2.4 Prozess Anzeige der Daten

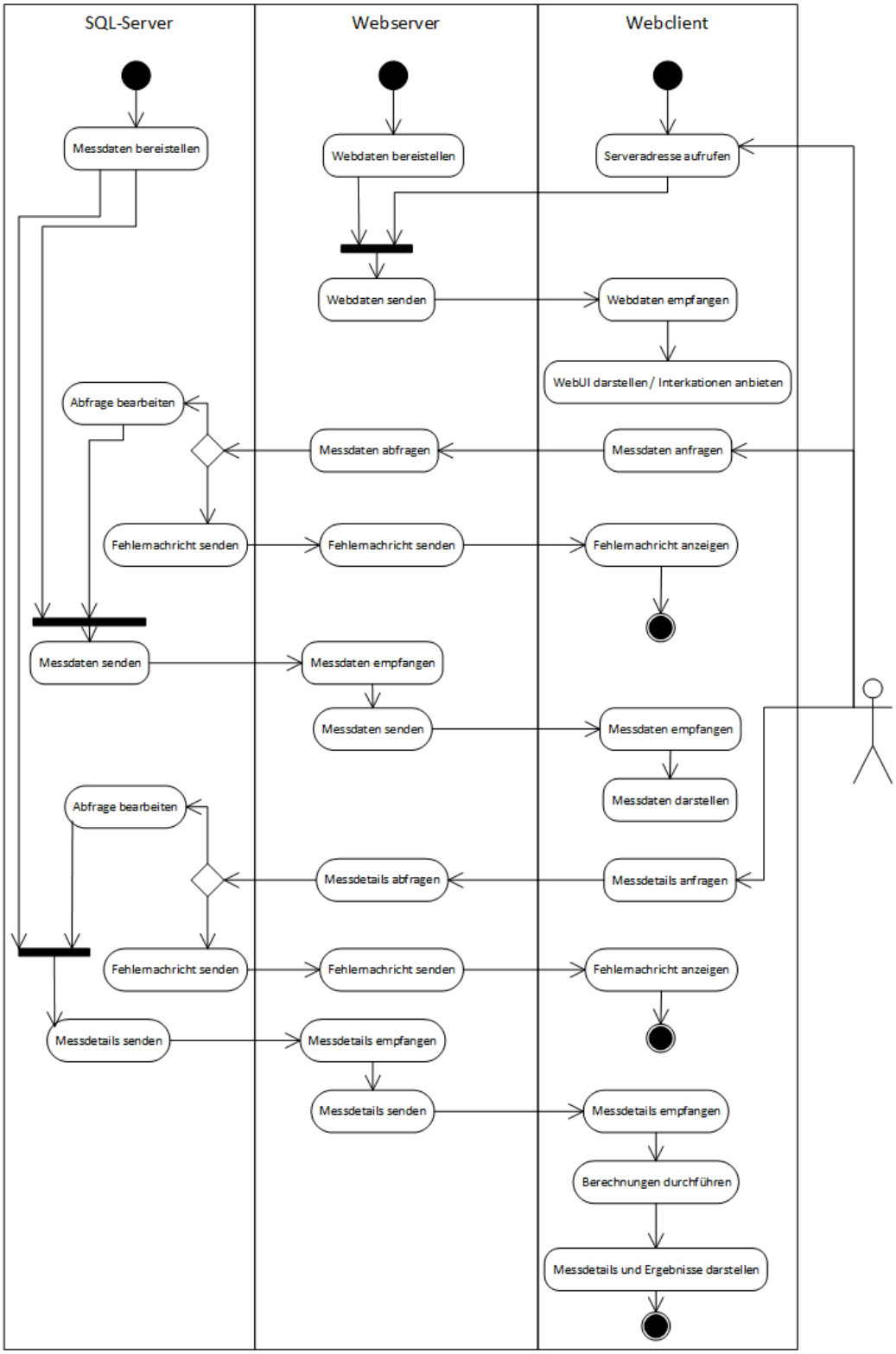


Abbildung 18 UML Aktivitätsdiagramm Anzeige der Daten

Das Aktivitätsdiagramm zeigt den Verlauf, wie der Benutzer mittels des Webclients die Messdaten und Details abrufen kann. Durch Aufrufen der Serveradresse in einem Browser werden die Dateien zur Darstellung abgerufen, heruntergeladen und i. d. R. vom Browser direkt umgesetzt und visualisiert. Somit steht das UI zur weiteren Benutzung zur Verfügung.

