TECHNISCHE DOKUMENTATION

Anhang D zur

Dissertation „Making Strategic Decisions under Time Pressure“

Technische Realisierung des Online-Experiment „Entscheidungsverhalten unter Zeitdruck“

– ein Umsetzungskonzept auf der Basis von MouselabWeb

Autor: Dipl.-Wi.-Ing. Marco Kremer

Inhalt

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Abkürzung	**Erläuterung**
DB	Datenbank
DBMS	Datenbankmanagementsystem

1. Einleitung

In der vorliegenden Arbeit wird die Entwicklung eines wissenschaftlichen Online-Experiments auf Basis der Software *MouselabWeb* beschrieben. Die *open source*-Software wurde für experimentelle Anwendungen konzipiert, die das Verhalten von Menschen in Entscheidungssituationen Computermaus-gestützt untersuchen. Es unterstützt damit das Rational des Experiments und wurde deshalb im Vorhinein ausgewählt. Die notwendigen Modifikationen der Software sollen unter Verwendung von Methoden des Software-Engineerings bestimmt werden.

Das in dieser Arbeit betrachtete Online-Experiment ist inhaltlich bereits konzipiert und designt. Im Vordergrund steht deshalb der Entwurf der technischen Realisierung mit MouselabWeb. Diese soll konsequent den Erfordernissen des Experiment-Designs folgen. Dabei werden Methoden des Software-Engineerings zur Beschreibung der Softwarespezifikation und Erstellung des Entwurfs zur Anwendung gebracht. Die softwaretechnische Implementierung, Evaluation und Evolution dieses Konzepts sind nicht Teil dieser Dokumentation. Sie finden sich in den weiteren Anhängen zum Experiment

Im ersten Schritt sollen die Anforderungen an das System herausarbeiten und Funktionalitäten sowie Restriktionen spezifiziert werden. Diese werden dem Funktionalitätsumfang von MouselabWeb gegenüber gestellt, um die fehlenden Spezifikationen herauszuarbeiten. Auf dieser Grundlage werden das Gesamtsystem und seine Komponenten modelliert, bevor sein Aufbau konstruiert wird. Anhand dieser Vorgaben kann schließlich der Ablauf des Experiments auf der Ebene des technischen Systems modelliert werden. Im Ergebnis besteht ein Entwurf zur technischen Umsetzung, bei dem der funktionale Anteil von MouselabWeb bestimmt ist und schließlich den Modifikationsaufwand an der Software abgeschätzt werden kann.

Die Struktur der Dokumentation folgt weitestgehend dem hier skizzierten Methodenkanon. In Kapitel 2 werden zunächst aus dem Experimentdesign die Anforderungen an die technische Umsetzung abgeleitet, welche dem Funktionsumfang von MouselabWeb gegenüber gestellt wird. Den Kern der Arbeit bildet Kapitel 3, in welchem ein Konzept für die technische Realisierung entlang des angesprochenen Methodenkanons entworfen wird. Das abschließende Kapitel 4 reflektiert die Ergebnisse des entworfenen Konzepts anhand der vorher identifizierten Anforderungen an das Experiment und MouselabWeb.

2. Systemanalyse

Dieses Kapitel ist der Softwarespezifikation gewidmet. Dabei werden zunächst die technischen Anforderungen an das System ermittelt. Diese werden anschließend dem Leistungsspektrum der zu verwendenden Software MouselabWeb gegenüber gestellt. Auf dieser Grundlage werden schließlich die fehlenden Funktionalitäten identifiziert.

2.1. Ableitung der technischen Anforderungen

In diesem Abschnitt werden aus dem gegebenen Experimentkonzept die Anforderungen an die technische Umsetzung abgeleitet und das System abgegrenzt. Das Experimentkonzept vermittelt dabei, welche Daten wo in welcher Form und zu welchem Zeitpunkt bereitgestellt bzw. erhoben werden sollen. Diese Anforderungen, welche mit einer fettgedruckten, fortlaufenden, römischen Zahl in Klammern gekennzeichnet sind, bilden schließlich die Ausgangssituation für den Vergleich der Anforderungen des Experiments mit der Leistungsfähigkeit der Software MouselabWeb. Der daraus resultierende Anpassungsbedarf beeinflusst die Entwicklung eines Konzepts zur technischen Realisierung wesentlich.

Im Bereich des Entscheidungsverhaltens ist eine zentrale Zielstellung, den Entscheidungsprozess nachvollziehbar abzubilden. Dies kann in Abhängigkeit vom konkreten Entscheidungsproblem über verschiedene Techniken, sogenannte *process tracing methods*, gelingen.[1] Der in dieser Arbeit betrachtete Problemtyp entstammt der Spieltheorie und wird in der Literatur als *nichtkooperatives Zwei-Personen-Spiel in Normalform* bezeichnet.[2] Bei diesem Entscheidungsproblem[3] haben zwei Personen aus einer vorgegebenen Anzahl an Alternativen (sogenannte *Strategien*) die für sie Nutzenmaximierende auszuwählen. Dabei hängt das individuelle Ergebnis (häufig als *Auszahlung* bezeichnet) von der Strategiewahl beider Spieler ab.

In Abbildung 1 ist beispielhaft ein Spiel in Normalform mit zwei Alternativen pro Spieler zu sehen, wie es auch im Experiment für den Probanden dargestellt werden soll **(I)**.[4] Im Experiment übernimmt ein Proband die Rolle eines Spielers **(II)** in einer definierten Anzahl an nacheinander folgenden Spielen (16 Spiele unterteilt in vier gleichgroße Blöcke, jedes Spiel unterbrochen von

[1] Eine kritische Einführung in diese Techniken liefert zum Beispiel Kühberger und Schulte-Mecklenbeck 2011.
[2] Eine nähere Besprechung spieltheoretischer Problemtypen findet sich zum Beispiel bei Gintis 2009.
[3] Entscheidungsprobleme werden in der Spieltheorie als *Spiel* bezeichnet.
[4] Die Darstellung der Aufgabenstellung umfasst implizit auch die Bereitstellung aller sonstiger notwendiger Informationen und Funktionalitäten während des Experiments für den Probanden durch den Experimentator.

einer Pause **(III)**). Jedes Spiel besitzt eine definierte Maximalbearbeitungszeit. **(IV)**. Der Spieler kann ein Spiel aber auch selbstständig vorher beenden **(V)**.

Abbildung 1 Auszahlungsmatrix eines 2x2-Spiels in Normalform.

		(Spalten-)Spieler S	
		Strategie S1	Strategie S2
(Zeilen-) Spieler Z	Strategie Z1	72 ; 93	31 ; 46
	Strategie Z2	84 ; 52	55 ; 79

Quelle: Eigene Darstellung.

Der Rückschluss auf den kognitiven Entscheidungsprozess, der dem Verhalten zugrunde liegt, soll ermöglicht werden über den Nachvollzug der Informationsakquise und -verarbeitung eines Probanden. Dazu ist es notwendig, sein Verhalten bei der Problembearbeitung so genau wie möglich zu erfassen. Dies umfasst Datenvektoren, die Zeitpunkt, Ort und Art der beobachteten Handlung beschreiben **(VI)**.

