Hochschule Reutlingen
Reutlingen University

virtualreality
LABORATORY

Dokumentation

orbis novem

Reutlingen, 10.07.2019

Inhaltsverzeichnis

1 Einleitung

Das Themenfeld der virtuellen Realität (VR) erweist sich als ein stetig wachsendes und spannendes Themenfeld innerhalb der Informatik. Derzeit wird VR, also das Platzieren dreidimensionaler Inhalte in einer digitalen Umgebung, vor allem mit der Computerspiele-Industrie assoziiert. Jedoch finden sich auch zahlreiche Einsatzgebiete in anderen Bereichen, wie der Architektur, der Medizin, dem Militär, dem Bildungswesen oder der medialen Berichterstattung. Das Erzeugen virtueller und die Gestaltung potenziell realitätsunabhängiger Umgebungen nach den eigenen Vorstellungen ist mit einem immensen Grad an schöpferischer Freiheit verbunden. Zudem ist die Exploration dieser Welten als Medium ein neuartiges Erlebnis für Rezipienten, also den Benutzern der jeweiligen VR-Anwendung. Dieses Medium bietet einen hohen Freiheitsgrad, besonders bezüglich der Interaktion.
Im Rahmen des Wahlfachs Virtual Reality Labor (VR Labor) des Masterstudiengangs Human Centered Computing der Hochschule Reutlingen werden von Studierenden verschiedene Projekte rund um das Themenfeld VR geplant und durchgeführt. Rezipienten werden mittels interaktionsbasierten Anwendungen für eine Reihe an Endgeräten und Systemen die Interaktion mit computergenerierten Welten ermöglicht. Eines der VR Labor-Projekte und Gegenstand der vorliegenden Dokumentation ist das orbis novem Projekt.

Dieses plant, zwei bereits entwickelte studentische Softwarekomponenten und diverse, vom VR Labor zur Verfügung gestellte Hardwarekomponenten zu kombinieren. Ziel dieser Kombination ist es, ein System anzubieten, mittels welchem ein Rezipient eine Immersion in einer VR-Umgebung erleben kann.
Der Projektname *orbis novem,* vom lateinischen für „Welt Neun“, ist dabei eine Anspielung auf die verwendete Neue Welt 9 VR-Umgebung.

1.1 Motivation und Zielsetzung

Innerhalb eines Projekts des VR Labors wird von Studierenden das orbis novem System entwickelt. Dieses System kombiniert mehrere VR Labor-Projekte und Hardware mit dem Ziel, eine vollständige Immersion für einen Rezipienten anzubieten. Der Begriff der Immersion wird im Folgenden kurz erklärt und in Kapitel 2 formal eingeführt.
Die grundlegende Motivation für das Projekt besteht darin, verschiedene an der Hochschule existierende Projekte zusammenzuführen und für das VR Lab (in einer neuen Form) nutzbar zu machen. Gleichzeitig sollen die Ergebnisse dieser studentischen Arbeiten erhalten bleiben und als Vorarbeit für weitere Projekte einen neuen Zweck erfüllen.

Eine VR-Anwendung verfolgt oftmals das Ziel, einem Rezipienten, während der Dauer der Nutzung, das überzeugende Gefühl zu geben, dass die aktuell erlebte VR-Umgebung real ist. Der Rezipient soll sich beim Explorieren der virtuellen Realität völlig in dieser verlieren [7]. Dies ist vergleichbar mit dem kurzzeitigen Eintauchen in die Phantasiewelt eines guten Buches oder

Theaterstücks. Mit anderen Worten soll eine vollständige Immersion (dt. Eintauchen) erreicht werden.
Bei einer Immersion handelt es sich um ein subjektiv erlebtes, vollständiges Eintauchen in eine VR-Umgebung, welche für den Rezipienten kurzzeitig die bekannte Realität ersetzt [2].
Konkreter gesagt beschreibt Immersion also den hervorgerufenen Effekt, welcher das Bewusstsein des Rezipienten, illusorischen Stimuli ausgesetzt zu sein, so weit in den Hintergrund treten lässt, dass die VR-Umgebung kurzzeitig als real empfunden wird [2].
Die vollständige Immersion wird dabei als ein Gefühl der völligen Abgrenzung von der Realität beschrieben und in diesem Sinne in dieser Ausarbeitung verwendet [7].

Auch das orbis novem System zielt also darauf ab, es dem Rezipienten zu ermöglichen, eine vollständige Immersion zu erleben. Auf die weiteren Ziele des Systems wird in Kapitel 3 explizit eingegangen.

1.2 Projektplanung

Das orbis novem System ist Bearbeitungsgegenstand des orbis novem Projekts im VR Lab, welches grundsätzlich auf zwei Semester ausgelegt ist. Im Verlauf der Bearbeitungszeit für das Projekt kam es leider immer wieder zu Unterbrechungen, was eine konsistente Projektplanung immer wieder zunichtemachte. Die Umstände sind hier kurz erläutert.

Die Grundlagen für das spätere orbis novem System wurden im Wintersemester 2017 / 2018 gelegt: Innerhalb dieses Wintersemesters bearbeiteten zwei Studenten, Benjamin und Claudiu, die erste Iteration des sogenannten iVR HUDs, eines späteren Teilsystems des orbis novem Systems. Deren Dokumentation des iVR HUD findet sich in [1]. In diesem Semester war auch Georgia bereits am Masterprojekt beteiligt. Während dieser Bearbeitungszeit entstand die Idee zum orbis novem System. Diese Idee wurde im Verlauf des Wintersemesters 2017 / 2018 von Georgia konkretisiert und Wissen über die späteren Teilsysteme erlangt. Im folgenden Sommersemester 2018 waren Benjamin und Claudiu nicht als Teilnehmer für das orbis novem System verfügbar und es fanden sich auch leider keine neuen Interessenten. Daher bearbeitete Georgia das Projekt im Sommersemester 2018 alleine und konnte es nicht regulär abschließen. Im Wintersemester 2018 / 2019 faden sich erneut keine Interessenten für das Projekt im VR Lab. Zudem wurde Georgia auf eigenen Antrag dieses Wintersemester als Urlaubssemester genehmigt. Im kommenden Sommersemester 2019 fanden sich dann endlich zwei weitere Interessierte, die sich am Projekt beteiligen wollten. Somit konnte das Projekt im Sommersemester 2019 abgeschlossen werden, wenn auch leider nicht im erwünschten Umfang da dazu mehr Teilnehmer nötig gewesen wären.

Die folgenden Studierenden sind damit im Sommersemester 2019 involviert:

- Baris Erol (Baris.Erol@Student.Reutlingen-University.de)
- Georgia König (Georgia_Marie-Theres.Koenig@Student.Reutlingen-University.de)
- Martin Watolla (Martin.Watolla@Student.Reutlingen-University.de)

Es sind damit besonders die Ergebnisse dieser Iteration, die im Folgenden betrachtet werden.

1.2.1 Meilensteine

Im Verlauf des Sommersemester 2019 wurde der Projektfortschritt anhand von drei Meilenstein-Meetings und einem Abschlussmeeting gemessen, wie es im VR Lab üblicherweise vom Projektmanagement zu Beginn des Semesters festgelegt wird. Anhand der Termine in der Abbildung 1 wurden die jeweiligen Termine vorbereitet.

20.03.19	Begrüßungsmeeting
27.03.19	
03.04.19	
10.04.19	TIC Career Day !Meeting findet statt
17.04.19	**Meilensteinmeeting**
24.04.19	Osterferien
01.05.19	Tag der Arbeit
08.05.19	Informatics Inside !Meeting findet statt
15.05.19	
22.05.19	**Meilensteinmeeting**
29.05.19	
05.06.19	
12.06.19	Pfingstferien
19.06.19	
26.06.19	**Meilensteinmeeting**
03.07.19	Semesterabschlussmeeting

Abbildung 1: Meilensteine im Sommersemester

Das Semesterabschlussmeeting wurde dabei in Absprache mit allen betreuenden Professoren vom 03.07. auf den 10.07. verlegt. Zudem wurde das Projekt und dessen Inhalte im Rahmen der Informatics Inside Konferenz in der Aula Reutlingen vorgestellt. Dabei wurde auch ein Poster (zu finden im BSCW) erstellt und die Teilnehmer der aktuellen Iteration beantworteten Fragen von Besuchern.

2 Grundlagen

In diesem Kapitel wird der für das Verständnis wichtige Begriff der Immersion eingeführt. Ebenso wird auf die einzelnen Teil- oder Subsysteme eingegangen, in die sich das geplante System einteilen lässt. Diese werden vom VR Lab zur Verfügung gestellt und im Folgenden kurz beschrieben.

2.1 Immersion

Wie bereits beschrieben, handelt es sich bei einer Immersion um ein subjektiv erlebtes, vollständiges Eintauchen in eine VR-Umgebung, welche für den Rezipienten kurzzeitig die bekannte Realität ersetzt. Die vollständige Immersion wird dabei als ein Gefühl der völligen Abgrenzung von der Realität beschrieben und in diesem Sinne in dieser Ausarbeitung verwendet.