Anschließend hat der Benutzer die Möglichkeit, sich sämtliche Messungen anzeigen zu lassen. Sollte die Anfrage fehlschlagen, z.B. durch mangelhafte Berechtigung oder falsche

Verbindungsdaten, erhält der Webserver eine Fehlernachricht und reicht diese an den Webclient weiter. Ansonsten werden die Messungen in einer Liste dargestellt. Daraufhin kann der Benutzer die Details einer beliebigen Messung abrufen. Das Verhalten und der Ablauf der Anfrage laufen analog zur Anfrage der Messungen.

6 Realisierung

In diesem Abschnitt werden kurz die Besonderheiten der jeweiligen Komponenten der Umsetzung erläutert. Der Quellcode wird in digitaler Form der Dokumentation beigefügt.

6.1 Server

Neben dem physischen Gerät setzt sich der Server aus zwei Komponenten zusammen: Einem Webserver und einem SQL-Server.

Der Webserver besteht wiederum aus zwei Teilen. Ein Teil empfängt und verarbeitet HTTP POSTs, greift bei Bedarf auf den SQL-Server zu und gibt entsprechend Strings zurück. Der andere Teil ist der Webclient, welcher bei Aufruf der Serveradresse – im Normalfall mit einem Browser – heruntergeladen wird.

6.1.1 Eingesetzte Werkzeuge

Zur Realisierung des SQL-Servers wurde MySQL, als Datenbankverwaltungssystem eingesetzt. Administration und Kontrollen wurden mit phpMyAdmin durchgeführt.

Der Teil des Webservers, welcher die HTTP POSTs verarbeitet wurde in PHP geschrieben. Die Logik des Webclients wurde mit HTML, CSS und JavaScript umgesetzt. Zusätzlich wurden die JavaScript Bibliotheken jQuery[4] – für einfachere DOM-Manipulation – und flot[5] – für die vereinfachte Erzeugung von grafischen Darstellungen – eingesetzt. Um die Programmierung des JavaScript Codes zu vereinfachen wurde die Programmiersprache CoffeeScript[6], welche in JavaScript transcompiliert wird, eingesetzt.

Zur Compilierung von CoffeeScript wurde Node.js[7] benutzt. Die Programmierung fand ausschließlich in Sublime Text[8] statt. Datenübertragungen zum Server liefen über SFTP und FTPS.

[4] jquery (http://jquery.com/)

[5] flot (http://www.flotcharts.org/)

[6] CoffeeScript (http://coffeescript.org/)

[7] Node.js (http://nodejs.org/)

[8] Sublime Text (http://www.sublimetext.com/)

6.1.2 Besonderheiten

Im Bereich des Servers wurde vor allem auf die Erweiterbarkeit und dynamische Anbindung Wert gelegt. So wurden sowohl in PHP, als auch in JavaScript Klassenlogiken implementiert, die es ohne weiteres erlauben die Webapplikation um weitere Teile zu erweitern. Zusätzlich wurden sämtliche Berechnungen zur Darstellung in JavaScript und die Anbindung des Webclients mit AJAX realisiert. Dies ermöglicht einerseits bei einer späteren Weiterentwicklung Daten und Visualisierungen auch schon während der Messung anzupassen und andererseits ein effizienteren Einsatz auf mobilen Endgeräten mit geringeren Datenraten.

Da die vorliegende Lösung nur ein Prototyp ist, wurde auf verschiedene Problemstellung der Verfügbarkeit, Integrität, Authentizität und Vertraulichkeit nicht näher eingegangen.

6.2 Mobile Applikation

Die mobile Applikation wurde in Form einer Android Anwendung realisiert, die sowohl auf Smartphones, als auch Tablets ab der Android Version 4.0 (API-Level 14) lauffähig ist. Dabei passt sich die Darstellung der Messdaten dynamisch der Größe des Displays an.

Die Messdaten werden in der MainActivity[9] in Form von Textlabels und einem Poincaré-Plot angezeigt. Neben der MainActivity, gibt es eine PreferenceActivity, über die der Nutzer diverse Einstellungen (z.B. Gurtauswahl) vornehmen kann. Die Einstellungen werden durch das Android-SDK automatisch im App-Verzeichnis gespeichert und beim Start der App ausgelesen. Beim Abschluss einer Messung bekommt der Anwender einen Dialog zur Eingabe der Benutzerdaten angezeigt. Die Eingabe ist optional.