Eine etablierte *process tracing* -Methode ist *mouse tracking* (Kühlberger et al. 2011). Diese Methode basiert auf einer Technik, welche unter Computer-Nutzern weit verbreitet ist: die Computermaus. Sie bietet sich u.a. deshalb für die Entwicklung eines Online-Experiments an und wird ausgewählt **(VII)**. Damit ist das Hardware-Setup für die Probanden bereits determiniert: ein Probanden-eigener PC inklusive Bildschirm zur Abbildung der Problemstellung und eine Computermaus **(VIII)**.

Die weiteren Anforderungen ergeben sich direkt aus ablauforganisatorischen Überlegungen: Probanden müssen sich zu Beginn des Experiments anmelden und während der gesamten Experimentdurchführung eindeutig identifizierbar sein **(IX)**. Ihre im Experiment erhobenen Verhaltensdaten müssen strukturiert **(X)**, gespeichert **(XI)** und für die Auswertung verfügbar gemacht werden **(XII)**. Der Experimentator soll auf die Daten online zugreifen können **(XIII)**.

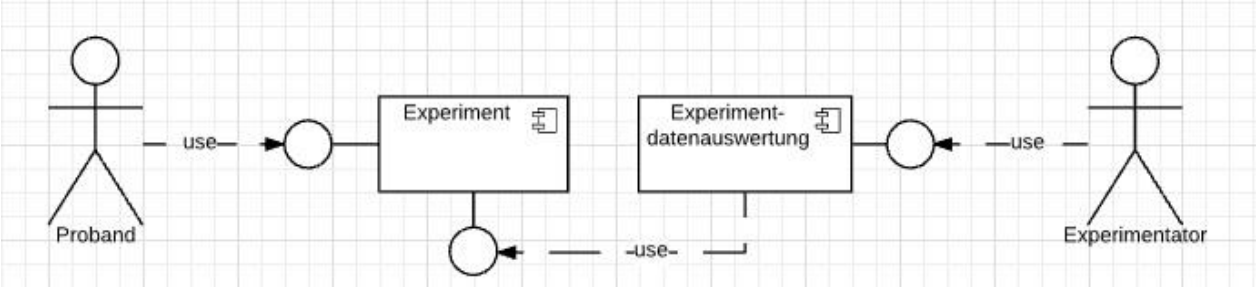
Nicht-funktionale Anforderungen (speziell die *Quality-of-Services*-Anforderungen), wie sie die Norm ISO/IEC 25010 (ISO/IEC 25010 2011; zitiert in Wikipedia 2017) aufzählt, spielen bis auf die Benutzbarkeit **(XIV)** und Informationssicherheit **(XV)** eine untergeordnete Rolle. Die Begründung hierfür liegt in der Natur des Experiments: es ist nicht kommerziell, einmalig angelegt, begrenzt auf einen konkreten Zeitraum (vier Wochen) und nicht ambitioniert in Bezug

auf erwartbare Zugriffszahlen. Somit werden Aspekte wie Leistungsfähigkeit, Betriebssicherheit, Verfügbarkeit und Wartbarkeit in dieser Betrachtung außen vor gelassen.

Die Benutzbarkeit ist hingegen wesentlich für die angestrebte Bindung der Probanden für die gesamte Dauer des Experiments. Probanden partizipieren freiwillig am Experiment, auch wenn über eine Teilnahmelotterie moderate monetäre Anreize geschaffen werden. Das hat zur Folge, dass die Probanden überwiegend intrinsisch motiviert sind. Ein starkes Motiv dabei ist Neugier auf Inhalt und Ergebnis des Experiments (Göritz 2006). Ist die Handhabung nicht zufriedenstellend gebrauchstauglich nach Definition von DIN EN ISO 9241, Teil 11 (DIN EN ISO 9241-110, zitiert in Wikipedia 2016), so ist ein ungewünschter, vorzeitiger Abbruch des Experiments wahrscheinlich. In Bezug auf die Informationssicherheit gilt es insbesondere den Zugriff auf die soziometrischen Daten so sicher zu gestalten, wie es das erwartbare Risiko einer unberechtigten Einflussnahme verlangt.

Das System „Experiment“ (selbst noch „*Black Box*“) und seine Grenzen, inklusive Schnittstellen und identifizierten Akteuren, lassen sich nun übersichtlich in einem Kontextdiagramm darstellen (Abbildung 2). Als Modellierungssprache wird die im Software-Engineering etablierte und weithin verwendete Unified Modeling Language (UML) in der aktuellen Version 2.5 gewählt (Object Management Group 2015).[5] Sie findet ebenfalls in allen nachfolgenden Abbildungen in Skizzenform Anwendung, deren Quellenangabe mit dem Zusatz „Realisiert mit *Lucidchart*“ versehen ist.[6]

Abbildung 2 Kontextdiagramm.



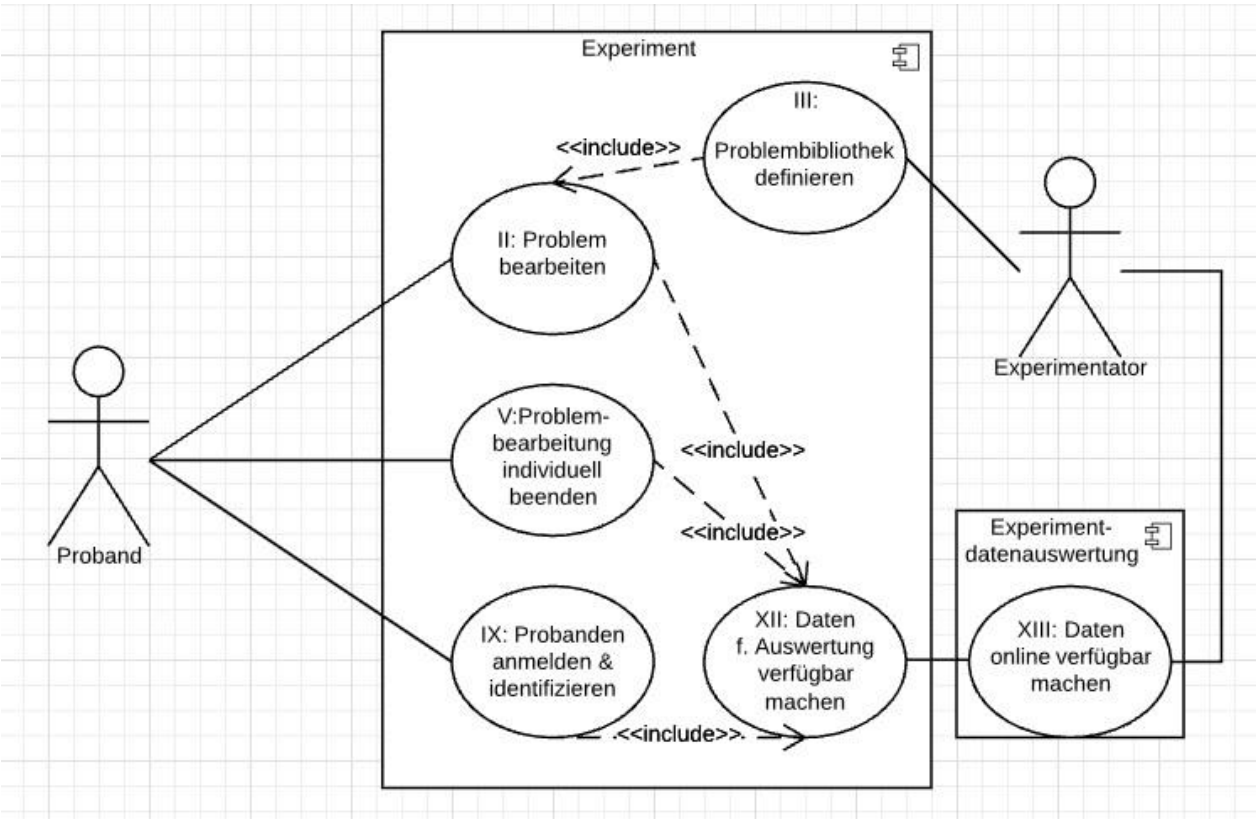
Quelle: Eigene Darstellung. Realisiert mit *Lucidchart*.

