

SOFTWARE ENGINEERING PRAKTIKUM

CAB-TRAIN

Software-Entwicklungspraktikum (SEP)

Sommersemester 2012

G r o b e n t w u r f



Auftraggeber

Technische Universität Braunschweig

Peter L. Reichertz Institut für Medizinische Informatik

Prof. Dr. Reinhold Haux

Mühlenpfordstraße 23

38106 Braunschweig


Betreuer: Markus Wagner, Thomas Franken

Auftragnehmer:

Name	E-Mail-Adresse
Aisha Shnati	aishash777@googlemail.com
Arnaud Chevrolier	arnaud.chevrolier@tu-bs.de
Bianca Oppermann	BiancaOppermann@gmx.de
Christoph Harburg	c.harburg@tu-bs.de
Svenja Molitor	svenja_moli@hotmail.de
Virnadita Sjahran	virnadita@yahoo.de

Braunschweig, 23.05.2012

Versionsübersicht

Version	Datum	Autor	Status	Kommentar
v0.1	16.05.2012	s. Auftragnehmer		Erste Version, Texte in vorgegebenes Layout eingefügt.
v0.2	18.05.2012	s. Auftragnehmer		In erste Version wurden Graphiken eingebunden.
v0.3	21.05.2012	s. Auftragnehmer		In Version zwei wurden Graphiken überarbeitet.
v0.4	22.05.2012	s. Auftragnehmer		In Version drei wurden Texte überarbeitet.
v0.5	23.05.2012	s. Auftragnehmer		Endgültige Formatierung.

Inhaltsverzeichnis

1	Einleitung	6
1.1	Projektdetails	7
2	Analyse der Produktfunktionen	11
2.1	Analyse von Funktionalität /F100/ und /F200/: Erkennen eines Sensors; Herstellen einer Verbindung	11
2.2	Analyse der Funktionalität /F250/: Accountverwaltung	12
2.3	Analyse von Funktionalität /F300/, /F400/und /F500/: Auswählen der Übung; Erklären der Übung; Starten einer Übung	14
2.4	Analyse von Funktionalität /F600/: Anzeigen des Übungsablaufs	15
2.5	Analyse von Funktionalität /F700/: Speichern von Daten	16
2.6	Analyse von Funktionalität /F800/: Beenden einer Übung	17
2.7	Analyse der Funktionalität /F1000/: Auswertung der Übung	19
2.8	Analyse der Funktionalität /F1100/: Programmstatus ändern	20
2.9	Analyse der Funktionalität /F1200/: Framework	21
2.10	Analyse von Anforderung /Q30/: Berechnungszeit	21
2.11	Analyse von Anforderung /Q50/: Plattformunabhängigkeit	21
2.12	Analyse von Anforderung /Q60/: Benutzerfreundlichkeit	21
2.13	Analyse von Anforderung /Q70/ und /Q100/: Code-Conventions	22
2.14	Analyse von Anforderung /Q80/ und /Q90/: Dokumentation	22
2.15	Analyse von Anforderung /Q120/: Fehlertoleranz bezüglich Benutzereingabe	23
3	Resultierende Softwarearchitektur	24
3.1	Komponentenspezifikation	24
3.2	Schnittstellenspezifikation	25
3.3	Protokolle für die Benutzung der Komponenten	26
3.3.1	Controller	26
3.3.2	GUI	27
3.3.3	Kommunikation	28
3.3.4	Position	29
3.3.5	Wii-Balance Board	29
3.3.6	Auswertung	30
3.3.7	Speicher	31

3.3.8 Übung	31
4 Verteilungsentwurf	33

Abbildungsverzeichnis

1.1	Aktivitätsdiagramm: Verbindung herstellen	7
1.2	Aktivitätsdiagramm: Accountverwaltung	8
1.3	Aktivitätsdiagramm: Speichern und Übungsablauf	9
1.4	Aktivitätsdiagramm: Beenden	10
2.1	Sequenzdiagramm: Erkennen eines Sensors, Herstellen einer Verbindung	11
2.2	Sequenzdiagramm: Accountverwaltung für richtige Benutzereingabe	12
2.3	Sequenzdiagramm: Accountverwaltung für falsche Benutzereingabe	13
2.4	Sequenzdiagramm: Auswählen, Erklären, Starten einer Übung	14
2.5	Sequenzdiagramm: Anzeigen des Übungsablaufes	15
2.6	Sequenzdiagramm: Speichern von Daten	16
2.7	Sequenzdiagramm: Beenden einer Übung ohne Unterbrechung	17
2.8	Sequenzdiagramm: Beenden einer Übung mit Unterbrechung	18
2.9	Sequenzdiagramm: Auswertung der Übung	19
2.10	Sequenzdiagramm: Programmstatus ändern	20
3.1	Komponentendiagramm: Komponentenspezifikation	24
3.2	State Chart: Controller	26
3.3	State Chart: GUI	27
3.4	State Chart: Kommunikation	28
3.5	State Chart: Position	29
3.6	State Chart: Wii-Balance Board	29
3.7	State Chart: Auswertung	30
3.8	State Chart: Speicher	31
3.9	State Chart: Übung	31
4.1	Verteilungsentwurf	33

1 Einleitung

Im Allgemeinen beschäftigt sich der Grobentwurf mit der Struktur einer Testimplementierung für das Projekt “CAB-Train: Framework für computerassistierendes Bewegungstraining“.

Zusätzlich wird im Grobentwurf zweitrangig auf das zu implementierende Framework eingegangen.

Anhand von diversen Diagrammen wird versucht einen umrisshaften Überblick über diese Strukturen zu verschaffen.

Um zunächst ein Training durchführen zu können, ist es notwendig, eine Bluetoothverbindung zwischen den beiden Komponenten, dem Rechner und dem Wii-Balance-Board, herzustellen. Dadurch wird gewährleistet, dass beispielsweise Steuerbefehle oder Daten untereinander ausgetauscht werden können.

Eine nähere Betrachtung der weiteren Komponenten und ihrer Verteilung befindet sich in dem Kapitel 3.

Zu Beginn wird im Unterkapitel 1.1, durch verschiedene Aktivitätsdiagramme, ein kurzer Überblick über den Programmablauf gegeben.

1.1 Projektdetails

Die Hauptanwendungen werden durch die Aktivitätsdiagramme in den Abbildungen 1.1-1.4 dargestellt.

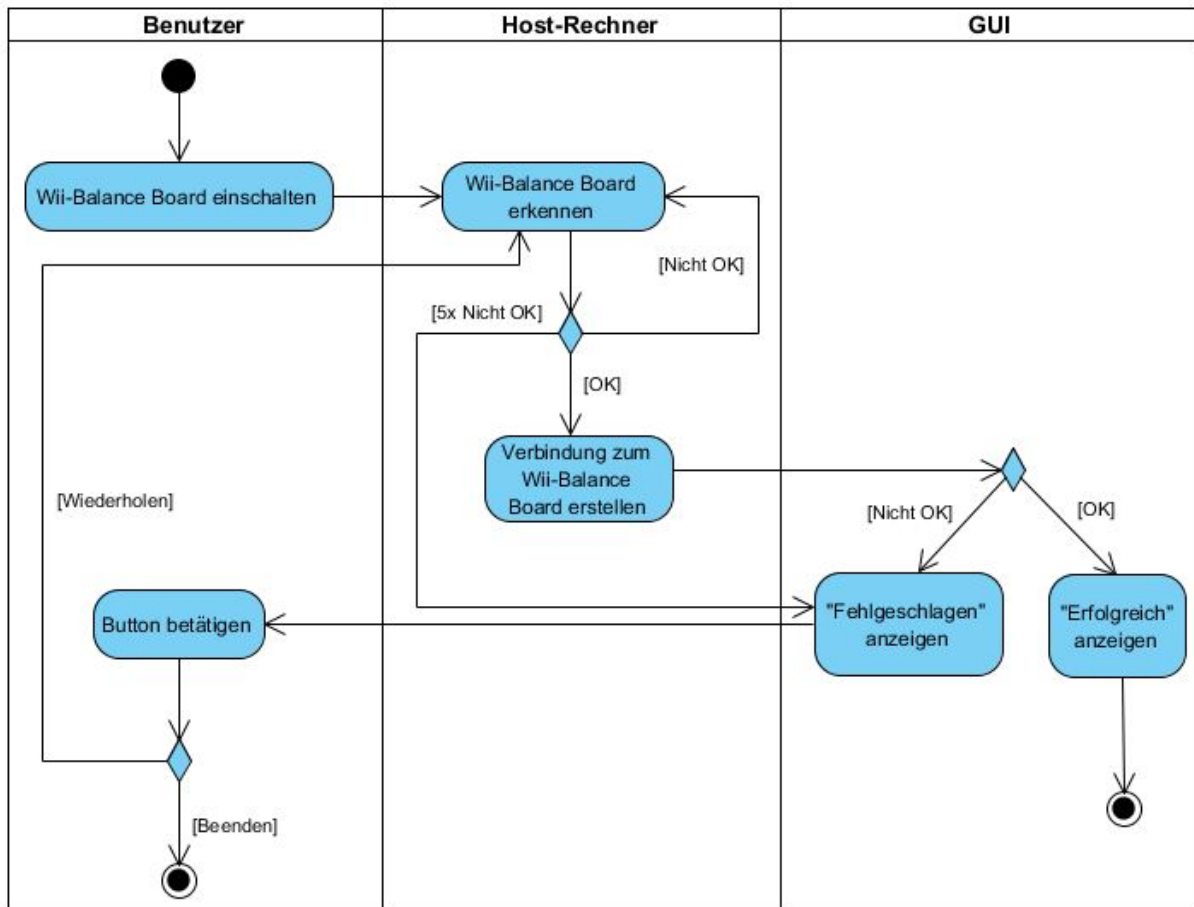


Abbildung 1.1: Aktivitätsdiagramm: Verbindung herstellen

Nach dem Starten des Programmes muss der Benutzer mindestens einen gewünschten Sensor einschalten (in diesem Fall, das Wii-Balance Board), damit dieser vom Hostrechner erkannt werden kann (Abbildung 1.1). Ist es für das Programm nicht möglich, einen Sensor nach fünf Versuchen zu erkennen, erscheint für den Benutzer ein Fenster, welches ihm den Fehlschlag anzeigt.