Es gibt mehrere ähnliche Konzepte im Zusammenhang mit der Immersion, namentlich Fluss, Präsenz, Engagement, kognitive Absorption, narrative Beteiligung, Marionettentheater und Transport. Die Definitionen von Immersion sind vielfältig und facettenreich, aber die berühmteste ist die von Janet Murray. Nach ihr ist Immersion. "die angenehme Erfahrung, in einem aufwendig gestalteten simulierten Ort zu sein" und "das Gefühl, von einem Ort umgeben zu sein, welcher eine völlig andere Realität darstellt, die unsere ganze Aufmerksamkeit, unseren gesamten Wahrnehmungsapparat erregt". Diese Definition weist auf einige Schlüsselaspekte hin, wie z.B. der Transport, Simulation, Empfindung, Aufmerksamkeit und Wahrnehmung. Eine vollständige Immersion wurde von Brown & Cairns beschrieben, als ein Gefühl von "völlig von der Realität abgeschnitten sein." Coomans & Timmermans definiert Immersion als “ein Gefühl des Seins, tief in einer "Scheinwelt" verwickelt, als ob sie real wäre." Dieser Glaube, nach Salen & Zimmerman wird als "immersiver Irrtum" bezeichnet. Dieser tritt auf, wenn "die virtuelle Realität so vollständig ist, dass der Spieler glaubt, dass er oder sie Teil eines imaginären Körpers ist“. [7]

Im Folgenden wird vertieft auf verschiedene Arten von Immersion eingegangen, namentlich räumliche Immersion und emotionale Immersion. Räumliche Immersion (engl. spatial immersion) wird dabei beschrieben als die subjektive Erfahrung des Seins an einem Ort oder in einer Umgebung, auch wenn man sich physisch an einem anderen befindet. Dagegen bezeichnet die emotionale Immersion (engl. emotional immersion) eine positive empathische

Verbindung zum Avatar in der Spielewelt dar. Emotionale Immersion wird oft erlebt als das Gefühl, sich so stark mit dem eigenen Charakter zu identifizieren, dass er zur primären Identität wird. Die emotionale Immersion ist zudem einnehmender als die räumliche Immersion, da das räumliche Eintauchen hauptsächlich eine sensorisch-motorische Neuanpassung hervorruft, während das emotionale Eintauchen eine kognitive Neuanpassung beinhaltet. Dadurch führt emotionale Immersion zu einem höheren Niveau der neuropsychologischen Aktivierung [7].

2.1.1 Einsatzszenario und Setup

Im Folgenden wird ein Einsatzszenario als Beispiel für die Nutzung des orbis novem System dargestellt. Dieses dient zudem als Erläuterung der Ausgangslage. Das Beispiel ist hierbei die Durchführung einer geführten Immersion für den Rezipienten durch einen Betreuer.
Geführt bezieht sich hierbei auf die gezielte Einweisung und Betreuung durch einen Akteur Betreuer. Damit ist nicht gemeint, dass der Spielverlauf für den Rezipienten geführt, das heißt kontrolliert wäre. Das Explorieren der VR Umgebung bleibt vom Betreuer unbeeinflusst, ausgenommen bei der gewünschten Beendigung oder Pausieren der Nutzung durch den Rezipienten.
Zudem wird an dieser Stelle der Begriff der erfolgreichen Immersion für diese Arbeit eingeführt und definiert: Innerhalb des orbis novem Systems handelt es sich um eine *erfolgreiche Immersion*, wenn alle später genannten Teilsysteme involviert sind und zum Immersionserlebnis des Rezipienten beitragen.

Auf den geplanten Setup, der die Immersion ermöglichen soll wird nun kurz eigegangen. Grundlegend ausgedrückt wird die Immersion erreicht, indem einem Rezipienten die Möglichkeit gegeben wird, mittels verschiedener Interaktionsmedien mit einer VR-Umgebung zu interagieren. Primäres Interaktionsmedium ist dabei eine Datenbrille, die optisch eine VR-Umgebung abspielt und diese gemäß der Kopfstellung des Rezipienten anpasst. Dazu kommt ein Laufsystem, auf welches der Rezipient aufsteigt. In Kombination mit der Datenbrille ermöglicht das Laufsystems (in Form einer Lokomotionsplattform) aufgrund von Bewegungssensoren die Illusion von Fortbewegung, obwohl sich der Rezipient physisch nicht vom Ausgangspunkt weg bewegt. Zuletzt wird Interaktion mit Elementen innerhalb der VR-Umgebung ermöglicht, indem diese vom Rezipienten per Blicksteuerung gelenkt werden können.
Zusammenfassend ausgedrückt hat der Rezipient, innerhalb einer geführten, erfolgreichen Immersion die Möglichkeit, eine maßstabsgetreue, digitale Repräsentation eines Hochschul-Gebäudes (der Name Neue Welt 9 spielt dabei auf das Fakultäts-Gebäude Nr. 9 an) zu explorieren. Er oder sie kann sich in der VR-Umgebung dank eines Laufsystems in Form einer Lokomotionsplattform realitätsnah bewegen und vorhandene Elemente innerhalb der VR-Umgebung mittels Blick-Steuerung beeinflussten. Die Datenbrille erkennt dabei mittels der später vorgestellten Blicksteuerungskomponente iVR HUD Befehle wie Blickrichtung und

Blinzeln des Rezipienten und reagiert unmittelbar auf diese [1].

2.2 Durchführung einer geführten Immersion

In diesem Kapitel wird zunächst der Ablauf einer geführten Immersion dargestellt. Dieser soll die nötige Interaktion mit den Subsystemen, sowie zwischen den Systemakteuren Rezipient und Betreuer festzuhalten.
Die Reihenfolge der Aktivitäten folgt dabei der folgenden Logik: Zunächst legt der Rezipient die Schuhüberzüge an und steigt Rezipient mit freien Händen aber mit Hilfe des Betreuers in den Virtualizer. und ohne Datenbrille in den Virtualizer. Sobald der Sitz des Geschirrs sitzt, nimmt er die Datenbrille entgegen und justiert diese über den Augen. Nun hat er alles erledigt, für das freie Hände nötig sind, weshalb er dann vom Betreuer die Controller mittels Handschlaufen angelegt bekommt. Zudem ist es die Verantwortung des Betreuers, sicherzustellen, dass die Kabelführung von der Datenbrille zum Steuerungscomputer den nun blinden Rezipienten nicht beeinträchtigt. Sobald die Exploration der Immersion beendet ist, wird in der umgekehrten Reihenfolge vorgegangen. Das Verständnis für den Ablauf einer geführten Immersion ist zudem Grundlage für das später skizzierte Test-Szenario für einen Systemtest. Ebenso ist es notwendig für die zukünftige Nutzung des Systems auf Messen oder als Show Case Objekt.

Es wird ein Aktivitätsdiagramm in der folgenden Abbildung 2 genutzt, welches die notwendigen Aktivitäten aufzeigt. Die Abbildung 2 kann zudem als Anleitung für die Nutzung auf Messen, Tag der offenen Tür und ähnlichem dienen.