[5] Eine Erläuterung zu Syntax und Semantik von UML 2.0 findet sich zum Beispiel bei Rupp et al. (2012).
[6] *Lucidchart* ist eine online verfügbare Software, die die Erstellung von Diagrammen im Baukastenprinzip unterstützt (Lucidchart 2017).

Probanden greifen über eine Schnittstelle auf das Experiment zu. Über eine weitere Schnittstelle erhält die Experimentdatenauswertung nutzenden Zugriff. Letztere stellt selbst eine Schnittstelle für den Experimentator bereit.

Aus der obigen Beschreibung des Experiments ergeben sich zwei nicht-funktionale und dreizehn funktionale Anforderungen an die technische Umsetzung, welche im folgenden Kapitel systematisch aufgestellt wird. Zur Übersicht sind diejenigen funktionalen Anforderungen an Experiment und -datenauswertung abschließend in einem *Use-Case*-Diagramm zusammengefasst, die Interaktionen mit den Akteuren *Proband* und *Experimentator* außerhalb des Systems definieren (Abbildung 3)[7].

Abbildung 3 *Use-Case*-Diagramm.



Quelle: Eigene Darstellung. Realisiert mit *Lucidchart*.

Im Allgemeinen stellt dieser Diagrammtyp das für einen Anwender sichtbare Verhalten eines Systems dar (Rupp et al. 2012). Jeder Use-Case (Darstellung in Abbildung 3 als Oval mit Referenznummer und Bezeichner) impliziert eine Kette von Einzelschritten, die zur Erfüllung der

[7] Die neun nichtgezeigten Anforderungen stellen systeminterne Funktionalitäten ohne Schnittstelle zu anderen Akteuren und Systemen dar und werden deshalb üblicherweise nicht in einem Use-Case-Diagramm abgebildet.

Funktionalität erforderlich ist. Kleuker (2013) erfasst auf diese Weise die Kernfunktionalitäten des zu modellierenden Systems aus Anwenderperspektive in abstrahierter Form. Zur Verdeutlichung der Verbindung der Use-Cases untereinander sind mit <<*include*>> markierte Pfeile eingetragen, wobei die Pfeilrichtung den Informationsfluss zwischen den Use-Cases beschreibt. Die Use-Cases werden als Grundlage für die Ablaufmodellierung verwendet. In Kapitel 3.3 wird für einen Use-Case der Ablauf exemplarisch entwickelt.

2.2. Vergleich mit dem Leistungsspektrum von MouselabWeb

MouselabWeb ist ein wissenschaftliches *open source*-Projekt von Willemsen und Johnson (2010). Es unterstützt den Aufbau von online-Experimenten, die das Entscheidungsverhaltens mithilfe einer Computermaus abbilden und analysieren.[8] Die Entwicklung basiert auf der Arbeit von Johnson et al. (1989), die das Prinzip digitalisiert haben. Eine Auflistung der Funktionalitäten von MouselabWeb, findet sich in der Software-Dokumentation (Willemsen und Johnson 2008).

Im Folgenden wird auf Grundlage der Dokumentation geprüft, ob MouselabWeb die 15 herausgearbeiteten Spezifikationen dieses Experiments bedienen kann. Die nachfolgende Tabelle 1 enthält das Ergebnis dieser Prüfung. Dabei markiert ein Haken (✓) die vollständige Bereitstellung der Funktionalität und ein Kreuz (×) das vollständige oder teilweise Fehlen. Die Spezifizierung des Fehls ist in der Spalte Bemerkung aufgezeigt.

Tabelle 1 Vergleich zwischen geforderten und bereitgestellten Funktionalitäten.

Funktionalität	**bereit-gestellt**	**Bemerkung**
I Problemstellung als Spiel in Normal-Form dargestellt	×	Matrixdarstellung von Informationen wird unterstützt; Design von Spielen in Normal-Form und Inhalte müssen ergänzend erstellt werden.
II Proband übernimmt Spieler-Rolle	✓	

[8] Die Methode wird als *Mousetracking* bezeichnet und gehört zu den *process tracing techniques*. Eine Einführung bieten Kühberger und Schulte-Mecklenbeck (2011).

III Experiment umfasst 16 Spiele, unterbrochen von Pausen	✓	
IV maximale Bearbeitungszeit ist vorgegeben	✓	
V Spieler kann Spiel beenden	✓	
VI Daten zu Zeit, Ort und Art jeglicher Handlung erfasst	×	HTML-Mouse Events[9] onmouseover, onmouseout, onclick und ondblclick werden unterstützt; onmousemove wird nicht unterstützt.
VII Experiment wird online durchgeführt	×	Dateien sind in HTML, Java Script und PHP codiert; Website muss selbst eingerichtet werden.
VIII PC-Maus wird genutzt	✓	
IX Probanden sind angemeldet und durchgängig identifiziert	✓	
X erhobene Daten sind strukturiert	×	Formular-Vorlagen existieren; Formulare müssen gem. VI angepasst werden.
XI erhobene Daten sind gespeichert	×	Eine Vorlage-Datei zum Speichern von Daten auf PHP-Basis existiert; eine eigene Experiment-Datenbank muss eingerichtet und betrieben werden und die Vorlage-Datei muss auf die verwendete Datenbank angepasst werden.
XII erhobene Daten sind für Auswertung verfügbar	×	Siehe XI.
XIII Experimentator kann online auf Experimentdaten zugreifen	×	Siehe XI.
XIV Website ist benutzbar	✓	
XV Informationen auf Website sind sicher	✓	

Quelle: Eigene Darstellung.