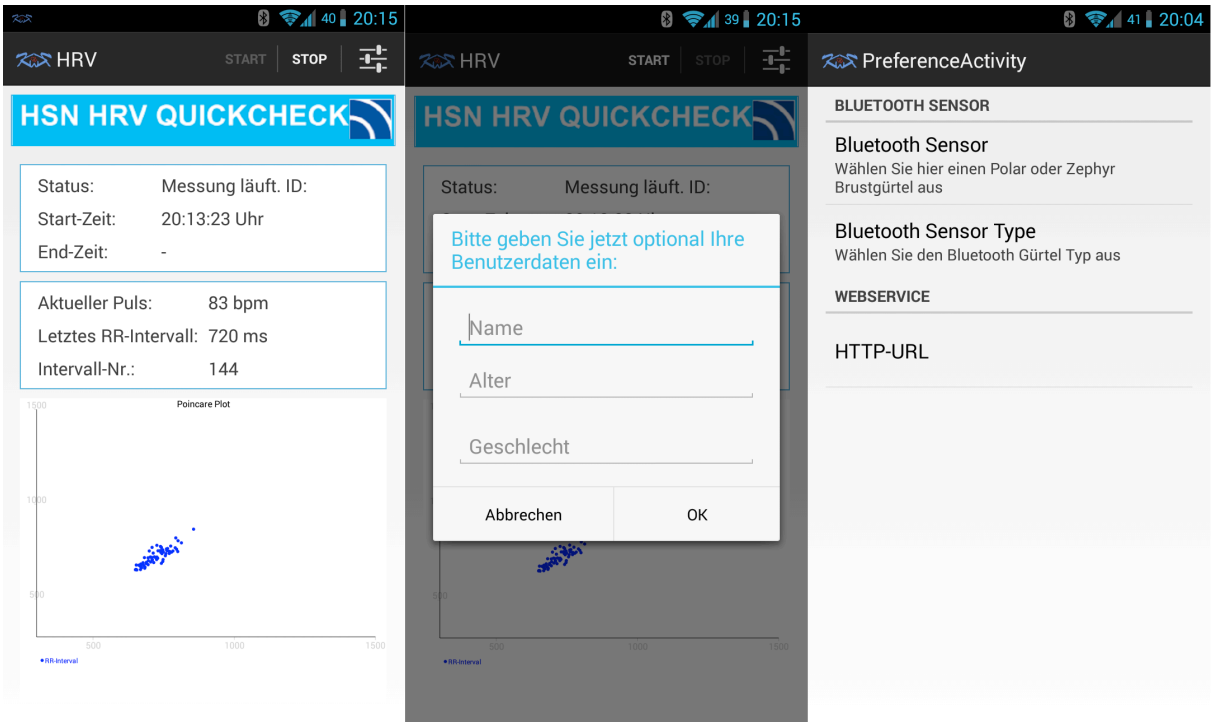


Abbildung 19 Screenshots Android App

[9] Eine Activity ist der sichtbare Teil der App (die Anzeige)

Die Messung selber erfolgt aus gewissen Gründen (siehe Punkt 6.2.2) nicht in der MainActivity, sondern in einem Android-Service. Hierbei wurde auf eine lose Kopplung geachtet, sodass der Mess-Service keine / kaum Abhängikeiten zu den Activitys / der GUI hat und dadurch mit wenig Arbeit in eine andere Applikation übernommen werden kann.

6.2.1 Eingesetzte Werkzeuge

Für die Entwicklung der Applikation wurden das offizielle Android-SDK inkl. Eclipse genutzt. Für den ersten Versuch der Bluetooth-Anbindung wurde der Quelltext der Open-Source App MyTracks von Google analysiert und Quelltextteile der Bluetoothanbindung und der Datenanlyse übernommen. Im Laufe des Projektes wurden diese Teile aber noch modifiziert, so dass die eigentliche Entwicklungsleistung (Empfang und Verarbeitung der RR-Intervalle) von den Projektmitgliedern geleistet wurde und nur das Grundgerüst der Bluetoothanbindung von Google stammt. Dieses wird auch in den offiziellen Android Tutorials verwendet.

Desweiteren wurde die Open-Source Bibliothek aChartEngine[10] für die Darstellung des Poincaré Plots eingesetzt. Die Bibliothek bietet die Möglichkeit alle gängingen Diagramme zu zeichen. Bei dem Poincaré Plot handelt es sich um ein Scatter-Chart, das mit nur wenigen Zeilen Quelltext nach seinen Wünschen angepasst werden kann. Das Hinzufügen neuer Werte geschieht dabei mit nur einer Zeile Code. Durch den Einsatz von aChartEngine ist der Aufwand der grafischen Darstellung signifikant gesunken.

6.2.2 Besonderheiten

Bei der Entwicklung der mobilen Applikation gab es in zwei Bereichen Besonderheiten, die bei der Entwicklung berücksichtigt werden mussten und im Folgenden näher beschrieben werden.