Ansonsten, bei erfolgreicher Erkennung, kann sich der Benutzer für einen Sensor entscheiden und diesen in einem Auswahlfenster selektieren. Anhand der Auswahl wird automatisch versucht eine Verbindung zu diesem aufzubauen. Sind die Gerätschaften (Hostrechner und Sensor) nicht im Stande eine Verbindung zueinander herzustellen, taucht eine Fehleranzeige auf (Abbildung 1.1). Dieses Fenster beinhaltet zwei Buttons, mit denen der Benutzer wählen kann, ob er das Programm beenden möchte oder erneut vom Programm nach den eingeschalteten Sensoren gesucht werden soll.

Konnte andererseits eine Verbindung aufgebaut werden, wird dem Benutzer ein erfolgreiches Verbinden der Geräte mittels der Benutzeroberfläche mitgeteilt. Daraufhin kann der Benutzer mit dem Betätigen eines “Weiter“-Buttons das Programm fortführen.

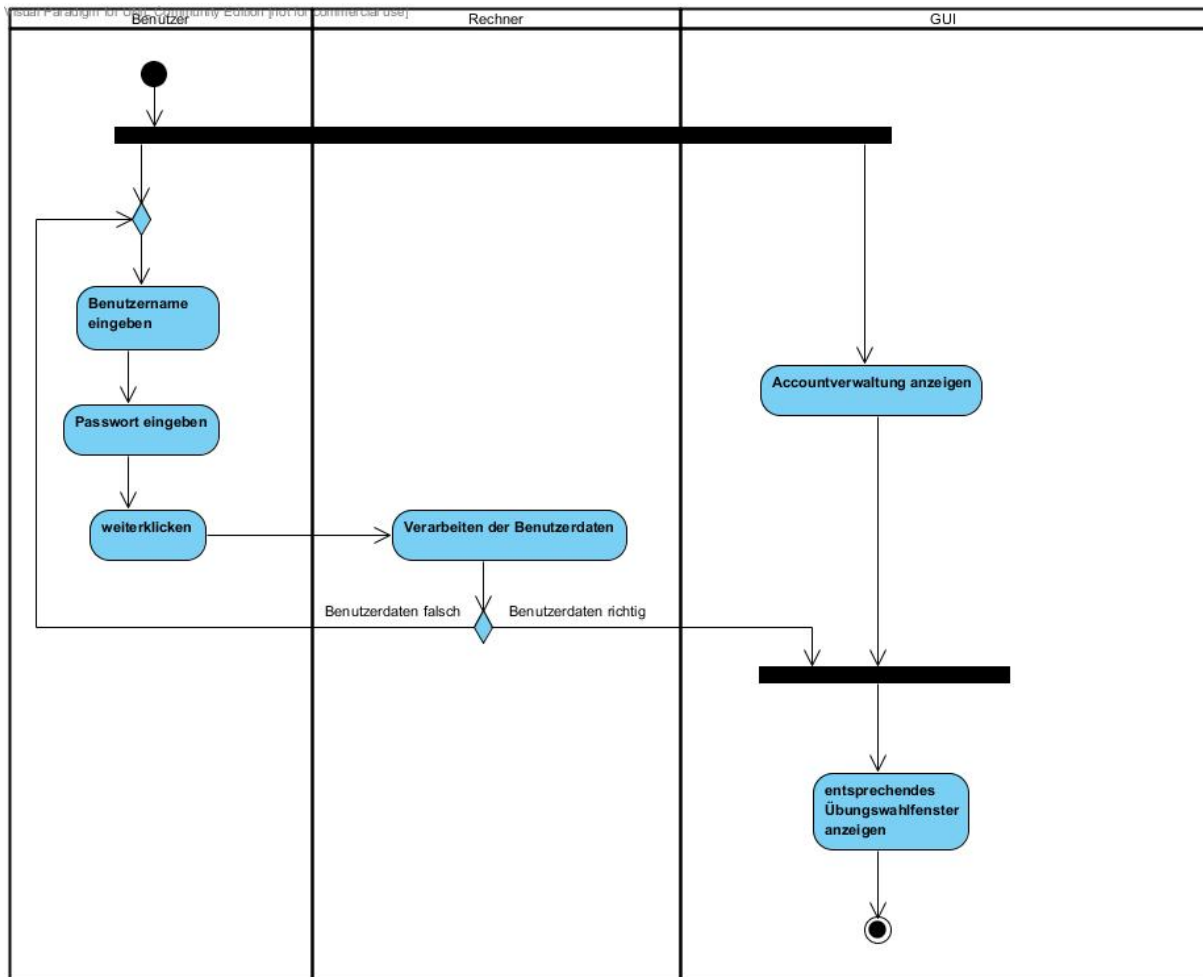


Abbildung 1.2: Aktivitätsdiagramm: Accountverwaltung

Anschließend wird der Anwender aufgefordert, seine Benutzerdaten, inklusive Passwort, einzugeben. Ist die Eingabe falsch, wird der Benutzer erneut aufgefordert, seine Daten einzutragen (Abbildung 1.2).

Beim korrekten Eintippen, gelangt der Anwender zu einem für ihn individuell zugeschnittenen Übungswahlfenster.

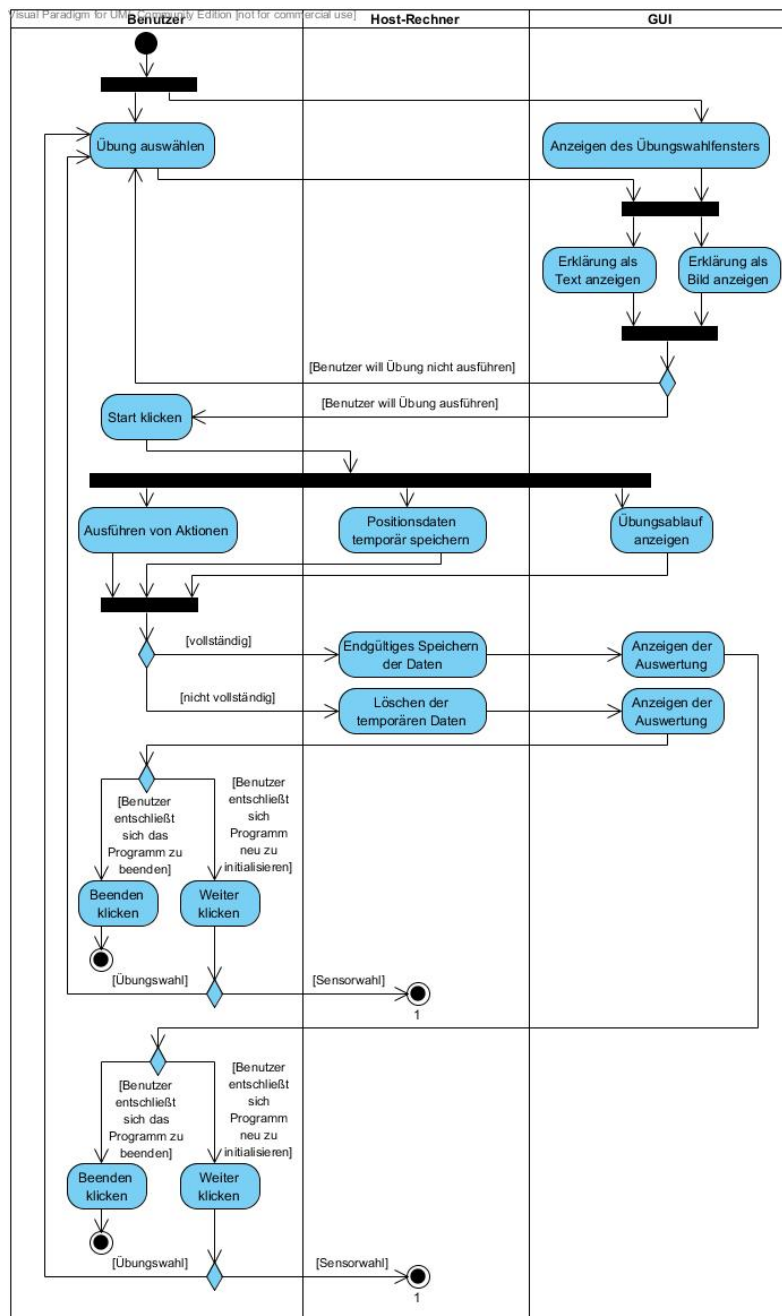


Abbildung 1.3: Aktivitätsdiagramm: Speichern und Übungsablauf

Nun kann der Benutzer aus einer Liste von Übungen eine wählen, welche ihm im Anschluss durch einen Text und ein Bild erklärt wird. Entscheidet sich der Nutzer, die gewählte Übung auszuführen, bestätigt er dies mit einem “Start“-Button. Ist dies nicht der Fall, kann er sich für eine andere Übung entschließen (Abbildung 1.3).

Startet der Anwender die Trainingseinheit und bedient den Sensor (in diesem Falle das Betreten des Wii-Balance Boards), initialisiert er damit eine temporäre Speicherung der Positionsdaten.

Zusätzlich wird ihm gleichzeitig der aktuelle Übungsablauf mit seiner derzeitigen Gleichgewichtsposition angezeigt.

Führt er die Übung vollständig aus, werden die zuvor temporär gespeicherten Daten, nun endgültig auf dem Hostrechner gespeichert und zur weiteren Verarbeitung bereitgestellt.

Andererseits, wenn der Benutzer die Übung zu lange unterbricht, werden die temporären Daten gelöscht.

Jedoch wird in beiden Fällen eine Auswertung mit unterschiedlichem Inhalt (siehe 2.7) aufgeführt.