[9] Eine Beschreibung der Events findet sich zum Beispiel bei HTML Event Attributes 2017. Onmousemove reagiert auf jedwede Mausbewegung und ist damit geeignet, den Pfad der Maus aufzuzeichnen.

Somit ergibt sich Modifikationsbedarf an insgesamt sieben Funktionalitäten, die die Software MouselabWeb betreffen (I, VI, X, XI), die Erstellung eigener Software-Lösungen benötigen (VII) oder die Einrichtung und Bereitstellung eigener Datenbank-Komponenten erfordern (XI-XIII). Ein Konzept zur Umsetzung dieser Modifikationen im Rahmen des Experiments als Gesamtsystems ist Gegenstand des folgenden Kapitels.

3. Technische Realisierung

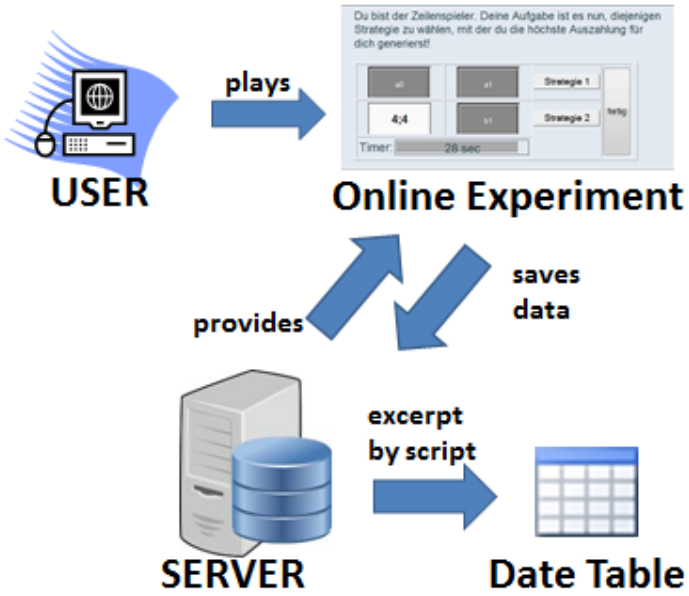
In diesem Abschnitt wird aus den in Kapitel 2 herausgearbeiteten 15 Software- und Systemspezifikationen einen Entwurf zur softwaretechnischen Umsetzung als Online-Experiment erstellt. Dabei wird der im vorhergehenden Kapitel ermittelten Modifikationsbedarf an der zu nutzenden Software MouselabWeb berücksichtigt. Das nachfolgend entworfene Konzept bildet für diese Anpassungen die Grundlage. Bei diesem Vorgehen werden Darstellungsformen des Software-Engineerings genutzt, wie sie für den Software-Entwurf im Rahmen eines Software-Projekts üblich sind (Kleuker 2013). Zunächst wird ein Systemüberblick in Form eines Komponentendiagramms entworfen, welches den Fokus auf die technischen Komponenten und ihren Verbindungen legt (Kapitel 3.1). In Kapitel 3.2 wird dann der Modellaufbau auf Darstellungsebene eines Klassendiagramms entworfen. Aus diesem sind alle relevanten Strukturzusammenhänge und Datentypen ablesbar. Schließlich wird exemplarisch für einen Use-Case ein Ablaufmodell in Form eines Aktivitätsdiagramms (Kapitel 3.3) entwickelt. Auf diese Weise wird bestimmt, wie das System die Aufgabe des Use-Cases löst.

3.1. Systemmodellierung

In Abbildung 4 ist der technische Versuchsaufbau mit seinen wesentlichen Komponenten entworfen. Aus dieser geht bereits die einleitend erwähnte Absicht hervor, Probanden über das Internet auf das Experiment zugreifen zu lassen. Ein multifunktionaler Server soll dazu Informationen bereitstellen und strukturiert speichern, um sie für die Auswertung verfügbar zu machen. Von diesem Aufbau ausgehend, werden die technische Struktur des Experiments zur Laufzeit modelliert. Um die Integration des Systems in seiner Umwelt abzubilden, wird üblicherweise das Komponentendiagramm gewählt. Dieses gibt einen Überblick über die Systemstruktur und enthält die einzelnen technischen Komponenten, die zur Laufzeit benötigt werden sowie ihrer Schnittstellen und daraus resultierender Abhängigkeiten untereinander (Rupp

et al. 2012). Abbildung 5 zeigt das Komponentendiagramm für die hier betrachtete Aufgabenstellung.

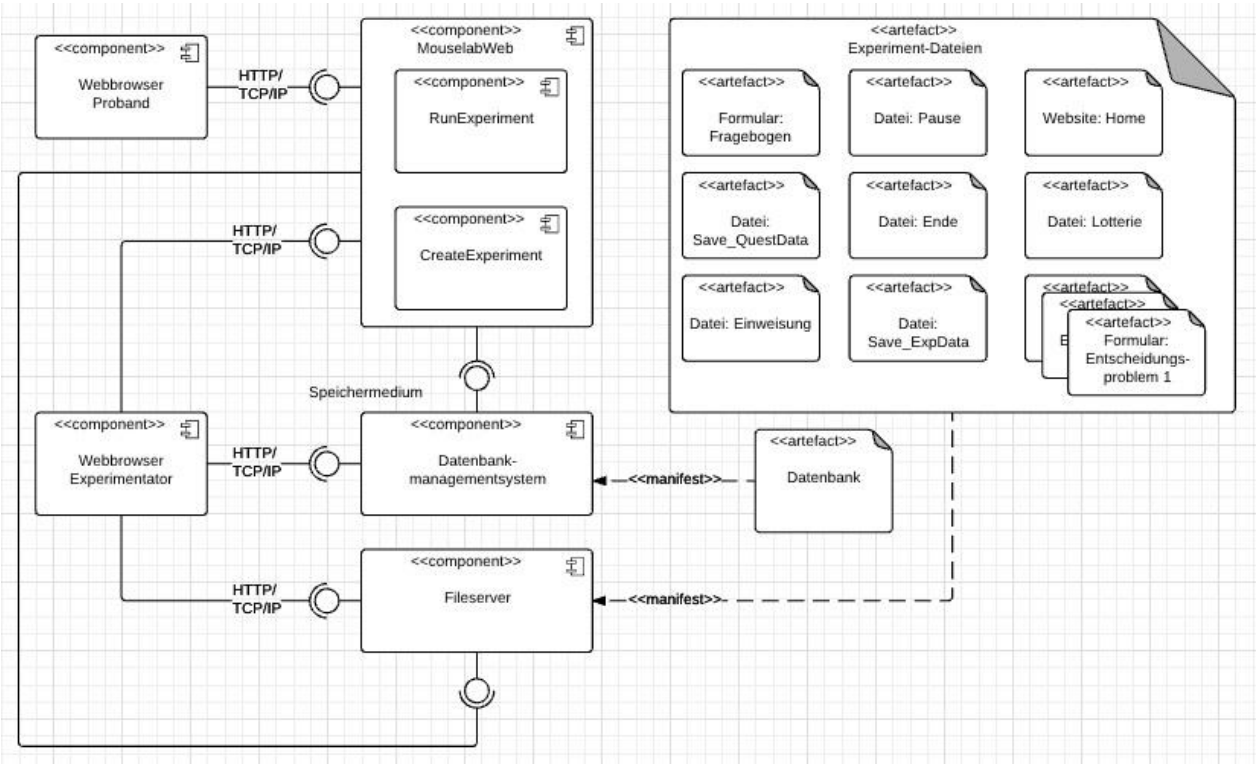
Abbildung 4 Technischer Versuchsaufbau des Experiments.