6.2.2.1 Anbindung Polar Bluetooth Gurt

Die erste Schwierigkeit bestand in der Anbindung des Polar Bluetooth Gurts, da nicht nur die Herzraten, sondern auch die RR-Intervalle ausgelesen werden mussten. Leider gab bzw. gibt es vom Hersteller keine API, über die sich die Daten abfragen lassen. Nach einiger Recherche zeigte sich, dass die Open-Source App MyTracks von Google, die aktuelle Herzrate aus dem Polar Gurt auslesen und anzeigen kann. Auf Basis der Analyse des Quelltextes von MyTracks wurde eine erste prototypische App entwickelt, die die aktuelle Herzfrequenz anzeigt.

Technisch gesehen wird dafür der Byte-Stream des Polar Gurtes ausgelesen, ein gültiges Paket kann dabei zwischen 8 und 16 Byte groß sein. In dem Parsing-Algorithmus (also die Art und Weise, wie die Daten aus dem Polar Gurt gelesen und interpretiert werden) von Google gab es jedoch zwei gravierende Schwachstellen, die die sofortige Nutzung für das HRV-Projekt unmöglich machten.

Zum einen wurde nur die Herzfrequenz ausgelesen, der Rest der Daten (wie die für das Projekt wichtigen RR-Intervalle) wurde verworfen. Zum anderen gab es in dem Algorithmus einen Fehler, der dazu führte, dass nur jedes 2-4 Paket aus dem Byte-Stream berücksichtigt wurde, der Rest an Daten wurde verworfen.

[10] https://code.google.com/p/achartengine/

Durch einen Hinweis in den Kommentaren des Quelltextes und genauerer Analyse des Byte-Stromes konnten die Positionen der RR-Intervalle identifiziert und anschließend ausgelesen werden. Es zeigte sich, das ein RR-Intervall 2 Bytes lang ist und pro Paket 1-4 RR-Intervalle gesendet werden (Paketlänge 8-16 Bytes).

Um das zweite Problem („überlesene“ / ignorierte Datenpakete) zu beheben, musste der Algorithmus in einigen Bereichen umgeschrieben werden. Das Problem bestand darin, dass immer 16-Bytes aus dem Stream eingelesen, dann das erste Paket verarbeitet und der Rest der Daten verworfen wurde. Wurden jetzt 16-Bytes aus dem Gurt ausgelesen, das erste Paket war aber nur 8-Byte groß, wurde das nächste Paket also einfach ignoriert und verworfen. Zur Lösung des Problems wurde der Algorithmus so umgeschrieben, dass immer 32-Bytes eingelesen werden (also 2 ganze Pakete mit max. Größe, 2x16 Byte) und alle Daten berücksichtigt werden. Das folgende Aktivitätsdiagramm zeigt den genauen Ablauf des Parsing-Algorithmus.