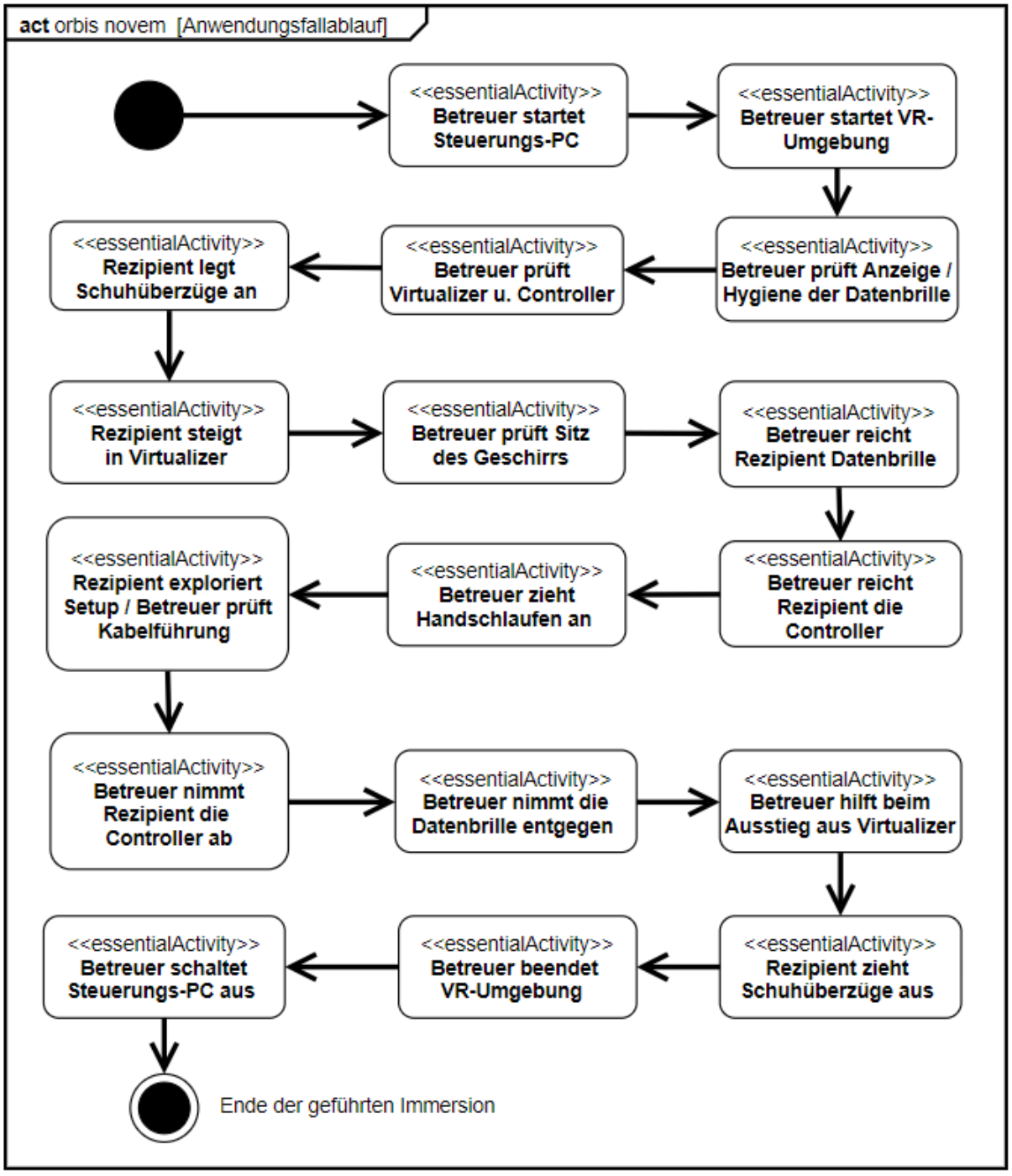

Abbildung 2: Aktivitäten für eine geführte Immersion

2.3 Teilsysteme

Es folgt eine kurze Vorstellung aller involvierten Teilsysteme, die innerhalb von orbis novem kombiniert werden. Damit bietet dieser Abschnitt eine Übersicht der Teilsysteme, mit einem Fokus auf deren primäre Funktion innerhalb der angestrebten Immersion.

2.3.1 HTC Vive Headset

Bei diesem Teilsystem handelt es sich um eine VR Datenbrille, namentlich dem Modell Vive des Herstellers High Tech Computer Corporation (HTC). Diese wird von einem Rezipienten aufgesetzt und ermöglicht es optisch in einen 3D-Raum einzutauchen. Mittels Tracking-Technologie kann der Rezipient sich zudem in der VR-Umgebung frei bewegen. Die Vive erkennt dabei die Bewegungen, Haltung und generelle Blickrichtung des Kopfes. Diese Daten haben einen direkten Einfluss auf die Szene der VR-Umgebung, die über das Display in der Datenbrille angezeigt wird. Die HTC Vive bietet zudem einen Satz handgehaltener Controller für die Interaktion zwischen Rezipienten und VR-Umgebung.

2.3.2 Handgehaltene Controller

Dieses Teilsystem repräsentiert die handgehaltenen und bewegungsgesteuerten Controller, mittels denen mit Elementen innerhalb einer VR-Umgebung interagiert werden kann. In der Regel trägt der Rezipient zwei Controller. Diese sind für die sichere Nutzung mit Hand-Schlaufen ausgestattet. Zudem werden deren Positionen und die wichtigsten Tasten standardmäßig auf dem Display der Datenbrille angezeigt, um dem Rezipienten die Steuerung zu erleichtern. Die Controller können beispielsweise mittels Vibrationen und Haptik-Simulationen zum Immersions-Erlebnis beitragen.

2.3.3 VR-Umgebung Neue Welt 9

Hinter diesem Teilsystem steckt eine, ebenfalls im Masterprojekt entwickelte VR–Umgebung, die auf der Computerspiel-Engine Unity basiert. Es stellt für den Nutzer einer Datenbrille eine maßstabsgetreue, digitale 3D-Repräsentation des gesamten Hochschulgebäudes Nummer 9 zur freien Exploration dar. Der Name ist eine Anspielung darauf. Die Neue Welt 9 ist eine der detailliertesten und umfangreichsten VR-Umgebungen, die den Studierenden zur Verfügung steht, da sie das primäres Artefakt eines eigenen langjährig laufenden Wahlfachs an der Fakultät Informatik darstellt. Ebenso ist die Neue Welt 9 sehr realitätsnah, was möglicherweise die Wahrscheinlichkeit einer vollständigen Immersion erhöht.

2.3.4 Blicksteuerungskomponente iVR-HUD nach [1]

Bei diesem Teilsystem handelt es sich um ein interaktives VR Head-up Display (HUD) wie es in Computerspielen geläufig ist. Der Name des Projektes bezieht sich auf das Ziel des Projektes, ein interaktives Virtual Reality HUD. Dabei steht HUD für Head-up Display. Head-up Displays dürften besonders Gamern bekannt sein, werden im Folgenden jedoch noch kurz erklärt:

Video- und Computerspiele nutzen ein HUD als Anzeigebereich am Bildschirmrand um den Spieler kontinuierlich über seine Spieldaten zu informieren. Dazu gehören die, für den Charakter oft lebenswichtigen Statistiken wie den aktuellen Gesundheitszustand, Attribute, Rüstungslevel, Munitionsübersicht oder die aktuelle Position auf der Spielkarte. Diese Anzeigen des HUD's sollten jedoch nicht das Gameplay negativ beeinflussen, zum Beispiel indem strategisch wichtige Elemente verdeckt werden. Ein gutes HUD ist im laufenden Gameplay für den Spieler praktisch unbemerkbar, ist aber dennoch mit einem Blick zugänglich, sobald es benötigt wird. Ein Beispiel wäre, dass der Spieler sich leicht auf einen Kampf im Gameplay konzentrieren kann, dennoch nebenbei die Gesundheit seines Charakters oder andere wichtige Statistiken im Blick behalten kann. Für viele Spieleentwickler ist das Design eines guten HUDs eine lohnende Herausforderung.

Nachdem erklärt wurde, was ein HUD ist, wird nun genauer auf die Komponente iVR HUD eingegangen. Ebenfalls im Rahmen des Masterprojekts im Studiengang Human-Centered Computing (HUC) an der Hochschule Reutlingen soll ein interaktives Virtual Reality Head-up Display (iVR-HUD) erzeugt werden. Genau genommen handelt es sich dabei um ein sogenanntes non-diegetic User Interface. Dieses wird entwickelt, um Interaktivität mithilfe von Blicksteuerung in ein Immersions-Erlebnis einzubinden. Damit sollen die Elemente im HUD durch Blick-Befehle des Nutzers bedient werden können. Indirektes Ziel des Projekts ist die Entwicklung eines Frameworks für ein interaktives VR-HUD mit Blicksteuerung, wobei der Begriff „Framework" hierbei zunächst recht frei interpretierbar ist und sich erst im Laufe der Entwicklung konkretisieren soll. Allgemein soll das Framework Entwicklern von VR-Anwendungen die Möglichkeit bieten, ein blickgesteuertes HUD möglichst leicht in ihre Anwendung integrieren und dieses an ihre speziellen Bedürfnisse anpassen zu können. Entsprechende Einschränkungen der Konfigurierbarkeit haben eine hohe Priorität, sind aber nicht erfolgsentscheidend. Wünschenswert wären zudem Schnittstellen für andere Eyetracking-Geräte, sofern zeitlich und technisch umsetzbar. Dabei beschränkt sich der Fokus durchgehend auf Anwendungsszenarien im VR-Lab für Demonstrationen, Produkterweiterungen und zukünftige Projekte. Aus der geleisteten Arbeit soll ein Framework abgeleitet werden, um die Nutzung des iVR-HUD zumindest für zukünftige Projekte im VR-Lab Reutlingen erleichtern zu können. Später kann mit den gewonnenen Erkenntnissen vielleicht sogar in den Fachbereich der Augmented Reality eingetaucht werden. Darunter wird eine zusammengesetzte Nutzungsansicht verstanden, die eine reale, vom Nutzer betrachtete Szene mit einer virtuellen Szene kombiniert. Die virtuelle Szene ist dabei vom Computer erzeugt und

enthält zusätzliche Informationen zur gleichzeitig betrachteten realen Szene [1].