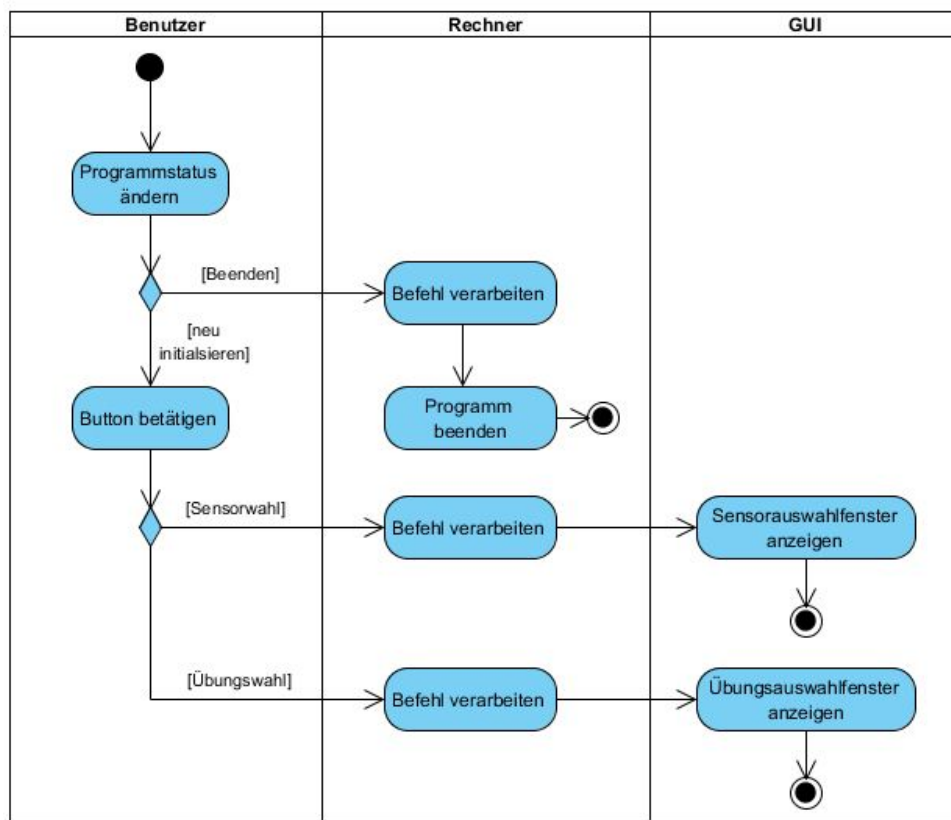


Abbildung 1.4: Aktivitätsdiagramm: Beenden

Infolgedessen wird es dem Anwender durch verschiedene Buttons ermöglicht, sich zu entscheiden, zum Einen das Programm komplett zu beenden oder zum Anderen es neu zu initialisieren. Entweder kann er mit einem Button ins Übungs- oder Sensorwahlfenster (siehe Abbildung 1.4 Endpunkt Nummer 1) zurückgelangen.

Eine ausführliche Beschreibung der einzelnen Funktionalitäten erfolgt im Abschnitt 2.

2 Analyse der Produktfunktionen

Im Folgenden werden die einzelnen Produktfunktionalitäten, sowohl funktionale als auch nicht funktionale, anhand von Sequenzdiagrammen beschrieben. Jedes Diagramm spiegelt dabei eine einzelne Funktion wieder. Jedoch ist es in einigen Fällen sinnvoll, mehrere Funktionalitäten in einem Punkt und damit in einem Diagramm zusammenzufassen.

Der Controller stellt jeweils das Bindeglied zwischen den einzelnen Akteuren, bzw. den beteiligten Paketen, dar.

2.1 Analyse von Funktionalität /F100/ und /F200/: Erkennen eines Sensors; Herstellen einer Verbindung

Die Funktionalität /F100/ beschreibt den Erkennungsvorgang des Wii-Balance Boards zum Endgerät. Daraufhin wird durch die Funktionalität /F200/ der Verbindungsaufbau zwischen diesen erläutert. Diese Funktionalitäten werden im folgenden Diagramm zusammen dargestellt.

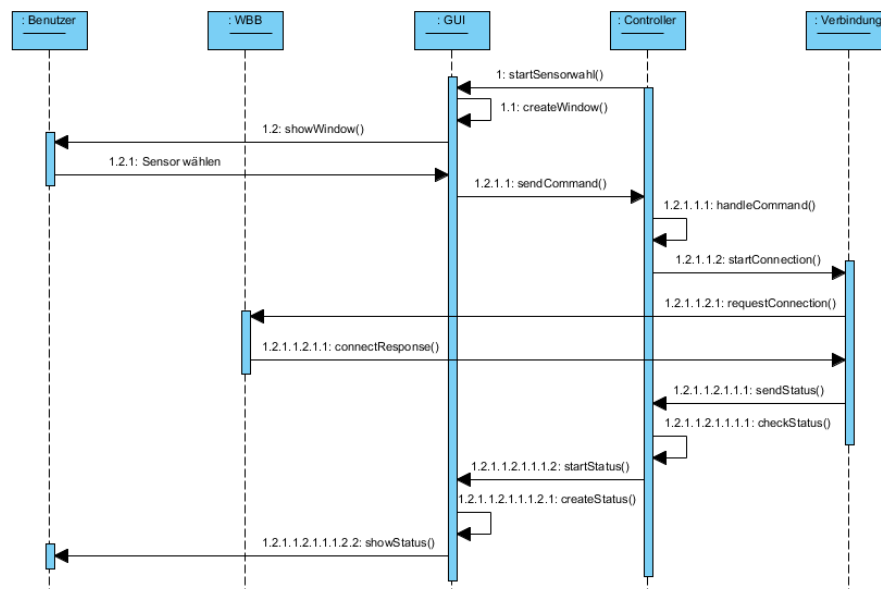


Abbildung 2.1: Sequenzdiagramm: Erkennen eines Sensors, Herstellen einer Verbindung

Zunächst wird in /F100/ (Abbildung 2.1) vom Controller die Sensorwahl auf der GUI initialisiert. Diese gibt dem Benutzer das Auswahlfenster aus. Daraufhin wird ein Bluetoothsignal

vom Wii-Balance Board an das Endgerät gesendet, sofern beide eingeschaltet wurden und sich der Sensor in Verbindungsreichweite befindet. Mit dem gleichen Prinzip können auch weitere Sensoren erkannt bzw. in den Prozess integriert werden (Framework).

Danach wird durch die Funktionalität /F200/ (Abbildung 2.1) eine Verbindung zum ausgewählten Sensor aufgebaut (In diesem Fall zum Wii-Balance Board). Hierfür muss das Endgerät eine Bluetoothschnittstelle für das Wii-Balance Board bereitstellen. Ist dies der Fall, versucht das Verbindungspackage via Bluetooth eine Verbindung mit dem Wii-Balance Board herzustellen. Hierzu muss sich das Board über eine gültige Mac-Adresse beim Endgerät identifizieren (Punkt 1.2.1.1.2.1.1).

Anschließend überprüft (Punkt 1.2.1.1.2.1.1.1) der Controller die Verbindung und sendet eine Aufforderung an die Benutzeroberfläche, die dann dem Benutzer auf dem Bildschirm den Verbindungsstatus anzeigt.

Der Ablauf wird durch die Einbindung des Verbindungspackages, welches die notwendigen Operationen zur Verfügung stellt, vereinfacht.

2.2 Analyse der Funktionalität /F250/: Accountverwaltung

Die Funktion /F250/ beschreibt die Accountverwaltung des Programms. Für diese Funktion werden zwei Sequenzdiagramme verwendet, da der Benutzer korrekte (Abbildung 2.2) oder inkorrekte (Abbildung 2.3) Benutzerdaten eingeben kann und sich das Programm anhand dieser Fälle unterschiedlich verhält.

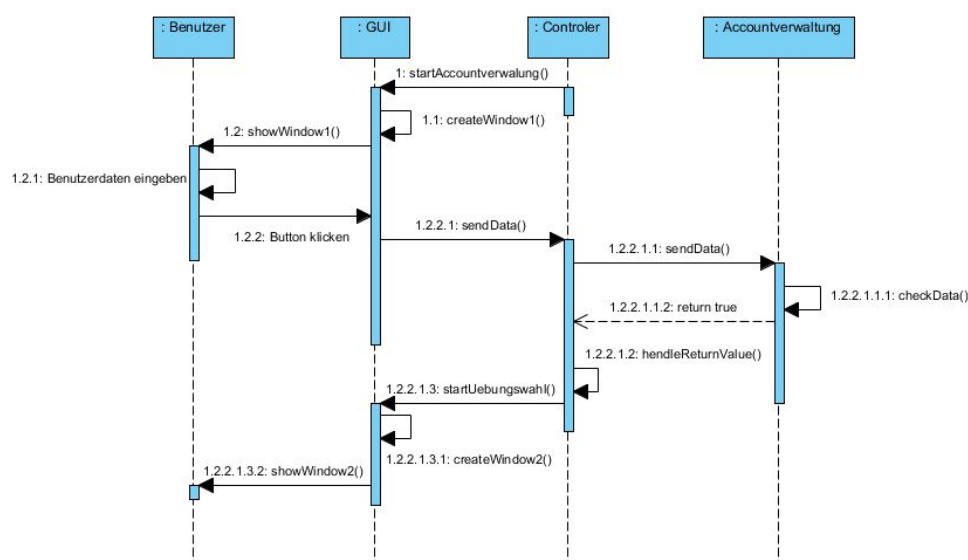


Abbildung 2.2: Sequenzdiagramm: Accountverwaltung für richtige Benutzereingabe

Als erstes wird die Accountverwaltung vom Controller auf der Benutzeroberfläche gestartet. Diese generiert dann das Eingabefenster für die Benutzerdaten und gibt diese dem Benutzer auf dem Bildschirm aus. Wenn dieser seine Daten (Benutzername, Passwort) in die dafür vorgesehenen Textfelder im Fenster eingegeben hat, und den Eingabebutton geklickt hat, werden diese von der Benutzeroberfläche über den Controller an die Accountverwaltung gesendet und auf Richtigkeit durch die Methode `checkData()` (Punkt 1.2.2.1.1.1) überprüft. Die Daten sind korrekt, wenn der Patient vorher von dem behandelnden Arzt als Account im Verwaltungsprogramm registriert wurde und diese Daten richtig eingegeben hat. Der boolesche Wert "true" wird an den Controller übergeben, der daraufhin über die Benutzeroberfläche dem Patienten ein Fenster (Punkt 1.2.2.1.3.2 `showWindow2()`) anzeigen lässt, welches eine individuell für ihn zusammengestellte Übungsliste enthält.