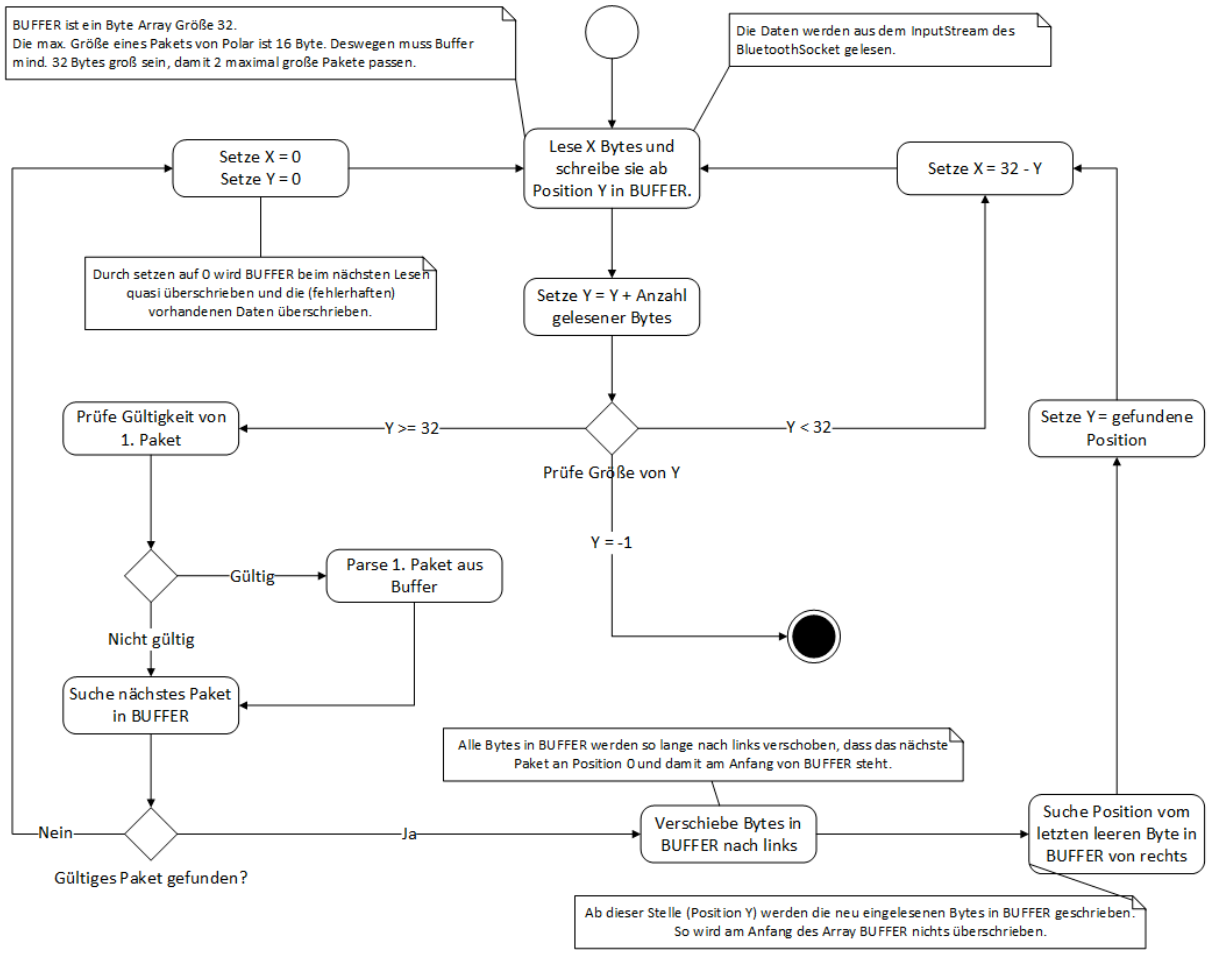


Abbildung 20 Aktivitätsdiagram Parsing-Algorithmus

6.2.2.2 „Robustheit“ und Autonomie des Mess-Service

Trotz dessen, dass die mobile App dieses Projekts ein Prototyp sein sollte, wurde einem Bereich eine besondere Bedeutung zugemessen.
Die Messung der HRV-Daten wurde in einen Android-Service ausgelagert, damit auch bei geschlossener Activity (Activity in den Hintergrund, Bildschirm schaltet sich aus, etc.) oder einem eingehenden Anruf die Messung ungehindert weiter laufen kann. Dabei wurde zusätzlich darauf geachtet, dass eine lose Kopplung existiert, also der Service kaum / keine

Abhängigkeiten zur GUI hat und damit leicht in anderen Applikationen eingesetzt werden kann.

Dafür wird der Android Service von der Activity beim Start der App explizit gestartet. Ist die Activity im Vordergrund (wird also dem Nutzer angezeigt), bindet sie sich an den Service. Über einen Message-Handler kann so eine Zwei-Wege-Kommunikation stattfinden und der Service z.B. die Messdaten zur Anzeige an die Activity senden.

7 Lessons Learned

Trotz des positiven Ergebnises des Projekts gab es auf dem Weg dahin verschiedene Schwierigkeiten, die es zu überwinden galt.

Ein Punkt, der häufiger und gravierender eintrat als geplant, war der Aufwand anderer Tätigkeiten, welche nicht zum Projekt gehörten. Besonders die Arbeitsverhältnisse und das Studium der Projektmitglieder hat mehr Zeit gefordert, als zuvor erwartet wurde. Infolgedessen war das kontinuierliche Arbeiten an dem Projekt nicht mehr gewährleitstet und Zeit ging verloren. Dies führte auch zu der geringfügigen Verzögerung im Zeitplan.

Scrum als Vorgehensmodell war für das Projekt eher hinderlich als hilfreich. Aufgrund nebenläufiger Projekte und der Berufe der Projektmitglieder war ein tägliches Arbeiten am Projekt meist nicht möglich. Zusätzlich standen die Aufgabenbereiche der einzelnen Personen frühzeitig fest, was ein autonomes Handeln ermöglichte und förderte. Dadurch wurden die Daily-Meetings obsolet und der zum Scrum gehörende Overhead, wie Rollenverteilung, Stories, Sptrints, etc. wandelten sich von unterstützenden Tools zu Zeitfressern.

Kurz nach dem Projektstart wurden bereits verschiedene programmiererische Versuche und Prototypen entwickelt. Dies sollte einerseits die Machbarkeitsstudie unterstützen bzw. abdecken und andererseits zum Testen der Technologien und möglichen Umsetzungen dienen. Wegen der raschen erfolgreichen Entwicklung fehlte ein passender Breakpoint, an dem mit der Modellierung hätte begonnen werden sollen. Dies hatte für die Programmierung keine Nachteile, weil die Schnittstellen bereits zuvor festgesetzt wurden und die einzelnen Programme relativ autark arbeiten. Auswirkung hatte dieses Vorgehen vor Allem auf die Dokumentation, weil im Nachhinein die Modellierung der Programmierung angepasst werden musste.