2.3.5 Laufsystem Cyberith Virtualizer

Bei dem Laufsystem handelt es sich um das Modell Virtualizer der Firma Cyberith**.** Dieses Teilsystem ermöglicht dem Rezipienten die Illusion der vollen Bewegungsfreiheit in einer virtuellen Umgebung. Es handelt es sich konkret um eine Lokomotionsplattform mit integrierten Sensoren zur Erkennung der simulierten Bewegung in VR-Anwendungen. Der Bewegungssensor im Sockel reagiert auf Gewichtsverschiebungen und Sprungbewegungen des aktuell in das zugehörige Geschirr angeschnallten Rezipienten. Dank des festinstallierte Simulators können VR-Umgebung ausgiebig durchschritten werden, ohne sich tatsächlich physisch von der Stelle zu bewegen. Dazu werden reibungsmindernde Schuhüberzüge an den Füßen getragen, um über den Sockel zu gleiten.
Die Firma bietet mehrere Modell-Versionen des Laufsystems an, wobei die im VR Lab genutzte Version in der Abbildung 3 festgehalten ist.

Abbildung 3: Modell des Virtualizer im VR-Lab

Bei dieser Komponente handelte es sich zum Zeitpunkt des Projektbeginns um eine Neuanschaffung der Hochschule Reutlingen. Dies machte, im Vergleich zu den anderen Hardware-Komponenten, eine längere Einarbeitungszeit erforderlich.

2.3.6 Steuerungscomputer

Der Steuerungscomputer dient als ‘Kommando-Zentrale‘ des orbis novem Systems, da dieses Teilsystem die Computer-Komponente repräsentiert, welche alle anderen Teilkomponenten

bündelt. An dieser Stelle laufen die anderen Teilsysteme zusammen, das heißt die Datenbrille greift auf die VR Umgebung zu, welche auf diesem Teilsystem gespeichert ist, der Virtualizer ist hier angeschlossen, etc. Der Steuerungscomputer muss in der Lage sein, die Technologien der anderen Teilsysteme zu unterstützen, beispielsweise in Bezug auf den verfügbareren Speicherplatz und die Rechenleistung. Der aktuell verwendete Steuerungscomputer wird in direkter Kooperation mit dem INF Service betrieben.

Nachdem nun ein Grundverständnis der involvierten Subsysteme geschaffen wurde, beschäftigt sich das folgende Kapitel mit der systemischen Betrachtung der Interaktion dieser innerhalb des orbis novem Systems.

3 Systemische Betrachtung

In diesem Kapitel werden die grundlegenden, das heißt eine Immersion unterstützenden Aspekte des orbis novem Systems erläutert, analysiert und modelliert. Dabei werden Prinzipien des System Engineering angewendet, um orbis novem als System in einem angemessenen Umfang zu betrachten. Zu diesem Zweck wird der Projektkontext illustriert, die wichtigsten Stakeholder und Anforderungen identifiziert, die Basisarchitektur, Anwendungsfälle, Systemkontext und Systemstruktur modelliert, sowie mögliche Testszenarien beleuchtet. Der Begriff Systems Engineering wird im Folgenden formal eingeführt.

3.1 Fragestellung innerhalb der systemischen Betrachtung

An dieser Stelle wird die Fragestellung der systemischen Betrachtung formuliert. Diese lässt sich grundlegend wie folgt zusammenfassen: Wie unterstützt orbis novem als System den jeweiligen Rezipienten dabei, eine vollständige Immersion zu erleben?
Um diese Fragestellung zu beantworten, wird nach Prinzipien des Systems Engineering (SE) vorgegangen. Um Systems Engineering einzuführen, ist es unerlässlich, zunächst den zugehörigen Begriff des Systems für diese Arbeit zu definiert.
Dazu dient die folgende Definition: Ein System ist ein von Menschen erstelltes Artefakt, bestehend aus Systembausteinen, die gemeinsam ein Ziel verfolgen, das von den Einzelelementen nicht erreicht werden kann. Ein Baustein kann aus Software, Hardware, Personen oder beliebigen anderen Einheiten bestehen [5].
Basierend auf dieser Definition kann nun die zugehörige Engineering-Disziplin erläutert werden:
Das System Engineering ist ein interdisziplinärer Ansatz und konzentriert sich auf die Definition und Dokumentation der Systemanforderungen in der frühen Entwicklungsphase, die Erarbeitung einer Systemarchitektur und die Überprüfung des Systems auf Einhaltung der gestellten Anforderungen unter Berücksichtigung des Gesamtproblems: Betrieb, Zeit, Test,

Erstellung, Kosten & Planung, Training & Wartung und Entsorgung. Dem Systems Engineering lassen sich dabei folgende Disziplinen oder genauer Aufgabenbereiche zuordnen: Projektmanagement, Risikomanagement, Requirements Engineering und Management, Systemarchitektur, Systemverifikation, Systemvalidierung und Systemintegration. Diese werden jeweils auf Systemebene betrachtet aber tauchen nicht in die Details einer Disziplin ein [5].
Um die genannte Fragestellung zu beantworten, wird orbis novem als System analysiert und beschrieben, sowie die involvierten Teilsysteme dargestellt. Die Vorgehensweise innerhalb der systemischen Modellierung wird im Folgenden vorgestellt.

3.2 Vorgehensweise

An dieser Stelle wird zunächst die gewählte Vorgehensweise aus dem Bereich des Systems Engineering erläutert, bevor mittels dieser das orbis novem System betrachtet wird.
Das generelle Vorgehen bei der systemischen Betrachtung orientiert sich dabei an dem Vorgehensmodell, beziehungsweise dem Prozess SYSMOD. Dieser Prozess, sowie die Grundlagen der verwendeten Modellierungssprache SysML sind dabei [5] entnommen. Bei dem SYSMOD Prozess handelt es sich um ein pragmatisches Vorgehensmodell aus dem Bereich des Modell-basierten Systems Engineering (MBSE). MBSE ist dabei die formalisierte Anwendung der Modellierung, um die Aktivitäten zu Systemanforderungen, Architektur, Analyse, Verifikation und Validierung von Beginn der konzeptuellen Architekturphase über die Entwicklung bis zu den späteren Phasen des Systemlebenszyklus zu unterstützen [5].
Innerhalb von SYSMOD wird die Modellierungssprache SysML™ genutzt, welche im Folgenden kurz erklärt wird: Die SysML (Systems Modeling Language) wurde im April 2006 veröffentlicht und im September 2007 offiziell von der OMG (Object Management Group) als offener Standard verabschiedet. Die SysML ist darauf ausgelegt, komplexe Systeme, welche Menschen und Einrichtungen, Hardware und Software, Hydraulik, Elektrik und Navigation umfassen, mittels einer gemeinsamen Designsprache entwickeln zu können [5].
Ein Merkmal der SysML erweist sich für die Nutzung als Modellierungssprache als besonders vorteilhaft:
Zum einen basiert die Sprache auf einer Erweiterung der OMG-eigenen Unified Modellig Language (UML) und ist damit zum anderen grafisch aufgebaut. Softwareentwickler, die mit UML vertraut sind, können daher leicht auf SysML umsteigen und für die Analyse verschiedener Systembausteine nutzen [5].
Gemäß des SYSMOD Vorgehensmodells werden für die Modellierung des orbis novem Systems die folgenden Phasen durchlaufen und damit in dieser Ausarbeitung dokumentiert:

1. Beschreibung des Projektkontext
2. Bestimmen der Anforderungen
3. Modellieren des Systemkontextes
4. Definieren von Use Cases und Prozessen
5. Modellierung der Systemstruktur
6. Fachwissen sammeln und formal beschreiben

Innerhalb der Phasen werden einzelne Schritte beschrieben, die jeweils Inputs benötigen und Outputs generieren. Diese Schritte können laut [5] als ein Werkzeugkasten betrachtet werden, aus dem sich passend zum gegenwärtigen Projekt bedient werden kann. Es ist damit kein starres Kochrezept, das nur in der beschriebenen Reihenfolge erfolgreich angewendet werden kann und es sind auch andere Schrittfolgen denkbar, insbesondere, wenn Modellierungsergebnisse entfallen, da sie nicht benötigt werden oder bereits vorliegen und dadurch im Rahmen des Vorgehens nicht erst erzeugt werden müssen [5].