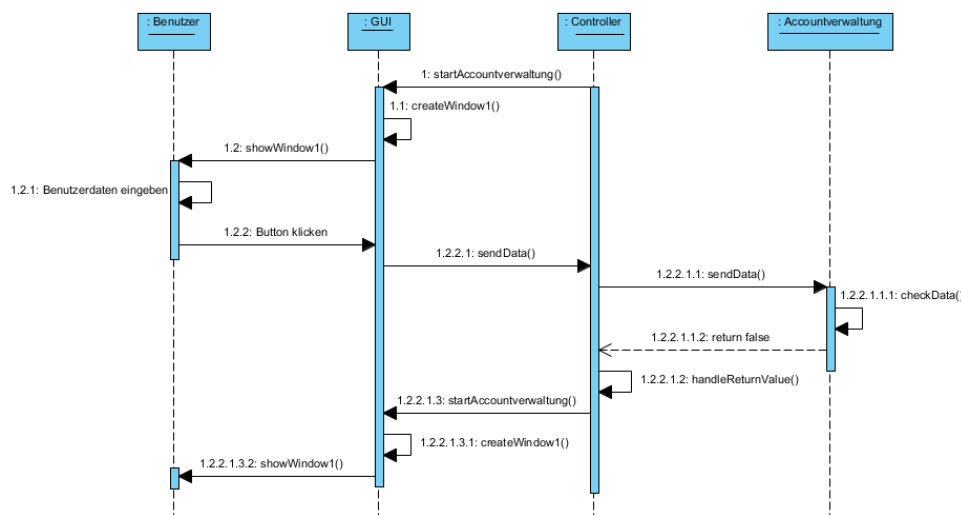


Abbildung 2.3: Sequenzdiagramm: Accountverwaltung für falsche Benutzereingabe

In diesem Szenario (Abbildung 2.3) kann der Ablauf bis zur Übergabe des booleschen Wertes an die Benutzeroberfläche übernommen werden. An dieser Stelle ändert sich der boolesche Wert von "true" zu "false", da der Anwender seine Daten nicht korrekt eingegeben hat. Der Controller erhält nun den booleschen Wert "false", woraufhin dem Benutzer über die Benutzeroberfläche das Accountverwaltungsfenster (Punkt 1.2.2.1.3.2: `showWindow1()`) erneut angezeigt wird. Nun aber mit der Mitteilung, dass er die Daten falsch eingegeben hat. Auf diese Weise wird es ihm ermöglicht, die Eingabe seiner Benutzerdaten zu korrigieren.

2.3 Analyse von Funktionalität /F300/, /F400/und /F500/:

Auswählen der Übung; Erklären der Übung; Starten einer Übung

Die Funktionalität /F300/ beschreibt, wie es dem Benutzer ermöglicht wird, seine gewünschte Übung auszuwählen. Hierbei wird dem Benutzer ein Auswahlmenü angezeigt, bei dem er die Übung über einen Button innerhalb des Fensters anklicken kann.

In der Funktionalität /F400/ wird beschrieben, wie dem Benutzer daraufhin die zugehörige Erklärung einer bestimmten Übung, die er zuvor im Auswahlmenü ausgewählt hat, angezeigt wird. Sobald er nämlich eine von ihm ausgewählte Übung angeklickt hat, öffnet sich ein Erklärungsfenster, in dem die Übungsausführung genau beschrieben wird.

Funktionalität /F500/ beschäftigt sich aufbauend auf den zuvor aufgeführten Funktionalitäten mit dem Starten einer Übung. Nachdem der Benutzer den Startbutton im Erklärungsfenster betätigt hat, kann die ausgewählte Übung begonnen werden, welche im Übungsbildschirm angezeigt wird.

Die zuvor aufgeführten Funktionalitäten werden in Abbildung 2.4 gemeinsam dargestellt. Grund dafür ist, dass diese Funktionalitäten eng miteinander verknüpft sind. Zwischen dem Auswählen einer Übung und dem Erklären der Übung besteht eine direkte Verbindung, da sie aufeinander aufbauend dargestellt werden. Das Starten einer Übung impliziert, dass die Erklärung dem Benutzer angezeigt wurde und innerhalb dieses Erklärungsfensters ein "Start"-Button betätigt wurde.

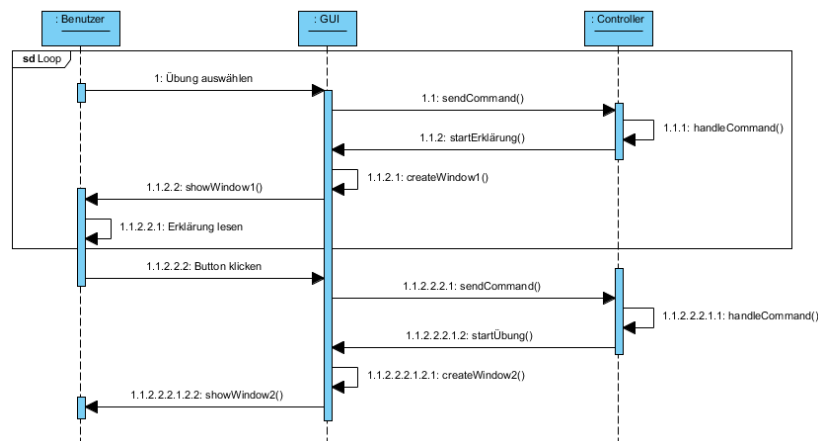


Abbildung 2.4: Sequenzdiagramm: Auswählen, Erklären, Starten einer Übung

Im ersten Schritt muss der Benutzer durch Anklicken eines Buttons seine gewünschte Übung auswählen. Nachdem der Controller, der den Befehl von der Benutzeroberfläche übergeben bekommen hat, den Befehl verarbeitet hat (Punkt 1.1.1), überträgt er den Befehl zum Anzeigen der

Erklärung (Punkt 1.1.2) an die Benutzeroberfläche, welche sie dem Benutzer ausgibt. Nun kann sich der Benutzer die Erklärung durchlesen. Will der Benutzer eine andere Übung ausführen, muss er eine andere Übung anklicken, wodurch der Verarbeitungsvorgang erneut durchgeführt wird (siehe Loop). Diese Schleife wird so lange durchlaufen, bis der Benutzer sich durch das Anklicken des “Start“-Buttons für eine Übung entschieden hat. Dieser Befehl wird wiederum vom Controller bearbeitet (`handleCommand()` Punkt 1.1.2.2.1.1) und die Übung kann, nach Ausgabe des entsprechenden Fensters auf der Benutzeroberfläche, gestartet werden.

2.4 Analyse von Funktionalität /F600/: Anzeigen des Übungsablaufs

Die Funktionalität /F600/ (Abbildung 2.5) beschreibt die graphische Wiedergabe des Übungsablaufs. Der Benutzer kann den aktuellen Trainingsstand, die auszuführenden Teilaufgaben und eine ablaufende Zeitangabe betrachten.

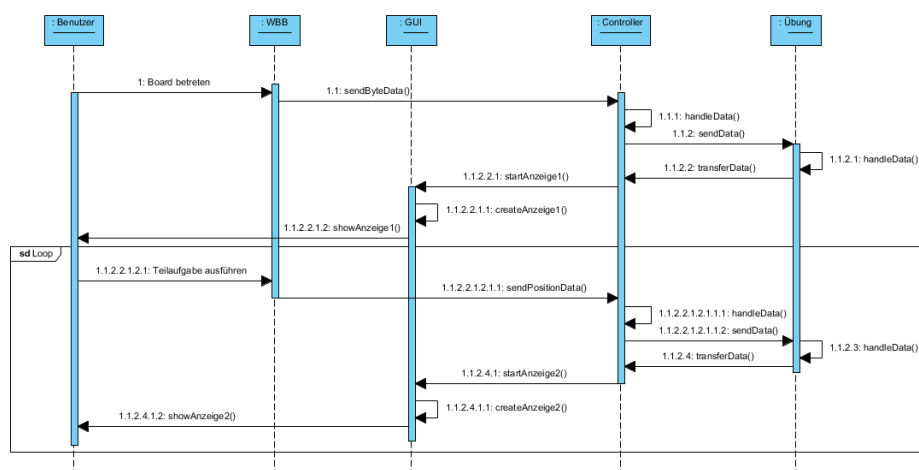


Abbildung 2.5: Sequenzdiagramm: Anzeigen des Übungsablaufes

Durch das Betreten des Wii-Balance Boards wird über den Controller die Übung gestartet. Nachdem die Daten von dem Übungsmodul verarbeitet und an den Controller weitergeleitet wurden, übergibt dieser den Befehl (Punkt 1.1.2.2.1) an die Benutzeroberfläche die erste Teilaufgabe anzuzeigen. Bei dieser erstmaligen Betätigung des Sensors fängt die Zeitangabe an abzulaufen und der Benutzer kann nun die vorgegebene Teilaufgabe ausführen. Indem der Anwender sein Gewicht nach links, rechts, vorne oder hinten verlagert, agiert (Punkt 1.1.2.2.1.2.1) er mit dem Wii-Balance Board. Diese Bewegungsdaten werden vom Controller bearbeitet und an das Übungspackage übergeben. Außerdem können auf diese Weise die Positionsdaten temporär gespeichert werden (siehe /F700/).

Anhand dieser Daten kann von nun an der aktuelle Trainingsstand (Status) abgerufen und dar-

aufhin angezeigt werden. Der gesamte Ablauf der Methoden geschieht innerhalb kürzester Zeit, sodass alles gleichzeitig auf dem Bildschirm erscheint.

2.5 Analyse von Funktionalität /F700/: Speichern von Daten

Die Funktionalität /F700/ (Abbildung 2.6) beschreibt, wie Daten innerhalb des Programms auf der Festplatte gesichert werden sollen. Dabei sollen sowohl die Benutzerdaten, Auswertungsdaten, als auch die vom Wii-Balance Board übermittelten Positionsdaten gespeichert werden und später von der Festplatte abrufbar sein.