8 Anhang

8.1 Implementierungsanleitung

Im Folgenden wird kurz erläutert, welche Ressourcen zur Implementierung des Projektes notwendig sind und wie dabei vorzugehen ist.

8.1.1 Benötigte Ressourcen

1. Android Smartphone
    a. GPS fähig
    b. WLAN / 3G
    c. Android ab Version 4.0
2. Webserver
    a. FTP Server
    b. PHP Version 5
3. MySQL Server
4. PC mit folgenden Eigenschaften
    a. FTP Client (Installation Server)
    b. Web Browser mit aktivierten Javascript

8.1.2 Notwendige Schritte

Für die Installation des HRV Quick Checks sind sowohl Arbeiten am Client, wie auch am Server notwendig.

8.1.2.1 Installation Server

Die Installation des Servers setzt voraus, dass ein SQL-Server und ein HTTP-Server zur Verfügung stehen oder selbstständig installiert werden können.

Der SQL-Server muss, je nach vergebenen Berechtigungen, vorbereitet werden. Der Name der gewünschten Datenbank kann frei gewählt werden und muss später in den Konfigurationen des Webclients eingestellt werden (s. 8.2.2). In der Datenbank können bereits die Tables „measures“, „rates“ und „users“ angelegt werden. Sollten die Tables oder die Datenbank nicht existieren, würde der Webserver bei der Testverbindung (s. 8.2.2) versuchen diese anzulegen.

Die Daten des Webservers müssen über den HTTP Server verfügbar gemacht werden, so dass von Außen ein Zugriff erfolgen kann. Zusätzlich sollte je nach System eine Schreibberechtigung für den Ordner „data“ zugewiesen werden.

8.1.2.2 Installation App

Die Distribution der App geschieht nicht über einen App-Market, sondern über eine APK-Datei. Diese kann einfach auf das Smartphone übertragen werden und dort per Klick installiert werden.

Um die App installieren zu können, muss auf dem Android-Gerät die Installation unbekannter Quellen zugelassen werden, weitere Informationen sind der Bedienungsanleitung des Android-Gerätes zu entnehmen.

8.1.3 Implementierungsaufwand

Die gesamte Implementierung der Applikation dauert ungefähr 1h. Hierbei wurde berücksichtigt, dass eventuell Einstellungen am Webserver vorgenommen werden müssen und eine Testmessung durchgeführt wurde. Prinzipiell ist der Aufwand sehr gering und es sollten keine größeren Probleme auftauchen.

8.2 Benutzeranleitung

8.2.1 Mobile Applikation

Nachdem die App installiert wurde, muss der Bluetooth Gurt im Smartphone bzw. Tablet „gepaired“ werden. Hierbei verbinden sich beide Geräte und „kennen“ sich dadurch. Dieser Vorgang muss nur einmal pro Gurt und Endgerät geschehen.

Wie genau der Pairing-Vorgang abläuft ist je nach Endgerät und verwendetem Gurt unterschiedlich. Die genaue Anleitung ist daher der Benutzeranleitung des Endgerätes bzw. des Bluetooth-Gurtes zu entnehmen.

Nachdem der Pairing-Vorgang erfolgreich abgeschlossen ist, kann die HRV-App gestartet werden. Über das Symbol oben rechts bzw. über den Menü-Button des Gerätes wird der Einstellungsdialog angezeigt.

Abbildung 21 Button Einstellungen

Im ersten Feld muss nun der zuvor gepairte Gurt ausgewählt werden.

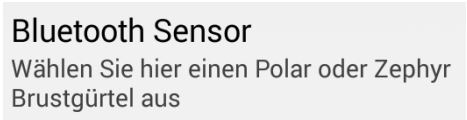


Abbildung 22 Bluetooth-Gurt Auswahl

Das zweite Feld muss aktuell nicht geändert werden, da die abgelieferte Version nur die Verbindung zu einem Polar-Gurt unterstützt.

Im dritten Feld muss die URL des Webservices angegeben werden. Standardmäßig ist dieses Feld mit dem Testserver der Projektmitglieder belegt. Diese Vorbelegung kann in dem Android-Projekt geändert werden.

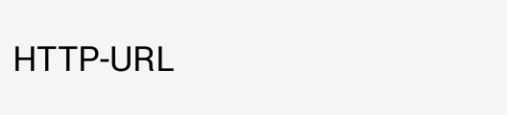


Abbildung 23 Einstellung HTTP-URL

Nachdem alle Einstellungen vorgenommen wurden, kann über den Start- bzw. Stopp Button die Messung gesteuert werden.