In Bezug auf die SysML kann sich die Frage ergeben, warum nicht direkt die zugrundeliegende UML als Modellierungssprache verwendet wurde. Darauf wird nun eigegangen:
Die UML ist zwar sehr mächtig, weist aber hinsichtlich des Systems Engineerings entscheidende Defizite wie die fehlende Anforderungsmodellierung auf. Ein weiterer Grund, der die Notwendigkeit von SysML begründet, ist die Tatsache, dass UML zum Teil softwarelastig und stark von der Objektorientierung geprägt ist, im Systems Engineering aber disziplinenneutral modelliert wird [5].
Die Nutzung von SysML brachte bei der Durchführung einen unvorhergesehenen Vorteil: Die aufgrund dieser Wahl notwendige, intensive Auseinandersetzung mit der weniger vertrauten, Modellierungssprache führte zu einer intensiveren Auseinandersetzung mit vorher ignorierten Aspekten des Systems. Das ergibt sich dadurch, dass die SysML sehr spezifische, klar zu einender abgegrenzte Möglichkeiten bietet um ein System zu modellieren.
Im Laufe der Nutzung wurde es notwendig, sehr genau festzulegen, als was sich ein Systemmerkmal klassifizieren und damit modellieren lässt. Dies führte mehrfach dazu, dass das orbis novem System, welches dem Autor zuvor bereits bekannt war, in einer komplett anderen Perspektive gesehen wurde. Die Betrachtung und Modellierung hat somit zu einem wesentlich tieferen Verständnis des Systems geführt, welches sich in der geleisteten Dokumentation wiederspiegelt. Dennoch wurden das Gesamtsystem sowie die Subsysteme nur soweit modelliert, dass ein tieferes Verständnis für die zentralen Aspekte erlangt werden kann, welche eine erfolgreiche Immersion unterstützen. In dieser ersten Iteration des iterativen SYSMOD-Prozess wurde darauf abgezielt, ein allgemeines Verständnis dieser unterstützen Systemaspekte zu erhalten und zu dokumentieren. In weiteren Iterationen können die erstellten Modelle und Diagramme weiter verfeinert und mittels Technologiewissen detailliert werden.

3.3 Anwendungsdomäne

An dieser Stelle wird die Anwendungsdomäne beschrieben. Das Themengebiet VR als Anwendungsdomäne des orbis novem System ist streng von dem recht ähnlichen Themengebiet der Augmented Reality (AR) zu unterscheiden:
Während VR das Sichtfeld eines Nutzers vollständig ersetzt, ergänzt AR dieses lediglich. Die visuelle Immersion, die markanteste Charakteristik von VR, entfällt bei dieser Technologie, da hierbei Bilder nur in einem begrenzten Bereich vor das Auge projiziert werden, beispielsweise auf einem Smartphone oder Tablet-Bildschirm [1].
Für die Anwendungsdomäne VR und AR sind der Begriff der Immersion und ferner dessen Unterarten hilfreich für das Verständnis. Im Zusammenhang mit dem orbis novem System lässt sich auch das Stichwort Edutainment nennen, einem Kofferwort aus education (Bildung) und entertainment (Unterhaltung). Der Begriff bezeichnet Konzepte der elektronischen Wissensvermittlung, mit welchen angestrebt wird, die jeweiligen Inhalte spielerisch und gleichzeitig unterhaltsam zu vermitteln. Dazu können entsprechende Fernsehprogramme, Computer-/Videospiele oder Multimedia-Softwaresysteme gezählt werden.

3.4 Projektkontext

In diesem Unterkapitel werden Inhalte aus der SYSMOD Phase *Beschreibung des Projektes und Projektkontextes* beschrieben. Zu diesem Zweck werden die Systemidee, die Systemziele, die Ausgangslage und das Projektumfeld erfasst.
Die grundlegende Systemidee und Systemziele, welche nach [5] in kurzer und lesenswerter Form beschrieben werden sollten, sind wie folgt formuliert:

Das orbis novem System soll es einem Rezipienten ermöglichen, eine vollständige Immersion zu erleben.

Das Ziel, das sich aus der gegebenen Systemidee ergibt, also der angestrebte Zustand in der Zukunft, der durch das System erreicht werden soll [5], lässt sich in primäres und sekundäres Ziel einteilen:

Das primäre Ziel ist es, eine vollständige Immersion für einen Rezipienten anzubieten. Das sekundäre Ziel ist die Kombination einer Reihe an studentischen VR Labor Projekten, sowie vom VR Labor zur Verfügung gestellter Hardware.

Die Abbildung 4 ‘Ziele des Gesamtsystems‘ zeigt die wichtigsten Ziele als Anforderungsdiagramm mit der <<objective>> Notation nach der SysML.

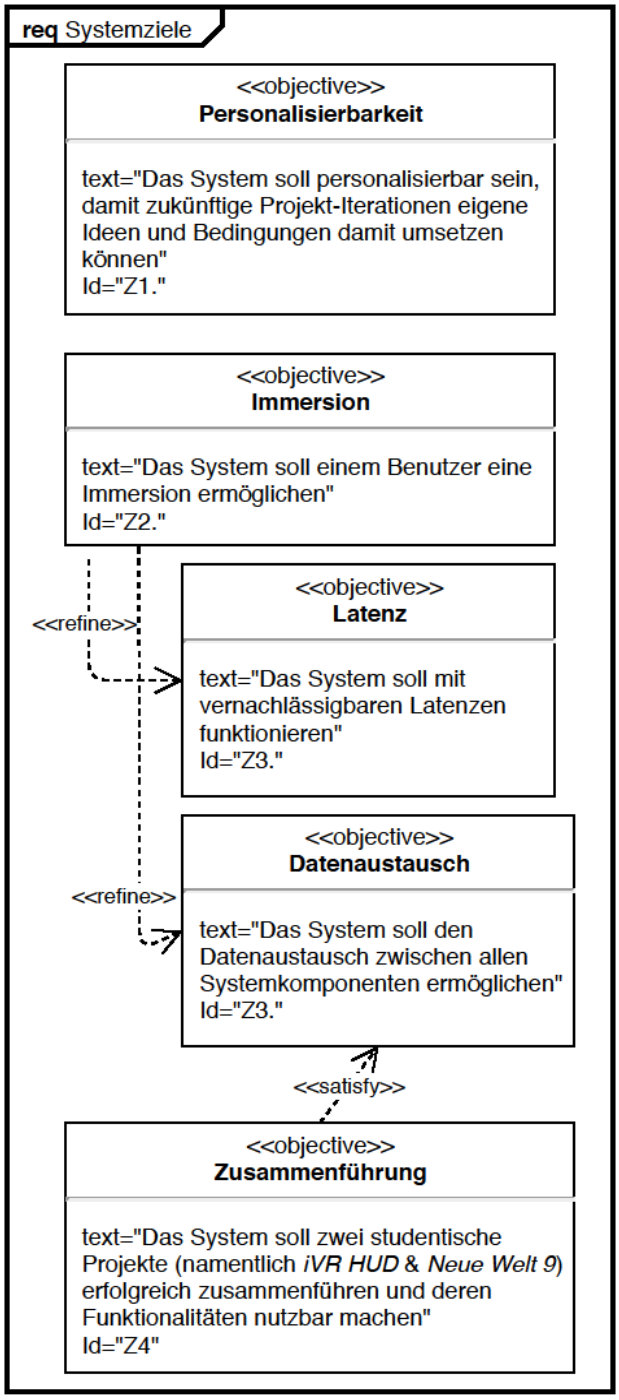

Abbildung 4: Ziele des Gesamtsystems

Eine Immersion kann nur gelingen, wenn alle Teilsysteme fehlerfrei ineinander greifen. Bereits kleine Störungen im Ablauf, Ausfall eines der Teilsysteme oder eine zu hohe Latenz können die Illusion aufheben und damit den Rezipienten irritieren. Um das primäre Ziel der Immersion zu erreichen muss das System mit einer vernachlässigbaren, das heißt den Spielfluss nicht unterbrechenden Latenz funktionieren. Zudem muss der Datenaustausch zwischen allen später identifizierten System-Komponenten gewährleistet sein [2].

Das primäre Ziel Immersion (ID: Z2) steht daher in einer <refine>> Beziehung zu den Zielen Latenz (ID: Z3) und Datenaustausch (ID: Z3), das heißt ersteres Ziel wird durch die zuletzt genannten Ziele präzisiert. Zudem wird das Ziel der Personalisierbarkeit (ID: Z1) aufgenommen. Das System soll personalisierbar sein, also nach eigenen Vorstellungen erweiterbar und modifizierbar. Z1 ergibt sich aus dem Projektumfeld des VR Labors: Die Teilnehmer künftiger Iterationen des Wahlfachs sollen in der Lage sein, das orbis novem System so zu erweitern, dass sie eigene Bedingungen und Idee damit umsetzten können. Das Ziel Zusammenführung (ID: Z4) steht in einer Erfüllungsbeziehung zum Ziel Datenaustausch. Z4 meint dabei die erfolgreiche Zusammenführung der beiden studentischen Projekte iVR HUD und Neue Welt 9. Diese Teilsysteme werden im Kapitel 2.5.1. genauer vorgestellt. Beide Teilsysteme müssen ausfallsicher zusammenarbeiten und deren jeweilige Funktionalitäten dem Rezipienten durchwegs zu Verfügung stehen.

3.5 Anforderungen

Nachdem zuvor der Projektkontext illustriert wurde, wird an dieser Stelle auf Inhalte der zweiten Phase des SYSMOD Prozesses, *Bestimmen der Anforderungen*, eingegangen.