Quelle: Eigene Darstellung. Zum Teil realisiert mit Clip Arts von *MS-PowerPoint*.

Probanden greifen über ihren Internetbrowser und einer Internetverbindung auf die Internetseite des Experiments zu. Diese wird bereitgestellt von MouselabWeb. Die Software übernimmt die Funktionen eines Applicationservers, welcher das Experiment zur Laufzeit komponiert und alle Interaktionen handhabt (*RunExperiment*). Dazu gehört auch das Zwischenspeichern von Formulardaten, bevor diese in die Datenbank übernommen werden. Die erforderlichen Dateien (in der Abbildung teilweise spezifiziert als Formular- oder Website-Datei) bezieht MouselabWeb über den File-Server, welcher alle Experiment-Dateien einbindet. Ebenfalls Teil von MouselabWeb ist die Komponente *CreateExperiment*. Sie ermöglicht dem Experimentator ein Experiment zu entwerfen. Dessen Zugriff erfolgt über das Internet. Die Ergebnisse des Experiments werden von MouselabWeb an das Datenbankmanagementsystem (DBMS) übergeben und von dort in die eingebundene Datenbank geschrieben. Der Experimentator besitzt über das Internet (passwortgeschützten) Zugriff auf das DBMS, welches wiederum Daten auf Anfrage aus der Datenbank (DB) extrahieren kann. Schließlich besitzt der Experimentator über das Internet Zugriff auf den File-Server um die Experiment-Dateien zu verwalten.

Abbildung 5 Komponentendiagramm.



Quelle: Eigene Darstellung. Realisiert mit *Lucidchart*.

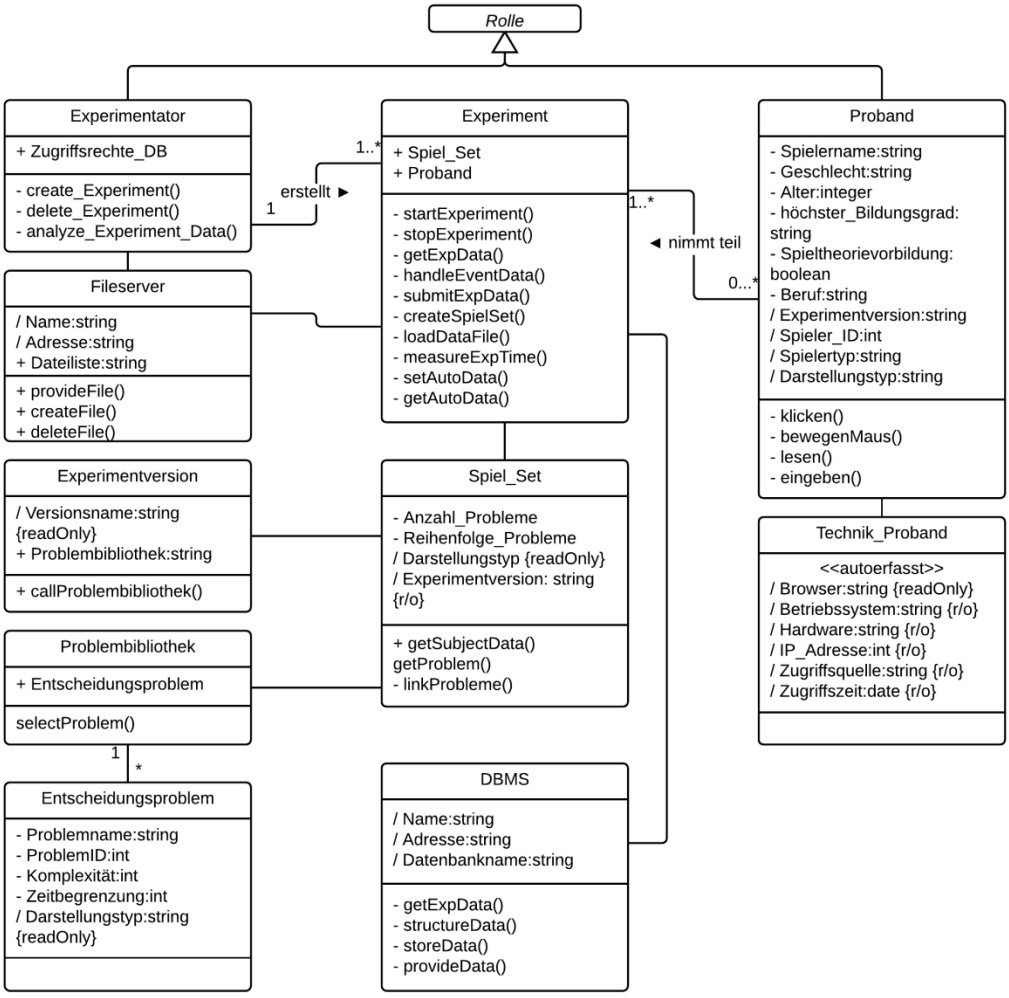
Im Hinblick auf die technische Realisierung ergibt sich damit folgender Stand: die Komponente MouselabWeb und seine Schnittstellen sind ebenso verfügbar wie die Komponenten von Probanden und Experimentator. Alle anderen softwaretechnischen Komponenten und Artefakte des Systems müssen angepasst, erstellt und eingebunden sowie Datenbank und (physischer) File-Server bereitgestellt werden. Die konkreten Modifikationen der Software MouselabWeb und die zusätzlich beizubringenden Software-Funktionalitäten werden im nächsten Schritt näher bestimmt durch die Aufbaumodellierung.

3.2. Aufbaumodellierung

Die Modellierung des inneren Systemaufbaus wird häufig über ein Klassendiagramm realisiert. Auf diese Weise lassen sich u.a. Klassen, ihre Schnittstellen und Assoziationen zwischen ihnen darstellen (Rupp et al. 2012). Ein gängiges Vorgehen zur Identifikation von Klassen und Objekten aus den Anforderungen beschreibt Kleuker: „Zum Finden von Klassen werden die bisher erstellten [Anforderungs-]Texte analysiert und für jeden Satz geprüft, ob hier ein Objekt

oder die Beschreibung eines Objekts benutzt wird […]“ (Kleuker 2013, S. 95). Dabei ist die Detailtiefe abhängig vom Projektstand. Für das erste Klassendiagramm eines neuen Systems steht die korrekte und verständliche Abbildung von Beziehungen zwischen den zentralen Elementen des Systems im Vordergrund (Rupp et al. 2012). Diesen Empfehlungen folgend, sind aus den Anforderungen die wesentlichen Objekte identifiziert und zu Klassen abstrahiert. Das Ergebnis dieses Vorgehens ist in Abbildung 6 gezeigt.