Hinweis: Schlägt der Start der Messung mit einem Verbindungsfehler fehl, sollte der Batteriestand des Gurtes und der Kontakt zwischen Gurt und Messperson geprüft werden. Es ist wichtig, dass die Kontakte des Messgurtes ausreichend feucht sind, da sonst keine Daten empfangen werden können.

8.2.2 Webclient

Jede Ansicht besitzt eine Kopfleiste, den sogenannten Header. In ihm sind der Name des aktuellen Bereichs und das Logo der Hochschule Niederrhein zu sehen, welches gleichzeitig ein Link zu deren Seite ist (siehe Abbildung 24).

Zusätzlich werden nach Notwendigkeit eine Zurück-Schalftfläche (siehe Abbildung 25) und eine Neuladen-Schaltfläche (siehe Abbildung 26) angeboten. Die Zurück-Schaltfläche bietet die Möglichkeit auf die vorhergehende Seite zurück zu navigieren. Die Neuladen-Schaltfläche lädt die aktuelle Seite neu.

Die meisten Elemente verfügen über eine kleine Hilfeansicht. Diese erscheint, sobald der Cursor über dem Element stehen bleibt. Die Hilfe zeigt entweder wofür das Element bestimmt ist (siehe Abbildung 25) oder wichtige Details (siehe Abbildung 27).

Abbildung 24 Hauptmenü

Zu Beginn erscheint der in der Abbildung 24 zu sehende Startbildschirm. In ihm hat der Anwender die Möglichkeit zu den Konfigurationen zu kommen oder die Übersicht über die Messungen aufzurufen.

Abbildung 25 Konfigurationsmenü

In den Konfigurationen kann der Anwender die Daten des SQL-Servers hinterlegen. Beim Aufruf der Seite werden die aktuellen Daten angezeigt. Zu den Daten gehören (v. o. n. u.):

- Server: Die Adresse zum SQL-Server
- Datenbank: Der Name der Datenbank
- Benutzer: Der Benutzer, der auf die Datenbank zugreifen darf
- Passwort: Das Benutzerpasswort

Nach der Eingabe oder Änderung der Daten können mittels der Save-Schaltfläche die Daten gespeichert werden. Mit der Test-Schaltfläche wird probeweise eine Verbindung zum SQL-Server aufgenommen. Sollten die Datenbank oder die Tabellen auf dem SQL-Server nicht existieren, wird versucht diese anzulegen. Unterhalb der Schaltflächen erhält der Anwender eine kurze Rückmeldung, ob die Aktion geglückt oder fehlgeschlagen ist.

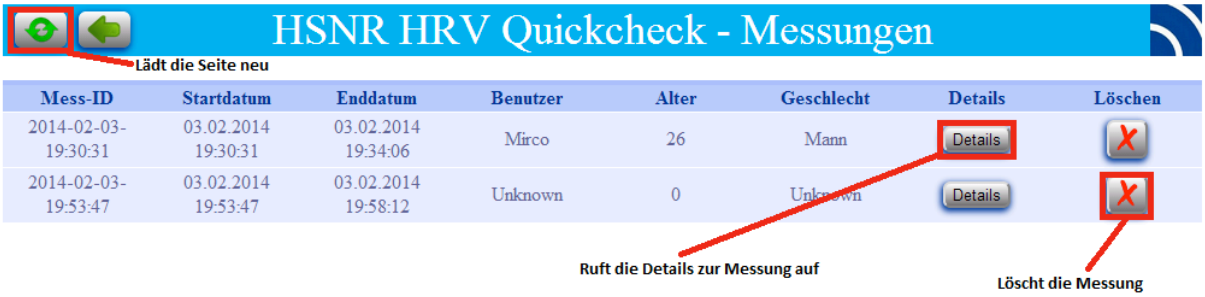


Abbildung 26 Messübersicht

Ist der SQL-Server erreichbar und liegen entsprechend Messungen vor, können diese in der Messungen Übersicht aufgelistet werden (siehe Abbildung 26). Zu den groben Informationen, wie die Mess-ID, Start, Ende und Angaben zum Benutzer, besteht die Möglichkeit die einzelnen Messungen zu löschen oder die Details abzurufen.