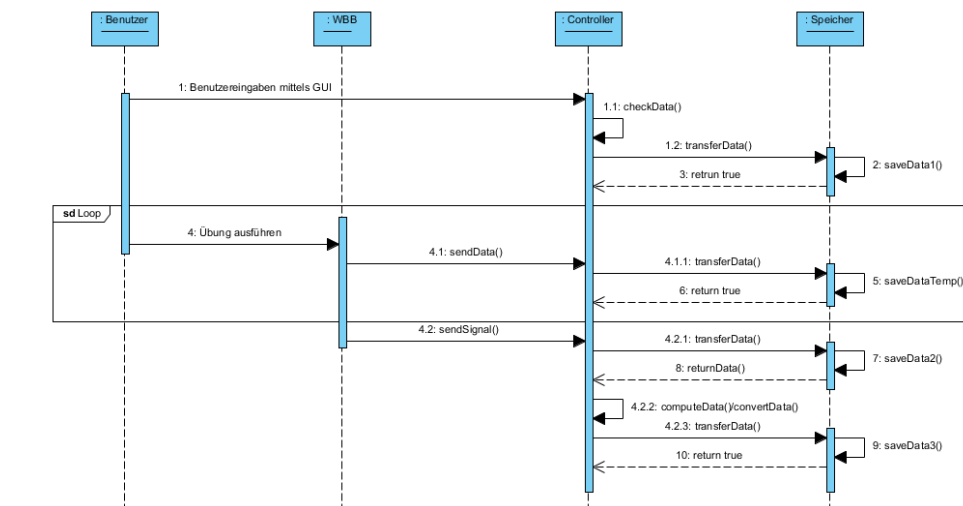


Abbildung 2.6: Sequenzdiagramm: Speichern von Daten

Nachdem der Benutzer seine ihm zugewiesenen Benutzerdaten, inklusive Passwort, in das Accountverwaltungsfenster eingegeben und seine Eingabe mit einem Button bestätigt hat, werden die von ihm angegebenen Daten vom Programm überprüft. Das bedeutet, dass in Schritt 1.1 kontrolliert wird, ob der Benutzername mit dem zugehörigen Passwort in dieser Kombination existiert. Ist dies der Fall, so werden die eingetragenen Daten an den Speicher übertragen und in einem Dokument abgespeichert.

Die Schritte zwischen Accountverwaltung und dem Beginn einer Übung werden an dieser Stelle der Übersicht halber ausgelassen (genaue Beschreibung in den Kapiteln 2.3 und 2.4) und es wird umgehend mit dem Ausführen einer Übung fortgefahren.

Sobald der Benutzer anfängt eine Übung auszuführen, werden konstant Positionsdaten vom Wii-Balance Board an den Controller und von diesem an den Speicher übertragen. Diese Daten werden zunächst in Punkt 5 temporär und lokal gespeichert.

Sobald der Benutzer seine Übung regulär beendet hat, werden keine Positionsdaten mehr an den Rechner übertragen, sondern lediglich ein Bluetoothsignal übermittelt. Zu diesem Zeitpunkt, werden dann die in Punkt 5 temporär gespeicherten Daten auf die Festplatte übertragen und

dort dauerhaft gespeichert. Nach der Speicherung erfolgt, wie in Abschnitt 2.7 noch genauer beschrieben wird, das Umwandeln und Berechnen der Daten, welche dann im letzten Schritt wiederum in ein Dokument gespeichert werden, um diese später leicht abrufen zu können. Wenn der Benutzer die Übung jedoch nicht regulär beendet, werden alle zuvor temporär gespeicherten Daten gelöscht. Dies wird nicht in einem einzelnen Diagramm dargestellt, da sich nur wenig in Abbildung 2.7 ändern würde.

2.6 Analyse von Funktionalität /F800/: Beenden einer Übung

Die Funktionalität /F800/ beschreibt sowohl das reguläre Beenden einer Übung (Abbildung 2.8), als auch das Beenden durch Unterbrechung (Abbildung 2.9) des Trainingsablaufs. Im Folgenden werden diese beiden Prozesse in zwei voneinander unabhängigen Sequenzdiagrammen dargestellt.

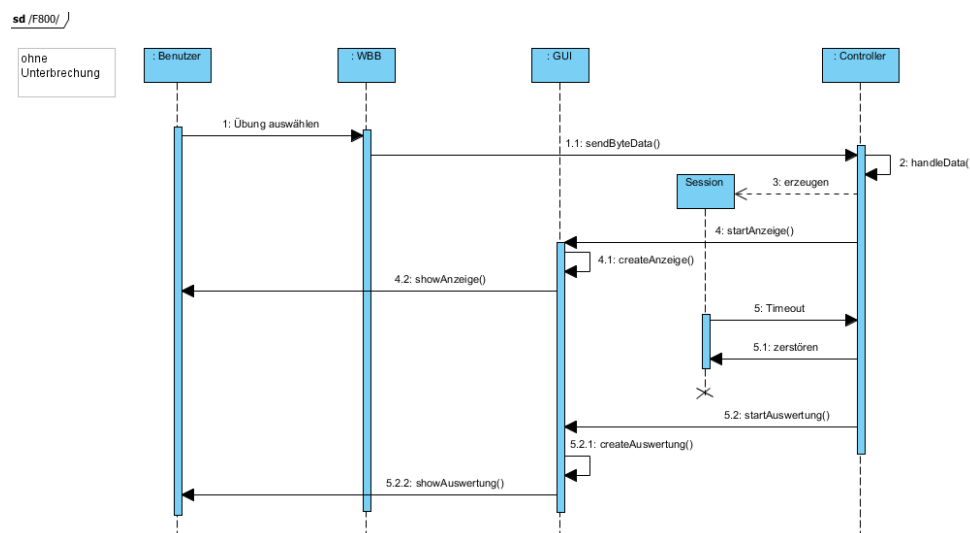


Abbildung 2.7: Sequenzdiagramm: Beenden einer Übung ohne Unterbrechung

Sobald der Benutzer beginnt eine Übung auszuführen, werden vom Wii-Balance Board konstant Positionsdaten an den Controller übertragen. Diese werden in Punkt 2 vom Controller verarbeitet. Das Verarbeiten meint unter anderem das in Abschnitt 2.6 beschriebene Speichern.

So wie der Controller die Positionsdaten erhält, wird synchron eine Anfrage an die Benutzeroberfläche gesendet, um die aktuelle Position auf dem Bildschirm darzustellen. Außerdem wird zeitgleich eine neue Session erzeugt, die die Übungsanzeige und Alles, was während des Durchführens einer Übung ausgeführt wird, repräsentieren soll.

Nachdem die Übung regulär vom Anwender abgeschlossen wurde, also die vorgegebene Zeit für den Trainingsverlauf abgelaufen ist, sendet die Session ein Timeout an den Controller (Punkt 5 in Abbildung 2.8). In diesem Fall wird die Session vom Controller beendet und daraufhin vom

Selbigen die Auswertung auf der Benutzeroberfläche aufrufen, welche diese schließlich dem Benutzer sichtbar macht.

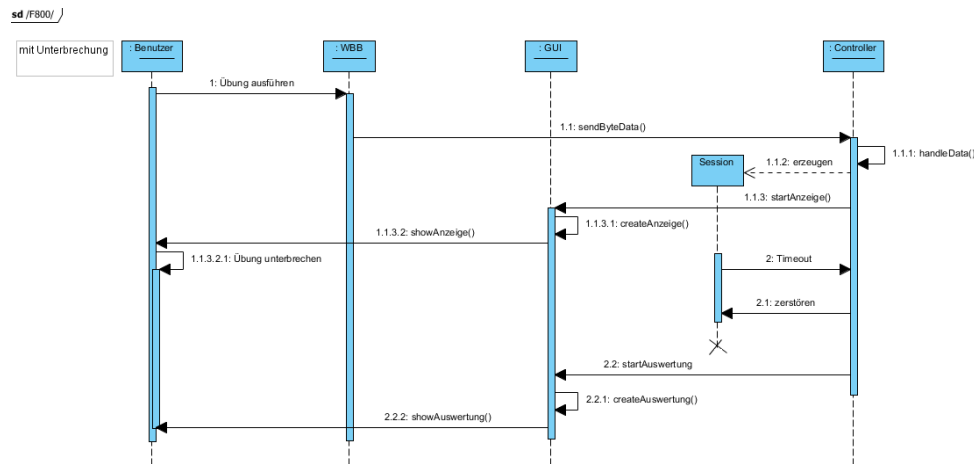


Abbildung 2.8: Sequenzdiagramm: Beenden einer Übung mit Unterbrechung

Das Beenden einer Übung durch Unterbrechung des Trainings verhält sich ähnlich wie das normale Beenden. Auch hierbei werden nach Beginn der Übung konstant Positionsdaten an den Controller gesendet, welche dann von diesem verarbeitet werden (Punkt 1.1.1). Somit wird auch in diesem Falle eine neue Session erzeugt.

Der Unterschied liegt jedoch darin, dass das Timeout (Punkt 2 in Abbildung 2.9) nicht wie oben beschrieben durch den normalen Zeitablauf ausgelöst wird, sondern indirekt durch den Benutzer selbst. Wenn der Benutzer das Wii-Balance Board aus unbekannten Gründen für mindestens zwei Minuten verlässt und somit zwei Minuten lang keine Positionsdaten mehr an den Controller übertragen werden, wird Punkt 2 ausgelöst und die Session zerstört (Punkt 2.1), obwohl die Übung nicht vollständig ausgeführt wurde.

Trotzdem wird auch hier eine Aufforderung an die Benutzeroberfläche gesendet, um dem Benutzer das Auswertungsfenster anzuzeigen. Dieses Fenster unterscheidet sich dann jedoch in seinem Inhalt (siehe Abschnitt 2.7).

2.7 Analyse der Funktionalität /F1000/: Auswertung der Übung

Die Funktion /F1000/ (Abbildung 2.10) beschreibt die Berechnung der Trainingsdaten, die zur Auswertung bereit gestellt werden, solange die Übung vom Patienten vollständig ausgeführt wurde.