Abbildung 6 Klassendiagramm.



Quelle: Eigene Darstellung. Realisiert mit *Lucidchart*.

Innerhalb des Systems lassen sich die beiden Rollen Experimentator und Proband den Akteuren im Experiment zu ordnen. Dabei ist der Experimentator für die Erstellung von Experimenten zuständig und Probanden für die Teilnahme. Bei den Probanden ist ferner von Interesse, mit

welcher Technik sie das Experiment durchführen. Das Experiment wird definiert durch ein bestimmtes Spiel-Set, welches sich wiederum aus einer bestimmten Anzahl an Entscheidungsproblemen zusammensetzt. Diese entstammen der Problembibliothek, welche über eine Vielzahl von Entscheidungsproblemen verfügt, die in verschiedenen Parametern variieren. Die Zusammensetzung des Spiel-Sets richtet sich ferner nach der zugewiesenen Experimentversion, welche jeweils eine unterschiedliche Problembibliothek besitzen.

Um diesen Aufbau realisieren zu können, müssen gemäß Tabelle 1 Anpassungen an MouselabWeb getätigt und weitere Voraussetzungen geschaffen werden. Die Klasse Experiment wird dem Aufgabenbereich der Komponente MouselabWeb zugeordnet, da hier viele Tätigkeiten eines Applicationservers zu bewältigen sind. Innerhalb der Klasse Experiment stehen eine Anzahl an Operationen durch MouselabWeb bereits zur Verfügung. Dazu gehören startExperiment(), stopExperiment(), loadDataFile() und measureExpTime(). Andere müssen ergänzt (getAutoData(), getExpData() und handleEventData()) oder erstellt werden (createSpiel-Set() und setAutoData()).Alle anderen Klassen müssen im Rahmen der Umsetzung erstellt werden, wobei MouselabWeb die Schnittstellen zu diesen Klassen bereits vorhält.

Ausgehend von dieser Darstellung des statischen Systems soll im nächsten Schritt das dynamische Verhalten in einer konkreten Anforderungssituation abgebildet werden. Dabei ist zu erwarten, dass sich aus dieser Abbildung insbesondere bei den Operationen der Klassen Ergänzungen ergeben, welche den Detaillierungsgrad weiter erhöhen (Rupp et al. 2012, S. 264).[10]

3.3. Ablaufmodellierung

Die Darstellung von komplexen Abläufen erfolgt üblicherweise in Aktivitätsdiagrammen, welche – ähnlich einer Prozessabbildung – die Aktivitäten der wesentlichen Systemkomponenten, Klassen und Akteure verbinden (Rupp et al. 2012). So lassen sich unteranderem die Abarbeitung von Use-Cases strukturiert veranschaulichen. Dieses soll hier geschehen, um vom bisher statisch betrachteten System das dynamische Verhalten abzubilden. Aufgrund von Umfangsrestriktionen kann an dieser Stelle nur einer der in Kapitel 2 identifizierten Use-Cases in ein Aktivitätsdiagramm entwickelt werden. Der zentrale und deshalb für die Darstellung ausgewählte

[10] Diese Ergänzungen haben sich in der Erarbeitungsphase dieser Arbeit tatsächlich ergeben. Sie sind hier allerdings nicht explizit angegeben, um die Konsistenz und Vollständigkeit der Darstellungen aus den Kapiteln 3.1 und 3.2 zu erhalten.

Use-Case für Probanden und Experimentator ist zweifelsohne *II Problem bearbeiten*. Dieser umfasst die Lösung einer Problemstellung innerhalb des Experiments durch den Probanden und generiert Verhaltensdaten zur Analyse für den Experimentator. Seine softwaretechnisch korrekte Realisierung ist deshalb elementar für den Erfolg des Forschungsvorhabens. Bei seiner Darstellung in Abbildung 7 sind die Verantwortungsbereiche eingetragen, um Aktivitäten direkt den jeweiligen Systemkomponenten, Klassen und Akteuren zuordnen zu können.

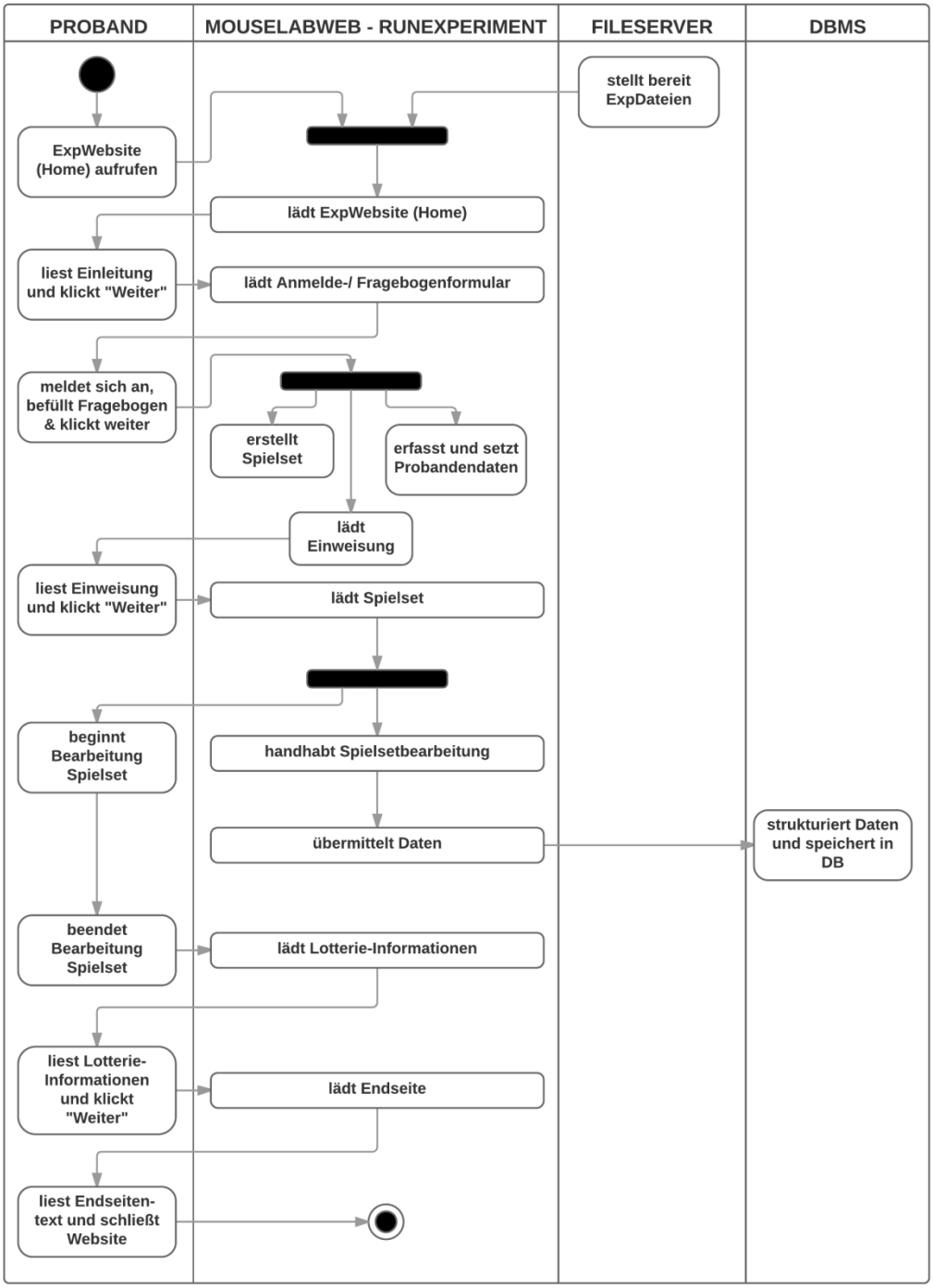
Die dargestellten Aktivitäten von MouselabWeb, Fileserver und DBMS finden sich größtenteils als Operationen der zugeordneten Klasse aus der Aufbaumodellierung wieder. Dabei ist die Klasse Proband als Akteur mit seinen Operationen lesen(), klicken(), eingeben() und Problem_bearbeiten() unter Spezifizierung des Kontextes in seinem Verantwortungsbereich abgebildet. Die im Klassendiagramm eingetragene Operation bewegenMaus() wird in dem vorliegenden Detaillierungsgrad des Aktivitätsdiagramms nicht verwendet. Um sie abbilden zu können, bedarf es einer näheren Spezifizierung der Aktivitäten. Da der Proband mithilfe der Computermaus das Experiment durchläuft, wird sich die Operation bewegenMaus() in allen seinen Aktivitäten wiederfinden.