Anforderungen sind die elementaren Grundlagen für die spätere Umsetzung des modellierten Systems, da sie bestimmen was das System leisten muss [5]. Die jeweiligen Anforderungen für das System ergeben sich aus der genannten Systemidee und Systemzielen, sowie der nun eigeführten Basisarchitektur und Stakeholdern.

3.5.1 Basisarchitektur

An dieser Stelle wird die Basisarchitektur erläutert, welche in der Abbildung 5 ‘Basisarchitektur‘ dargestellt ist.

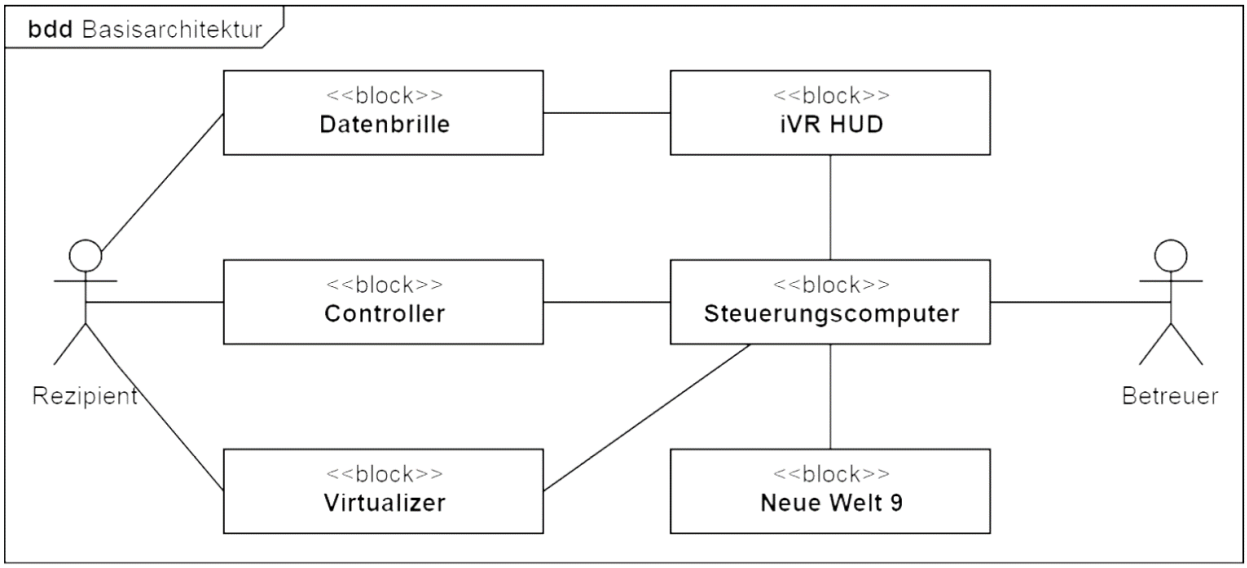

Abbildung 5: Basisarchitektur

Laut [5] wird die Basisarchitektur mit Blockdiagrammen, Skizzen und textuellen Erläuterungen beschrieben und wird dann genutzt um die Stakeholder zu identifizieren. Da das orbis novem System explizit darauf abzielt, mehrere bereits vorhandene Systeme zu kombinieren wurden

- *Welche technischen Komponenten sind vorgegeben?* und
- *Welche Architekturentscheidungen sind vorgegeben*?

als Leitfragen für die Modellierung der Basisarchitektur herangezogen [5].

Die Basisarchitektur ist dabei primär beeinflusst durch die Komponenten, welche kombiniert werden sollen. Dazu gehören auf der einen Seite die genannten VR Labor-Projekte Neue Welt 9 und das iVR HUD. Auf der anderen Seite kommen die zur Verfügung gestellten Hardware-Komponenten des Laufbandes und der Datenbrille hinzu.
Ebenfalls zur Verfügung gestellt wird ein Bedienrechner, der die zuvor genannten Komponenten bündelt, eine GUI zum System enthält und Zugriff durch die Stakeholder erlaubt. In der nachfolgenden Modellierung wird dieser als Steuerungscomputer bezeichnet. Die Basisarchitektur enthält zudem zwei Akteure, mit der Rolle des bereits beschriebenen Betreuers und Rezipienten. Diese gehören auf dieser Ebene der Modellierung mit zum System und deren explizite Abbildung hat die folgenden Gründe:

- Erstens erbringt der Akteur Betreuer Funktionalitäten, welche zur Immersion beitragen und ist damit für die Zielsetzung des Systems relevant. Beide Akteure leisten zudem Reaktionen auf den jeweiligen Systemzustand.
- Zweitens sind mehrerer Schritte innerhalb der geführten Immersion gefährlich, wenn sie nicht betreut werden.

Dazu gehört unter anderem das Starten und Beenden der Übertragung auf die Datenbrille. Während diese getragen wird ist der Rezipient für alle anderen Inhalte blind. Damit ist das Greifen und Festmachen der Controller an beiden Händen, das Ein- und Aussteigen auf die Laufbandfläche, sowie das Festschnallen das Geschirrs des Laufbandes (welches vergleichbar mit einem Klettergurt-Geschirr ist) alleine und mit aufgesetzter Datenbrille nicht ratsam. Hinzukommt ein Problem mit der aktuellen Kabelführung, welche den blinden Rezipienten beeinträchtigen kann. Aus all diesen Gründen ist es absolut notwendig, dass eine geführte Immersion, das heißt mit mindestens einem Betreuer durchgeführt wird.

Es bleibt abschließend zu erwähnen, dass es im orbis novem System zu jedem Zeitpunkt nur einen einzigen Rezipienten geben kann, da nur eine Person gleichzeitig eine Datenbrille tragen und in das eine Laufband eingeschnürt sein kann.

3.5.2 Stakeholder

In diesem Unterkapitel wird auf Stakeholder eingegangen. Dabei wurde bei der Auswahl die Leitfragen *Wer hat Interesse am System? Was wäre, wenn wir diesen Stakeholder und seine Interessen nicht berücksichtigen? Wer wird das System verwenden? Wer ist betroffen, wenn das System ausfällt? Wer entsorgt das System?* nach [5] betrachtet.
Die folgenden fünf Stakeholder werden als die wichtigsten festgehalten:

- Der **Rezipient**, also der aktuelle Nutzer des Systems. Er oder sie möchte eine vollständige Immersion und damit indirekt Wissenszuwachs erleben.
- Der **Professor**, welcher zum aktuellen Zeitpunkt das Wahlfach VR Labor und dessen Teilnehmer betreut. Dieser Stakeholder möchte das System als Lehrobjekt für das VR Labor nutzen, sowie als allgemeinen Forschungsgegenstand für die Fakultät.
- Der **VR Labor Student**, der aktuell am Wahlfach teilnimmt. Dieser hat ein Interesse am System als Lernmöglichkeit und Basis oder Erweiterung für ein potenzielles eigenes Projekt.
- Die **Hochschule** die das VR Labor beinhaltet. Diese sei als Stakeholder genannt, da die Hochschule Räume, Strom und WLAN zur Verfügung stellt, sowie rechtlich haftbar ist und das System zuletzt entsorgt. Dieser Stakeholder hat Interesse am System als Lehrmöglichkeit für Studierende, sowie als Show Case Objekt, beispielsweise für Messen.
- Der **INF Service**, beziehungsweise dessen Mitarbeiter in der Fakultät Informatik. Dieser Stakeholder wird festgehalten, da er oder sie das System und dessen Equipment betreibt und wartet, teils in Kooperation mit den anderen Stakeholdern. Ebenso ist dieser Stakeholder am System in Bezug auf das Thema IT-Sicherheit interessiert.

Die Stakeholder können teilweise sowohl Rezipient als auch Betreiber für das System sein, auch wenn die dahinter stehende Person in der jeweiligen Rolle unterschiedliche Anforderungen an das System stellt.
Die Stakeholder werden in der Abbildung 6 in einer StakeholderMap notiert:

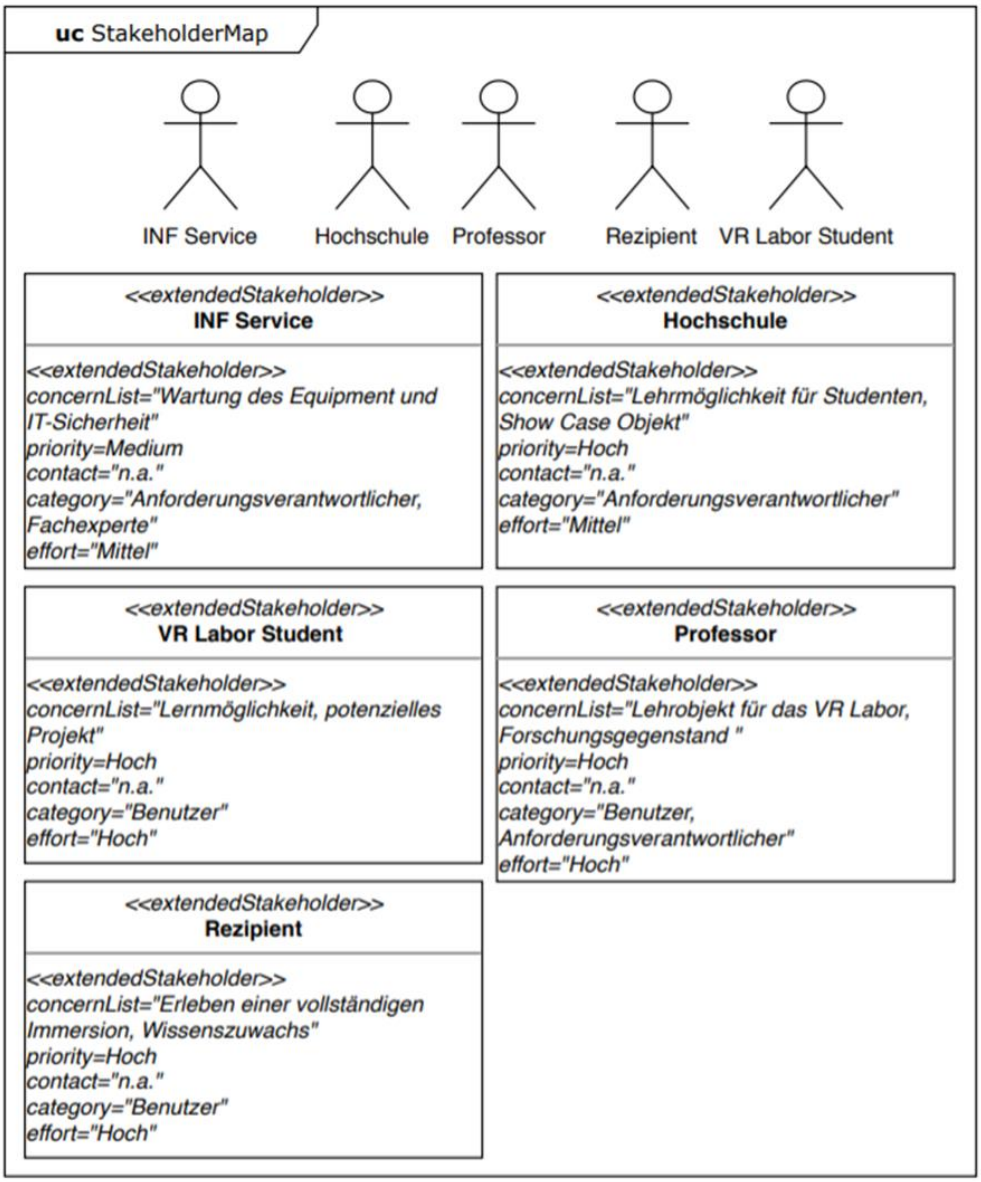

Abbildung 6: StakeholderMap

Dabei wird die SysML-Notation <<extendedStakeholder>> genutzt. Nach dem SysML-Profil handelt es sich dabei um erweiterte Stakeholder, da sie um eine Kategorie (genutzt wurde die Kategorien Fachexperte, Anforderungsverantwortlicher und Benutzer), einen Ansprechpartner, eine Priorität und eine Aufwandschätzung (hoch, mittel, niedrig) erweitert sind. Dabei ist zu erwähnen, dass ein Anforderungsverantwortlicher befugt ist, Anforderungen vorzugeben und auch zu bezahlen [5]. Im Folgenden werden nun die Anforderungen betrachtet.

3.5.3 Übersicht der Anforderungen

Basierend auf den vorherigen Überlegungen werden an dieser Stelle die relevantesten Anforderungen aufgeführt. Um die Auswahl der Anforderungen sinnvoll zu begrenzen, wird in

dieser Ausarbeitung das FURPS-Modell zur Orientierung genutzt: Dieses Modell dient der Kategorisierung von Anforderungen und wurde von Robert Grady entwickelt.
FURPS steht dabei für:

- Functionality – Funktionale Anforderungen
- Usability – Benutzbarkeitsanforderungen
- Reliability – Zuverlässigkeitsanforderungen
- Performance – Leistungsanforderungen
- Supportability – Testbarkeit, Konfigurierbarkeit, Installation, Service [5].

Insgesamt wurden die Anforderungen erfasst, welche in der folgenden Tabelle 1 mit einer ID und Kurzbeschreibung aufgelistet sind. Diese betrachten bewusst das System und dessen Verhalten als Gesamtheit und fokussieren sich nicht auf die Anforderungen auf einzelne Subsysteme. Spezielle Anforderungen an Subsysteme können von zukünftigen Iteration festgehalten werden, sollten aber dann mittels des aktuellen Technologiewissens formuliert sein.

ID	Kurzbeschreibung
REQ_F	Das System bietet alle notwendigen Interaktionen zwischen den Komponenten mit den angemessenen Funktionalitäten
F1	Stabile Verbindung zwischen Controllern und Steuerungscomputer
F2	Stabile Verbindung zwischen iVR HUD und Steuerungscomputer
F3	Stabile Verbindung zwischen Virtualizer und Steuerungscomputer
F4	Stabile Verbindung zwischen Datenbrille und iVR HUD
F5	Stabile Verbindung zwischen der Neue Welt 9-Umgebung und Steuerungscomputer
REQ_U	Das System verfügt über ein hohes Maß an Benutzerfreundlichkeit
U1	Das Betreiben und Nutzen der Hardware-Komponenten ist intuitiv
U2	Betrieb, Wartung und Nutzung darf keinen störenden Einfluss auf sonstige Raum-Nutzung haben
U3	Intuitive Nutzung des System über den Steuerungscomputer
REQ_R	Das System verfügt über eine hohe Zuverlässigkeit
R1	Das System arbeitet möglichst ausfallsicher
R2	Übertragung aller Daten mit konstant vernachlässigbarer Latenz
REQ_P	Das System verfügt über eine angemessene Leistungsfähigkeit
P1	Das System muss mit vernachlässigbaren Antwort-Zeiten arbeiten

P2	Das System arbeitet möglichst Ressourcen-sparend
REQ_S	Das System verfügt über eine angemessene Supportfähigkeit
S1	Die Anschaffung und Betrieb des Systems sind möglichst kostengünstig
S2	System muss automatisch ab Einschaltung alle Immersionsfunktionalitäten anbieten
S3	Automatische Kalibrierung bei Aufstieg auf den Virtualizer

Das jeweilige Präfix macht dabei deutlich zu welcher Kategorie nach FURPS die Anforderung gezählt wird. Alle Anforderungen und deren Enthält-Beziehungen nach SysML sind zudem in zwei Anforderungsdiagrammen in den Abbildungen 7 und 8 dargestellt.