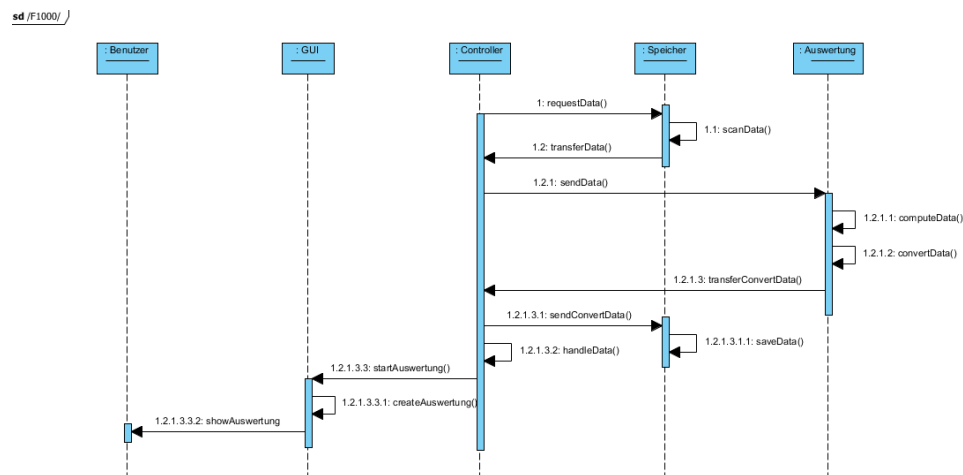


Abbildung 2.9: Sequenzdiagramm: Auswertung der Übung

Der Controller erfragt vom Speicher die endgültig gespeicherten Positionsdaten (Punkt 1), die von der Auswertung zur Berechnung des Mittelwertes benötigt werden. Dieser wird aus den Ergebnissen der einzelnen Teilaufgaben berechnet. Nachdem der Controller die benötigten Daten erhalten hat, gibt er sie an die Auswertung weiter. Diese Daten werden dazu benutzt die Auswertung, welche vom Programm durchgeführt wird, zu erstellen (Punkt 1.2.1.1).

Durch die Methode 1.2.1.2 werden die berechneten Daten in verschiedenen graphischen Darstellungen (Tabelle, Ampelsystem) umgewandelt. In der Tabelle werden die Ergebnisse der einzelnen Teilaufgaben so wie der Mittelwert aufgelistet. Die Ampel stellt lediglich den Mittelwert visuell nach einem bestimmten Bewertungsmaß dar. So kann der Patient seinen Trainingsstand auf Anhieb erkennen. Um diese Graphiken mit Hilfe der Benutzeroberfläche darzustellen, werden die umgewandelten Daten zunächst an den Controller übermittelt. Auf diese Weise können die Daten sowohl zur Sicherung an den Speicher weiter gegeben werden, als auch an die Benutzeroberfläche, um die Auswertung darzustellen.

2.8 Analyse der Funktionalität /F1100/: Programmstatus ändern

Die Funktion /F1100/ (Abbildung 2.11) beschreibt die Möglichkeit des Nutzers den Status des Programmes nach dem Beenden einer Übung zu ändern. Es besteht die Möglichkeit das Programm komplett zu beenden, eine neue Übung oder einen neuen Sensor auszuwählen.

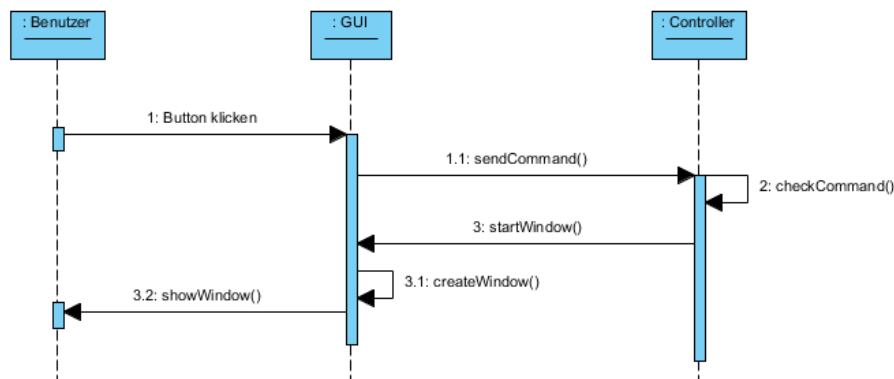


Abbildung 2.10: Sequenzdiagramm: Programmstatus ändern

Durch Auswahl eines Buttons kann der Nutzer entscheiden, ob er das Programm beenden oder neu initialisieren möchte. Entscheidet er sich das Programm zu beenden (Abbildung 1.4), wird das Programm geschlossen.

Wählt er wiederum einen der beiden anderen Button, kann er das Programm auf zwei verschiedene Art und Weisen neu initialisieren. Je nach Auswahl wird nach dem Verarbeiten des Befehls (Punkt 2) durch den Controller ein entsprechendes Fenster bei der Benutzeroberfläche angefordert (Punkt 3). Dieses gibt die Benutzeroberfläche schließlich dem Benutzer aus (Punkt 3.2).

Bei dem Button "Weiter zur Übungsauswahl" wird das aktuelle Fenster geschlossen und erneut das Übungswahlfenster angezeigt. Dabei werden die Sensorwahl und die Benutzereingabe übersprungen und man kann mit demselben Sensor eine neue Übung auswählen.

Bei Auswahl des Buttons "Weiter zur Sensorauswahl" werden wieder neue Sensoren gesucht und im Sensorwahlfenster angezeigt. Das bedeutet, dass der gesamte Programmablauf erneut beginnt.

2.9 Analyse der Funktionalität /F1200/: Framework

Die Funktionalität /F1200/ beschreibt, dass unser Projekt als Framework implementiert wird und dadurch generisch fungiert. Ziel ist es, das Framework so zu gestalten, dass es durch einen Informatiker ohne großen Aufwand erweiterbar ist.

Die Haupteigenschaften eines Frameworks sind, dass ein strikter Leitfaden in der Produktimplementierung vorhanden ist. Das heißt, das komplette Framework besteht aus mehreren Klassen, die in einer speziellen Hierarchie angeordnet sind und zusammenarbeiten wodurch es flexible, erweiterbare und wiederverwendbare Entwürfe bereitstellt. Dabei ist auf eine Plattformunabhängigkeit zu achten.

Anhand von Kapitel 3 und den Abschnitten aus Kapitel 2, die sich mit den nichtfunktionalen Anforderungen beschäftigen, wird auf die Implementierung des Frameworks für dieses Projekt näher eingegangen.

2.10 Analyse von Anforderung /Q30/: Berechnungszeit

Diese nichtfunktionale Anforderung gibt an, dass die Berechnung der Mittelwerte und die Umwandlung in eine Tabelle und ein Ampelsystem, die für die Funktion /F1000/ (Auswerten der Übung) benötigt wird, nicht länger als 2 Minuten dauern darf. Nach der Übung soll dem Benutzer nämlich eine Bewertung seines Trainingserfolges auf der Benutzeroberfläche angezeigt werden.

2.11 Analyse von Anforderung /Q50/: Plattformunabhängigkeit


Die Software wird in Java geschrieben und es werden nur Funktionen genutzt, die auf dem Betriebssystem (Windows, UNIX) zur Verfügung stehen. Die Benutzeroberfläche wird in Swing implementiert, um auf jedem Betriebssystem dasselbe Design zu besitzen.

2.12 Analyse von Anforderung /Q60/: Benutzerfreundlichkeit

Da es sich bei den Benutzern des Produkts primär um ältere Menschen handelt, soll das Produkt mit wenigen EDV-Kenntnissen einfach und verständlich zu bedienen sein. Der Benutzer wird anhand mehrerer Fenster linear durch das Programm geführt, da nur die unbedingt erforderlichen Informationen und Buttons pro Fenster angezeigt werden.

2.13 Analyse von Anforderung /Q70/ und /Q100/: Code-Conventions

Um eine leichte Weiterentwicklung/Wartbarkeit und eine gute Strukturierung des Quellcodes zu gewährleisten, wird auf die Einhaltung der Code-Conventions geachtet. Dies beinhaltet folgende Punkte:

- - Code sofort vollständig kommentieren (Parameter einer Methode)
- - Aussagekräftige Namen (Methoden, Felder, Klassen) 
- - Separation of concerns
- - Einheitliche Klammerung
- - Nach if ... else grundsätzlich geschwungene Klammern (für bessere Lese- und Nachbearbeitungsmöglichkeit)
- - Zentrale Sammlung von Konstanten, Hilfsmethoden
- - Anwendung von Design Pattern
- - Pakete, Methoden, Attributnamen -> kleiner Anfangsbuchstabe
- - Klassen, Interfaces -> großer Anfangsbuchstabe
- - Konstanten komplett groß, mit Unterstrich
- - Ordnung innerhalb der Klassen:
 1. Attribute
 2. Konstruktoren
 3. Akzessoren (get/set)
 4. Methoden

2.14 Analyse von Anforderung /Q80/ und /Q90/: Dokumentation

Die Dokumentation wird ausführlich und klar strukturiert erarbeitet. Sämtliche Funktionen der Software werden ausführlich und einfach in Englisch erklärt, um eine verständliche Software zu gewährleisten, die die Weiterentwicklung durch andere Informatiker möglichst einfach hält. Die Kommentare im Quellcode enthalten alle wichtigen Informationen (wie in 2.13 aufgeführt).

2.15 Analyse von Anforderung /Q120/: Fehlertoleranz bezüglich Benutzereingabe

Das Programm wird bezüglich der Fehleingaben des Benutzers ein sehr hohes Maß an Fehlertoleranz besitzen. Fehleingaben des Benutzers werden abgefangen und es wird durch entsprechende Ausgaben auf dem Bildschirm auf die Fehler hingewiesen. Daraufhin besteht die Möglichkeit die fehlerhafte Eingabe zu korrigieren.

3 Resultierende Softwarearchitektur

3.1 Komponentenspezifikation

In Abbildung 3.1 werden die einzelnen Komponenten mit ihren Schnittstellen dargestellt. Die unterschiedlichen Komponenten besitzen jeweils mehrere Bestandteile, die sich aus den einzelnen Produktfunktionen ergeben. Aus diesem Grund unterscheidet sich die Anzahl der Komponenten mit der Anzahl der Produktfunktionen aus Kapitel 2.

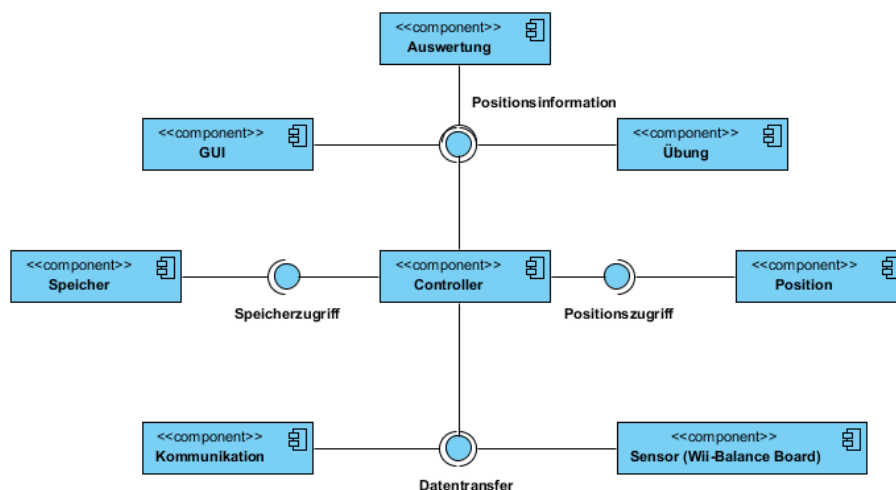


Abbildung 3.1: Komponentendiagramm: Komponentenspezifikation

Die Komponente Controller stellt für alle anderen Komponenten Schnittstellen bereit und dient somit als Vermittler zwischen diesen.