Ähnlich verhält es sich mit den Operationen der Klasse Experiment, welche im Aktivitätsdiagramm dem Akteur (und gleichzeitiger Systemkomponente) MouselabWeb - RunExperiment zugeordnet wird. Da MouselabWeb in wesentlichen Funktionen modifiziert werden soll, ist es hilfreich, die Darstellung der Aktivitäten weiter zu detaillieren. Dies ist nachfolgend in Abbildung 8 für die Aktivität *handhabt Spiel-Setbearbeitung* bis auf Operationsebene der Klasse durchgeführt und in ein eigenes Aktivitätsdiagramm überführt.

Die Aktivität *handhabt Spiel-Setbearbeitung* umfasst sämtliche Aktivitäten, die die Software während der Bearbeitung des Experiments durch einen Probanden durchführen muss. Ein Spiel-Set besteht stets aus 16 einzelnen Spielen. Für jedes Spiel müssen von MouselabWeb - RunExperiment dieselben Aktivitäten durchgeführt werden. Die Aktivität *handhabt Spiel-Setbearbeitung* lässt sich deshalb gut in Schleifenform modellieren. Innerhalb der Schleife finden sich die Aktivitäten zum Eventhandling, Zeit messen und Experimentdaten erfassen, welche den Klassenoperationen handleEventData(), measureExpTime() und getExpData() entsprechen. Sie laufen während der Problembearbeitung des Probanden permanent ab. Ist die maximale Bearbeitungszeit erreicht oder beendet der Proband die Bearbeitung, folgen die Aktivitäten zum

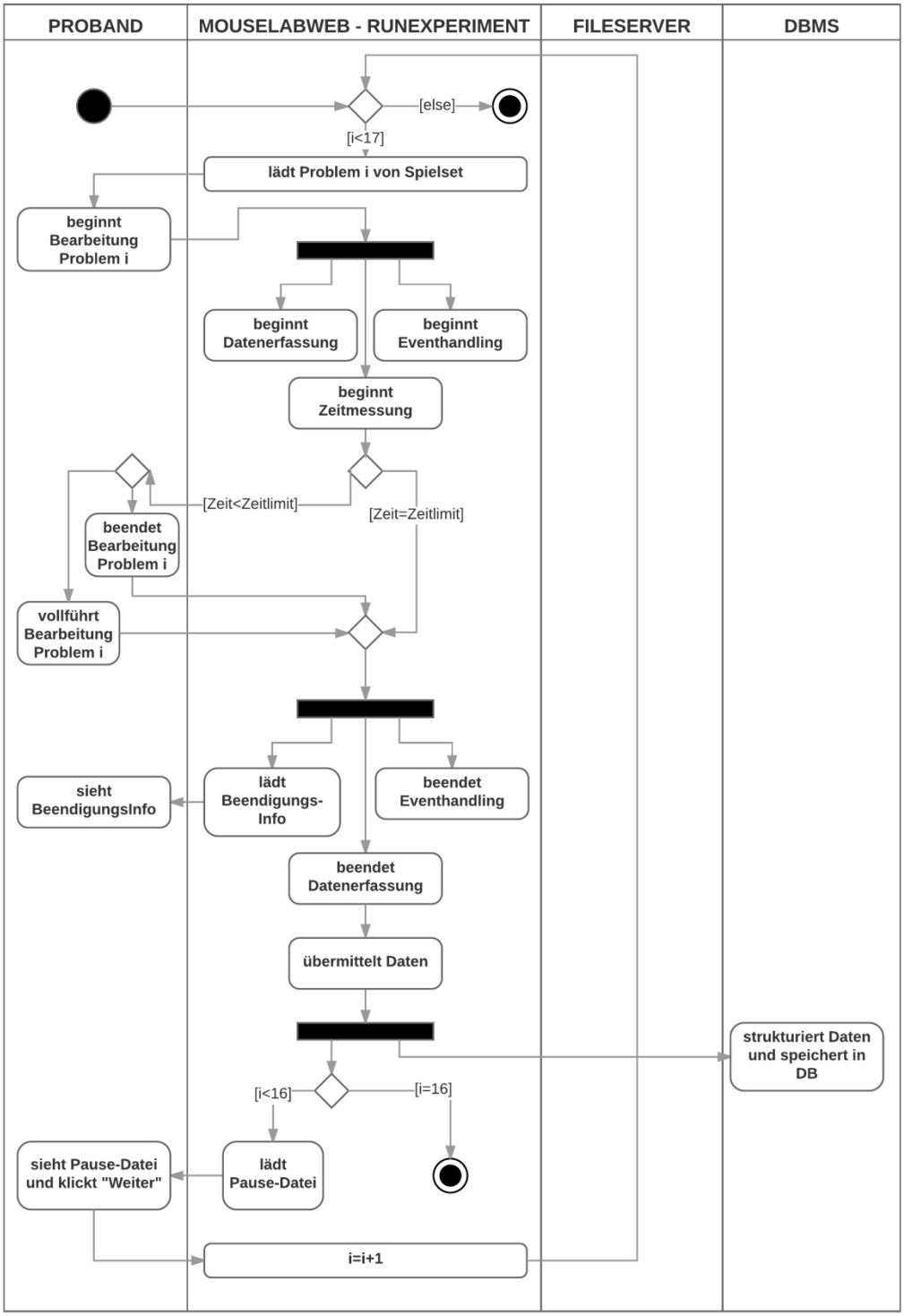
Übermitteln der Daten an das DBMS und die Vorbereitung des folgenden Problems. Sie entsprechen den Operationen submitData() und loadDatafile().