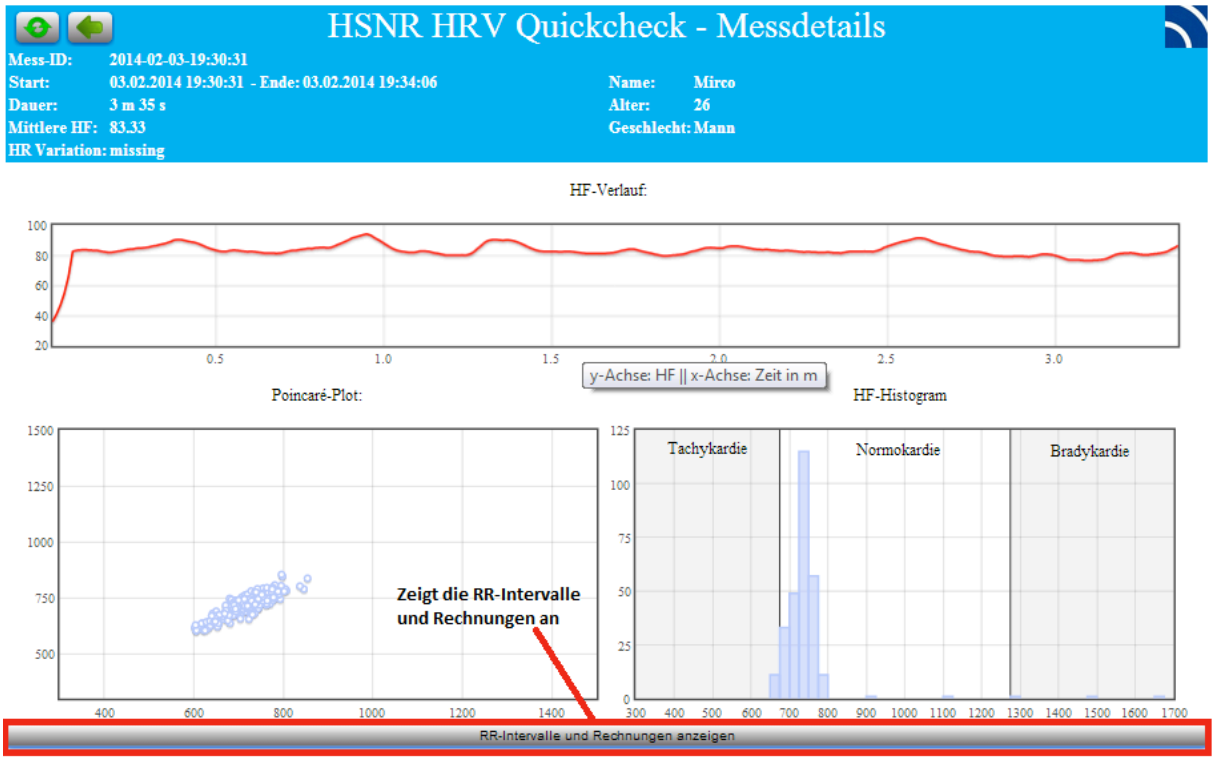


Abbildung 27 graphische Darstellung der Messung

Werden die Details abgerufen, erscheint, wie in Abbildung 27 zu sehen, eine Übersicht über die Messung mit graphischen Darstellungen.

RR-Intervalle und Rechnungen ausblenden

Sequenznummer	RR (i)	RR (i-1)	Zeit (ms)	RR	HF	Varianz	Standardabweichung	Variationskurve
76	674	704	59795	707.09	84.85	326.81	18.08	2.56
77	703	674	61075	705.55	85.04	310.79	17.63	2.5
78	716	703	61075	704	85.23	249.82	15.81	2.25
79	747	716	62356	707.27	84.83	402.74	20.07	2.84
80	680	747	62356	704.55	85.16	462.25	21.5	3.05
81	642	680	63635	696.91	86.09	717.72	26.79	3.84
82	635	642	63635	688.45	87.15	906.79	30.11	4.37
83	617	635	64910	681.73	88.01	1325.11	36.4	5.34
84	602	617	64910	674	89.02	1840.73	42.9	6.36
85	605	602	66201	665.91	90.1	2171.72	46.6	7
86	606	605	66201	657	91.32	2286.73	47.82	7.28
87	621	606	67484	652.18	92	2355.06	48.53	7.44

Abbildung 28 Liste aller RR-Intervalle

Bei Betätigen der Schaltfläche am Ende der Seite, erscheint die Liste aller RR-Intervalle und einigen Rechnungen, die u.a. für die graphischen Darstellungen notwendig sind (siehe Abbildung 28).

8.3 Abbildungsverzeichnis

8.4 Tabellenverzeichnis

8.5 Abkürzungsverzeichnis

Bzw	Beziehungsweise
v. o. n. u.	Von oben nach unten
API	application programming interface

8.6 Literaturverzeichnis

Gadatsch, Andreas. 2010. Grundkurs Geschäftsprozess-Management. Wiesbaden : Vieweg Teubner, 2010.

IMD. *http://imd-hrv.de.* unbekannt. unbekannt unbekannt. http://cdn.imd-hrv.de/downloads/Fachseminar_IMD_HRV.pdf (Zugriff am 14. 1 2014).