Die Komponenten GUI, Auswertung und Übung greifen indirekt über die Schnittstelle Positionsinformation via Controller auf die aktuelle Gewichtsverteilung des Benutzers zu. Aufgrund der dadurch erhaltenen Daten wird über die Verbindung zwischen GUI und Übung dem Benutzer seine Gleichgewichtsverlagerung visuell mitgeteilt. Außerdem kann durch die empfangenen Daten die Berechnung für die Auswertung stattfinden, die wiederum über eine Verbindung zwischen GUI und Auswertung dem Benutzer dargestellt wird.

Über die Schnittstellen Speicherzugriff und Positionszugriff werden die Positionsdaten wiederum über den Controller beim Übungsablauf direkt an den Arbeitsspeicher übertragen. Danach werden über dieselben die Auswertungsdaten mit den zuvor temporär gespeicherten Daten an die Festplatte übergeben.

Eine genauere Beschreibung der einzelnen Schnittstellen erfolgt in Abschnitt 3.2 und die Beschreibung der verschiedenen Komponenten in Unterkapitel 3.3.

3.2 Schnittstellenspezifikation

Schnittstelle	Operation	Beschreibung
/S10/: Speicherzugriff	storeUserData()	Speichern der Benutzerdaten
	storeTempData()	Temporäres Speichern der Positionsdaten
	storePositionData()	Endgültiges Speichern der Positionsdaten
	loadUserData()	Laden der Benutzerdaten
	loadTempData()	Laden der temporär gespeicherten Positionsdaten
	loadPositionData()	Laden der endgültig gespeicherten Positionsdaten
	storeScoreData()	Speichern der Bewertungsdaten
	loadScoreData()	Laden der Bewertungsdaten

Schnittstelle	Operation	Beschreibung
/S20/: Positionsinformation	getAnalysisData()	Daten der Auswertung bereitstellen
	getTrainData()	Daten des Trainingablaufes bereitstellen
	changeListener()	Prüfen, ob Änderungen mit getPosition() oder getTrainData() auszuwerten sind

Schnittstelle	Operation	Beschreibung
/S30/: Datentransfer	sendPositionData()	Senden der Positionsdaten vom Wii-Balance Board zum Rechner
	receivePositionData()	Aufnahme der Positionsdaten vom Wii-Balance Board durch den Rechner

Schnittstelle	Operation	Beschreibung
/S40/: Positionszugriff	getPosition()	Aktuelle Lage des Schwerpunktes des Benutzers auf dem Wii-Balance Board bereitstellen

3.3 Protokolle für die Benutzung der Komponenten

3.3.1 Controller

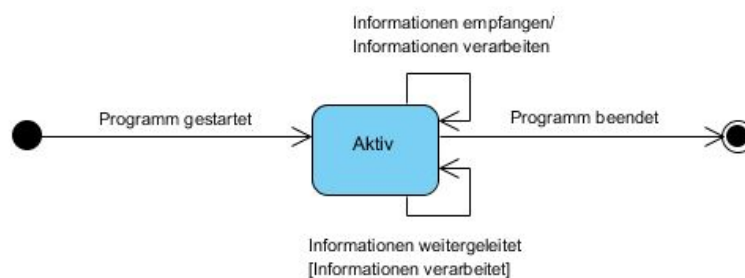


Abbildung 3.2: State Chart: Controller

Die Controllerkomponente (Abbildung 3.2) dient im Wesentlichen als Verbindungselement und somit als Vermittler zwischen den einzelnen Komponenten. Dessen Hauptaufgabe ist es, die von den anderen Komponenten erhaltenen Informationen, über die dafür vorgesehenen Schnittstellen, zum richtigen Zeitpunkt, für die entsprechenden Komponenten zur Verfügung zu stellen. Da der Controller auf diese Anwendung spezifiziert ist, ist eine Wiederverwendung dieser Komponente nicht möglich.

und so vielleicht der Gesundheitszustand des Benutzers verschlechtert wird. Außerdem können das Übungswahlfenster und die Auswertungsanzeige übernommen werden.

Der einzige Teil der Komponente der nicht wiederverwendbar ist, ist die Übungsanzeige selbst, da sie für jede Übung, insbesondere für jeden anderen Sensor spezifisch gestaltet werden muss.

3.3.3 Kommunikation

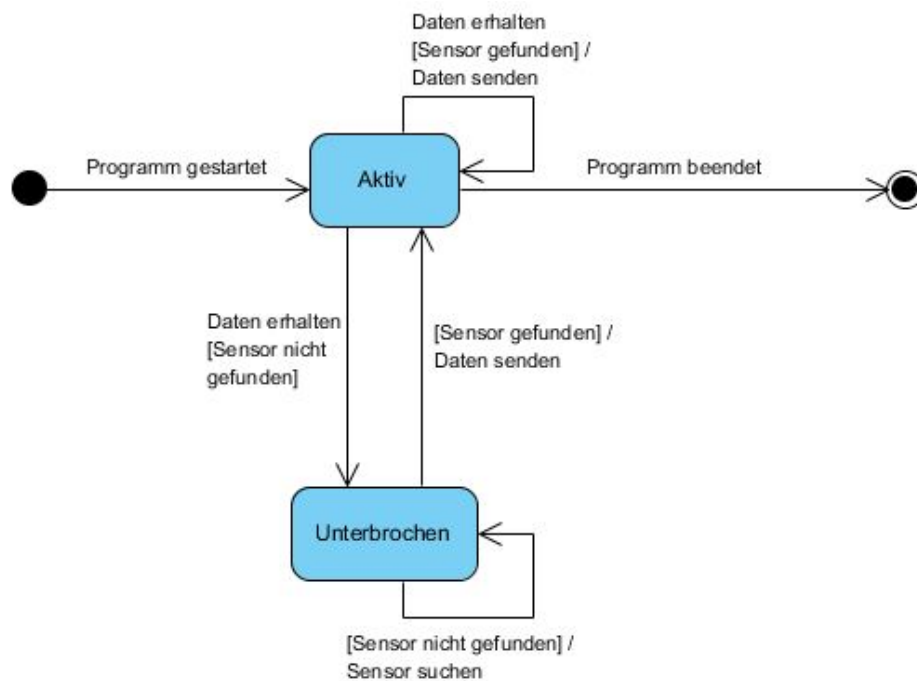


Abbildung 3.4: State Chart: Kommunikation

Die Kommunikationskomponente (Abbildung 3.4) bezieht sich unter Anderem auf die Datenübertragung vom verbundenen Sensor, im konkreten Fall dem Wii-Balance Board, zum Hostrechner, auf dem der Controller diese Daten verwaltet. Die Datenübertragung findet hierbei als 24 Byte-Daten statt.

Außerdem bedeutet Kommunikation innerhalb unseres Systems eine Signalübertragung zwischen Hostrechner und Wii-Balance Board via Bluetooth.

3.3.4 Position

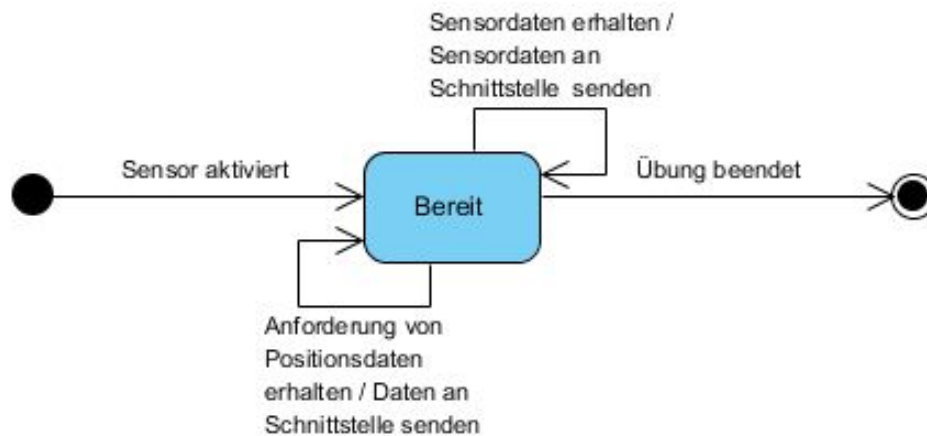


Abbildung 3.5: State Chart: Position

Die Positionskomponente (Abbildung 3.5) verarbeitet die vom Wii-Balance Board, über den Controller, übermittelten 24-Byte Daten. Auf diese Weise werden die Positionsdaten für die übrigen Komponenten verwertbar gemacht.

Da die Verarbeitung von Daten bei jedem Eingabegerät individuell ist und sich in diesem Falle auf das Wii-Balance Board bezieht, ist eine Wiederverwendung dieser Komponente nicht möglich.