Abbildung 7 Aktivitätsdiagramm von Use-Case *II Problem bearbeiten*.



Quelle: Eigene Darstellung. Realisiert mit *Lucidchart*.

Abbildung 8 Aktivitätsdiagramm von [*handhabt Spiel-Setbearbeitung*].



Quelle: Eigene Darstellung. Realisiert mit *Lucidchart*.

Die vorliegenden Aktivitätsdiagramme können nun in mit einander verknüpfte Algorithmen umgesetzt werden, welche die jeweils geforderten Funktionalitäten abbilden.

4. Schlussbetrachtung

Das Ziel dieser Projektarbeit war die systematische Entwicklung eines Konzepts für die technische Umsetzung eines Online-Experimentes auf der Basis der Experiment-Software MouselabWeb. Dieses konnte mit Hilfe von Methoden des Software-Engineerings erreicht werden. Das Konzept fußt auf einer umfassenden Spezifikation der Anforderungen (Kapitel 2) bis auf Ebene *Use-Cases*. Ein Vergleich der geforderten Funktionalitäten mit den von MouselabWeb zur Verfügung gestellten, ermöglichte eine dezidierte und gezielte Ermittlung des Anpassungsumfangs. Auf dieser Grundlage konnten in den Folgeschritten das Experiment als (software-)technisches System und sein Aufbau modelliert werden.

Das System besitzt als zentrales Element die Software MouselabWeb, die bereits über Schnittstellen zu allen anderen Komponenten des Systems verfügt. Allerdings besteht Anpassungsbedarf bei der Erweiterung von Funktionalitäten, welcher auf der Grundlage der Aufbauübersicht und der Aktivitätsdiagramme umgesetzt werden kann. Die Erstellung und Anbindung von Systemkomponenten, wie der Datenbank samt Datenbankmanagementsystem und Fileserver inklusive benötigter Experimentdateien bilden den zweiten großen Block der Umsetzung. Insbesondere bei der Erstellung der Problembibliothek müssen die Inhalte aus dem Experimentdesign ermittelt werden. Hilfreich für die Zuordnung von Funktionalitäten und Einordnung der Dateien nach ihrer Verwendung im Experiment ist dabei die Ablaufmodellierung der Use-Cases. In dieser Arbeit wurde der Ablauf eines Use-Cases exemplarisch vorgestellt.

Für die Umsetzung stehen damit die Systemübersicht in Form eines Komponentendiagramms, der innere Aufbau des Systems in Form eines Klassendiagramms und ein Ausschnitt der Wirkungsweise in Form von Aktivitätsdiagrammen zur Verfügung. Diese bilden den in dieser Arbeit avisierten (software-)technischen Entwurf des Systems. Er kann in einem (außerhalb dieser Betrachtung liegenden) Folgeschritt implementiert werden.

Literaturverzeichnis

ISO/IEC 25010, 2011: 25010:2011 Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models. Online verfügbar unter https://www.iso.org/standard/35733.html (kostenpflichtig), zuletzt geprüft am 01.07.2017.

DIN EN ISO 9241-110, 2008-09: Ergonomie der Mensch-System-Interaktion - Teil 110: Grundsätze der Dialoggestaltung. Online verfügbar unter https://www.beuth.de/de/norm/din-en-iso-9241-110/110514174, zuletzt geprüft am 01.07.2017.

Gintis, Herbert (2009): Game theory evolving. A problem-centered introduction to modeling strategic interaction. 2. ed. Princeton, NJ: Princeton Univ. Press.

Göritz, Anja S. (2006): Incentives in web studies: Methodological issues and a review. In: *International Journal of Internet Science* 1 (1), S. 58–70.

HTML Event Attributes (2017). Online verfügbar unter https://www.w3schools.com/tags/ref_eventattributes.asp, zuletzt aktualisiert am 18.08.2017, zuletzt geprüft am 18.08.2017.

Johnson, Eric J.; Payne, John W.; Schkade, David A.; Bettman, James R. (1989): Monitoring information processing and decisions the mouselab system. Auftragsforschung.

Kleuker, Stephan (2013): Grundkurs Software-Engineering mit UML. Der pragmatische Weg zu erfolgreichen Softwareprojekten. 3., korr. und erw. Aufl. Wiesbaden: Springer Vieweg (SpringerLink : Bücher).

Kühberger, Anton; Schulte-Mecklenbeck, Michael (Hg.) (2011): A handbook of process tracing methods for decision research. A critical review and user's guide: Psychology Press. Online verfügbar unter http://lib.myilibrary.com/detail.asp?ID=293053.

Kühlberger, Anton; Schulte-Mecklenbeck, Michael; Ranyard, Rob (2011): Introduction: Windows for understanding the mind. In: Anton Kühberger und Michael Schulte-Mecklenbeck (Hg.): A handbook of process tracing methods for decision research. A critical review and user's guide: Psychology Press, S. 1–17.

Lucidchart (2017): Flowchart Maker & Online Diagram Software | Lucidchart. Online verfügbar unter https://www.lucidchart.com/, zuletzt aktualisiert am 21.08.2017, zuletzt geprüft am 21.08.2017.

Object Management Group (2015): UML 2.5. Hg. v. Object Management Group. Online verfügbar unter http://www.omg.org/spec/UML/2.5/, zuletzt geprüft am 01.07.2017.

Rupp, Chris; Queins, Stefan; die SOPHISTen (2012): UML 2 glasklar. Praxiswissen für die UML-Modellierung. 4., aktualis. u. erw. Aufl. München: Hanser.

Wikipedia (2016): Benutzerfreundlichkeit. Online verfügbar unter https://de.wikipedia.org/wiki/Benutzerfreundlichkeit, zuletzt aktualisiert am 25.10.2016, zuletzt geprüft am 01.07.2017.

Wikipedia (2017): ISO/IEC 9126. Online verfügbar unter https://de.wikipedia.org/wiki/ISO/IEC_9126, zuletzt aktualisiert am 22.01.2017, zuletzt geprüft am 01.07.2017.

Willemsen, Martijn C.; Johnson, Eric J. (2008): MouselabWEB Documentation. Version 1.00beta, Aug 14, 2008. Online verfügbar unter http://www.mouselabweb.org/download.php, zuletzt geprüft am 18.08.2017.

Willemsen, Martijn C.; Johnson, Eric J. (2010): Visiting the decision factory: observing cognition with MouselabWEB and other information acquisition methods. In: *A handbook of process tracing methods for decision making*, S. 21–42.