3.3.5 Wii-Balance Board

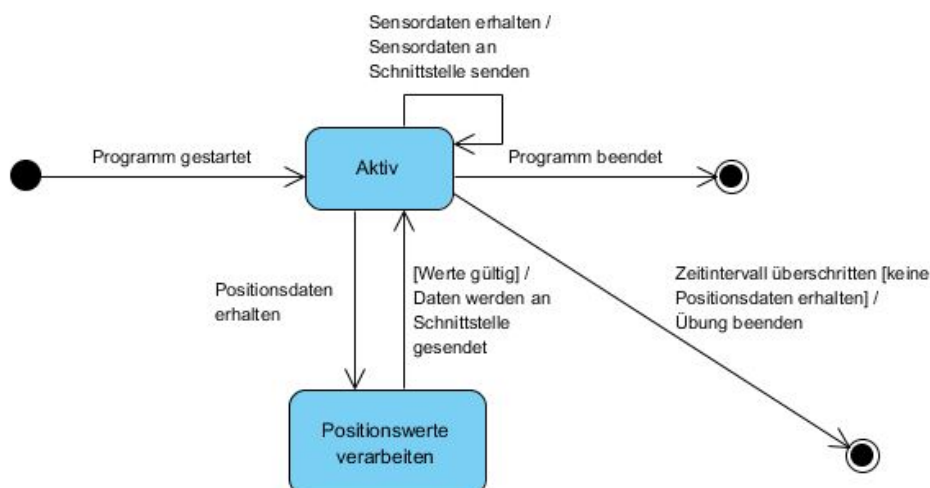


Abbildung 3.6: State Chart: Wii-Balance Board

Die Wii-Balance Board-Komponente (Abbildung 3.6) dient ausschließlich dem Aufnehmen von Daten und dem Weiterleiten über die dafür vorgesehene Schnittstelle via Bluetooth.

Da diese Komponente speziell für Anwendungen vorgesehen ist, die sich mit der Steuerung durch Gewichtsverteilung beschränkt, ist sie nicht für davon abweichende wiederverwendbar.

3.3.6 Auswertung

Die Auswertungskomponente (Abbildung 3.7) erhält mit Hilfe der Schnittstelle Positionsinformation die notwendigen Daten aus dem Speicher, um die Berechnung für die Bewertung auszuführen.

Zusätzlich können das Ampelsystem und die Tabelle (Punktesystem), die in dieser Komponente umgewandelt werden, über die eben genannte Schnittstelle an die GUI weitergeleitet werden.

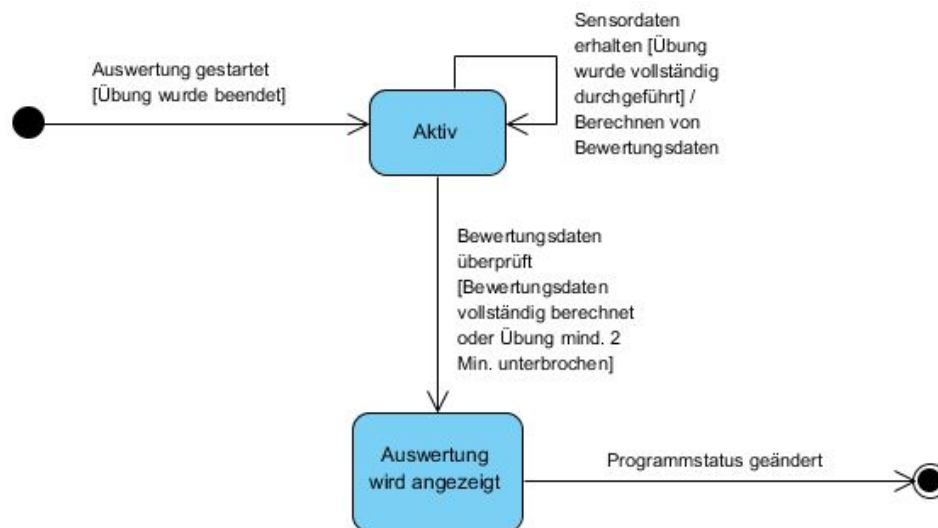


Abbildung 3.7: State Chart: Auswertung

Diese Komponente ist wiederverwendbar, gleichgültig, welchen Sensor der Benutzer beim Starten auswählt. Denn auf diese Weise erhält der Anwender immer dieselbe Auswertung als Punkte- und Ampelsystem dargestellt, um eine möglichst gute Vergleichbarkeit zwischen den verschiedenen Trainingseinheiten zu gewährleisten. Dies wird durch die Generizität des Programmfragments (Implementierung des Frameworks) sichergestellt.

3.3.7 Speicher

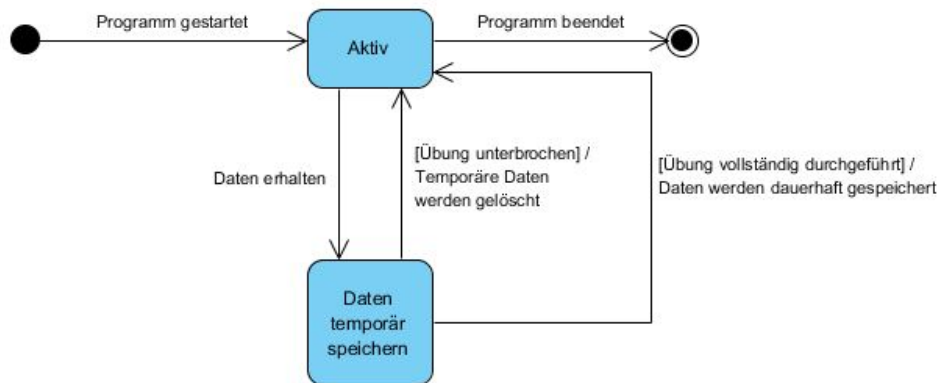


Abbildung 3.8: State Chart: Speicher

Die Speicherkomponente (Abbildung 3.8) ermöglicht es, über die Schnittstelle Speicherzugriff alle Trainingsdaten, wie beispielsweise die Positionsdaten oder Benutzerdaten, jedes Patienten lokal auf der Festplatte des Hostrechner zu sichern. Voraussetzung hierfür ist, dass der Patient seine Benutzerdaten eingegeben und die Übung vollständig durchgeführt hat. Die gespeicherten Daten sind für den Benutzer jederzeit verfügbar.

Die Speicherkomponente kann bei abweichenden Implementierungen im Rahmen des Frameworks ohne Probleme wiederverwendet werden.

3.3.8 Übung

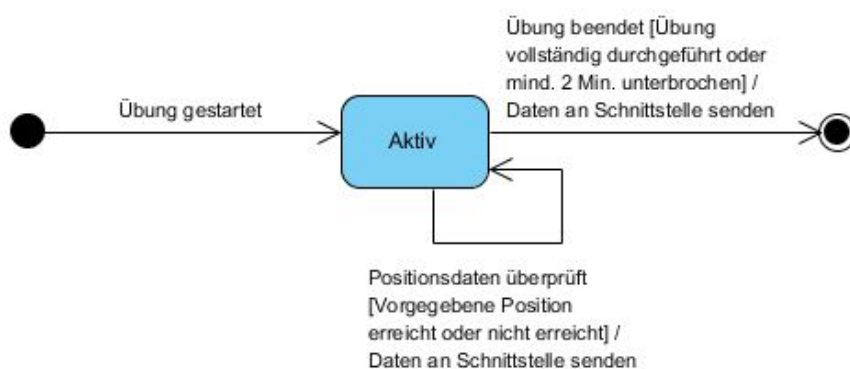


Abbildung 3.9: State Chart: Übung

Die Übungskomponente (Abbildung 3.9) überwacht, während des Übungsablaufs, direkt, mit Hilfe der Schnittstelle Positionsinformation, die Positionen des Patienten, die wiederum über die Schnittstelle Positionszugriff via Controller die entsprechenden Daten erhält. Es wird überprüft,

in wie weit der Patient die vorgegeben Punkte mit Hilfe seiner Gewichtsverteilung erreicht hat. Die Wiederverwendbarkeit ist bei dieser Komponente nicht gegeben, da sie für jede Übung neu angepasst werden muss.

4 Verteilungsentwurf

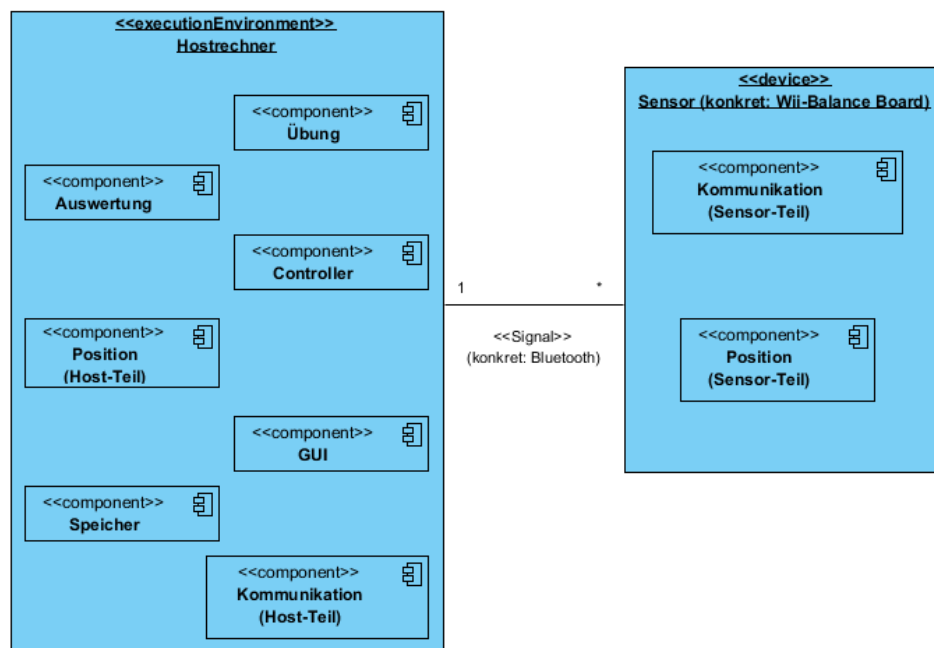


Abbildung 4.1: Verteilungsentwurf

Bei dem zu implementierenden System handelt es sich um eine für Patienten im ambulanten Rehabereich zugeschnittene Anwendung. Das System kommuniziert, wie in Abbildung 4.1 zu sehen, mit einem Sensor, in diesem speziellen Fall dem Wii-Balance Board. Die Kommunikation findet per Bluetooth statt und übermittelt die Positionsdaten des Patienten auf dem Wii-Balance Board und somit die Gewichtsverteilung(den Schwerpunkt) an den Hostrechner.

Die von dem Wii-Balance Board zu bewerkstelligen Komponenten beschränken sich auf den Kommunikations- und den Positionsteil des Boards.

Alle rechenintensiven Bereiche sowie die Speicherung der Daten, inklusive deren Ausgabe, werden auf dem Hostrechner implementiert.

Diese Spezifikation bezieht sich auf den konkreten Fall des Wii-Balance Boards. Bei Benutzung eines anderen Sensors, muss sie eventuell angepasst werden.