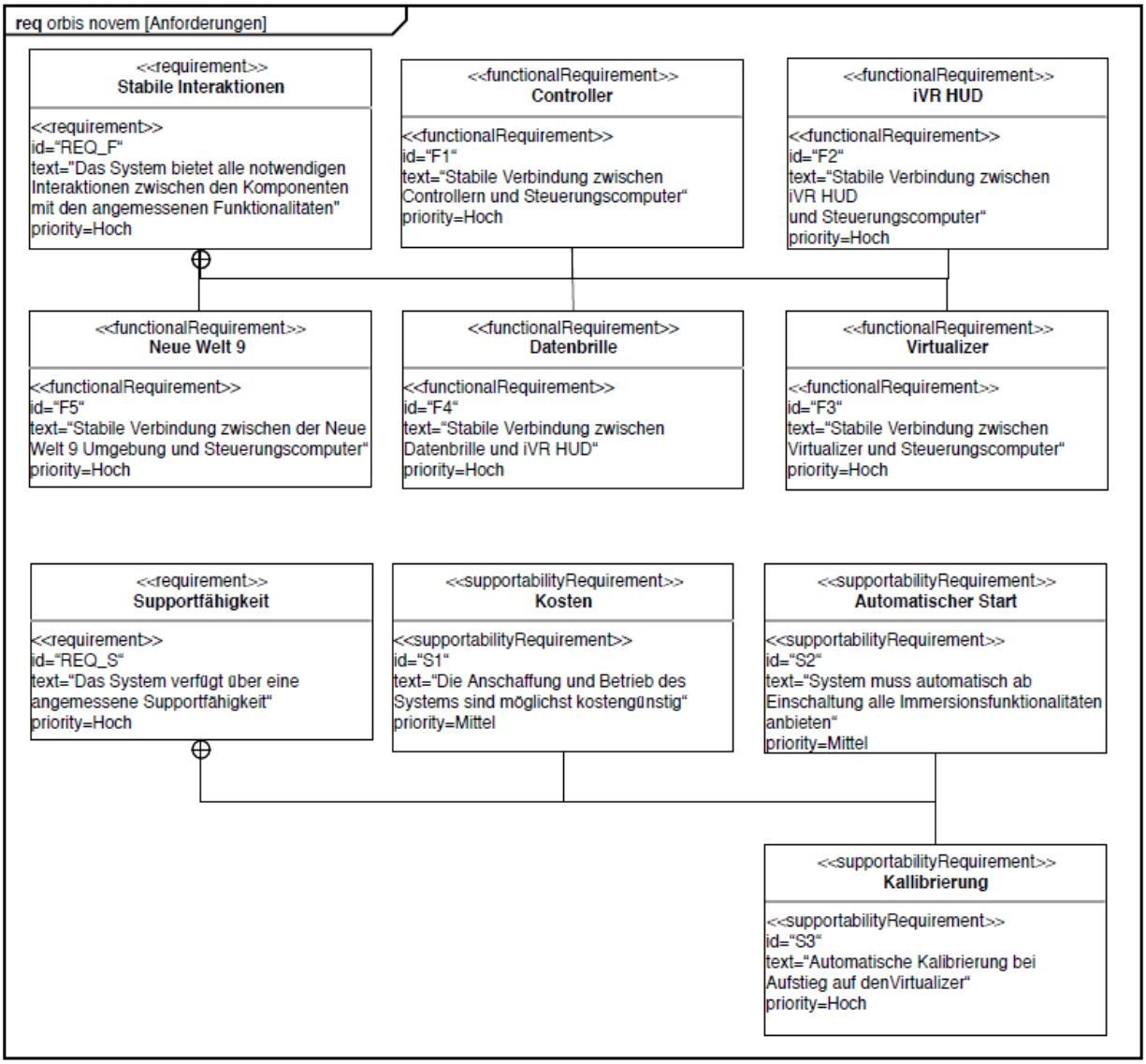

Abbildung 7: Funktionale Anforderungen

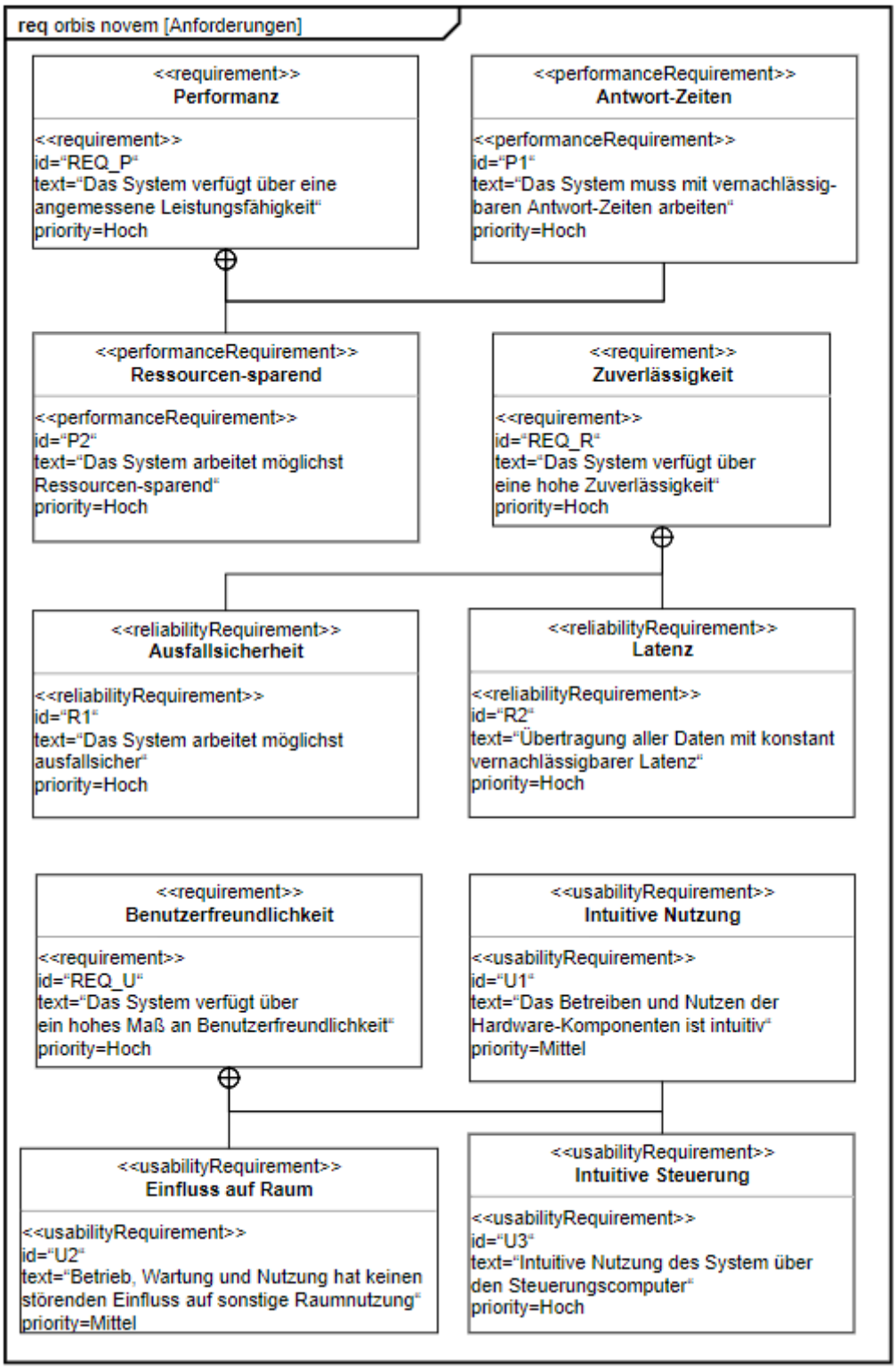

Abbildung 8: Sonstige Anforderungen

3.5.4 Systemkontext

In diesem Unterkapitel werden Inhalte aus der SYSMOD-Phase *Modellieren des* Systemkontextes beschrieben. Zu diesem Zweck wurde der Systemkontext von orbis novem untersucht und modelliert. Dieser wird mittels eines Systemkontextdiagramms gemäß der SysML dargestellt.
Das Systemkontextdiagramm stellt die unmittelbare Umgebung des Systems dar und gibt erste Auskünfte über die Kommunikation vom und zum System. Die außen stehenden Interaktionspartner sind die Systemakteure. Die Interaktion mit der Umgebung wird mit Objektflüssen und Ports beschrieben. Der Systemkontext sollte dabei in keinem Vorgehen fehlen. Das Systemkontextdiagramm zeigt zudem die Systemgrenze. Die explizite Abgrenzung Was gehört zu meinem System und was liegt außerhalb? ist dabei die wichtigste Information [5].
Die Systemgrenze ist relativ deutlich anhand dieser beiden Überlegungen zu ziehen:

1. *Welche Komponenten weisen Charakteristiken und Funktionalitäten auf, die einen Beitrag zur Immersionserfahrung des Rezipienten haben?*
2. *Welche dieser Komponenten liegen innerhalb der Kontrolle der jeweils am orbis novem Projekts beteiligten Entwickler?*

Das System und dessen Systemkontext ist vor allem durch die Bedingungen des VR Labors der Fakultät Informatik geprägt: Immerhin dient dessen Räumlichkeit als primärerer Platz für die Lagerung im inaktiven Zustand und für die Nutzung im aktiven Zustand.

Allerdings soll es stets möglich sein, das System zu transportieren. Dies ist besonders für Messen oder Demonstrationen gewünscht, wo das System und besonders das Laufsystem als Show Case Objekt genutzt werden. Der Systemkontext wird daher möglichst unabhängig von dem Raum, der vom VR Labor zur Verfügung gestellt wird, modelliert. Damit haben die Modelle auch Gültigkeit, wenn das System in anderen Räumen, aber unter vergleichbaren Bedingungen genutzt wird.

3.5.5 Systemakteure

Um den Systemkontext zu modellieren, werden zuerst die Systemakteure gesammelt. Systemakteure sind die direkten Interaktionspartner, für die Dienstleistungen und Schnittstellen entwickelt, beziehungsweise deren Einflüsse auf das System berücksichtigt werden müssen [5].
Das folgende Detail ist dabei zu verstehen: Ein Akteur ist kein konkretes System oder konkrete Person, sondern eine Rolle. Die Systemakteure beschrieben zudem die Grenze des Systems. Systemakteure sind die Interaktionspartner des Systems, also Elemente außerhalb des Systems. Ein Akteur ist kein konkretes System oder konkrete Person, sondern eine Rolle. Die

Systemakteure beschrieben die Grenze des Systems. Hierbei ist zu beachten, dass im SysML-Profil das Modellelement Akteur unterschieden wird in u.a. Benutzer, Fremdsystem, mechanisches System, Umgebungseinfluss, Aktuator und Sensor [5].
Diese Kategorien werden durch unterschiedliche Symbole des Akteurs dargestellt und im Folgenden kurz erklärt, sowie auf das System angewendet. Alle gesammelten Systemakteure lassen sind dem Systemkontextdiagramm in der Abbildung 9 ‘Systemkontext‘ entnehmen:

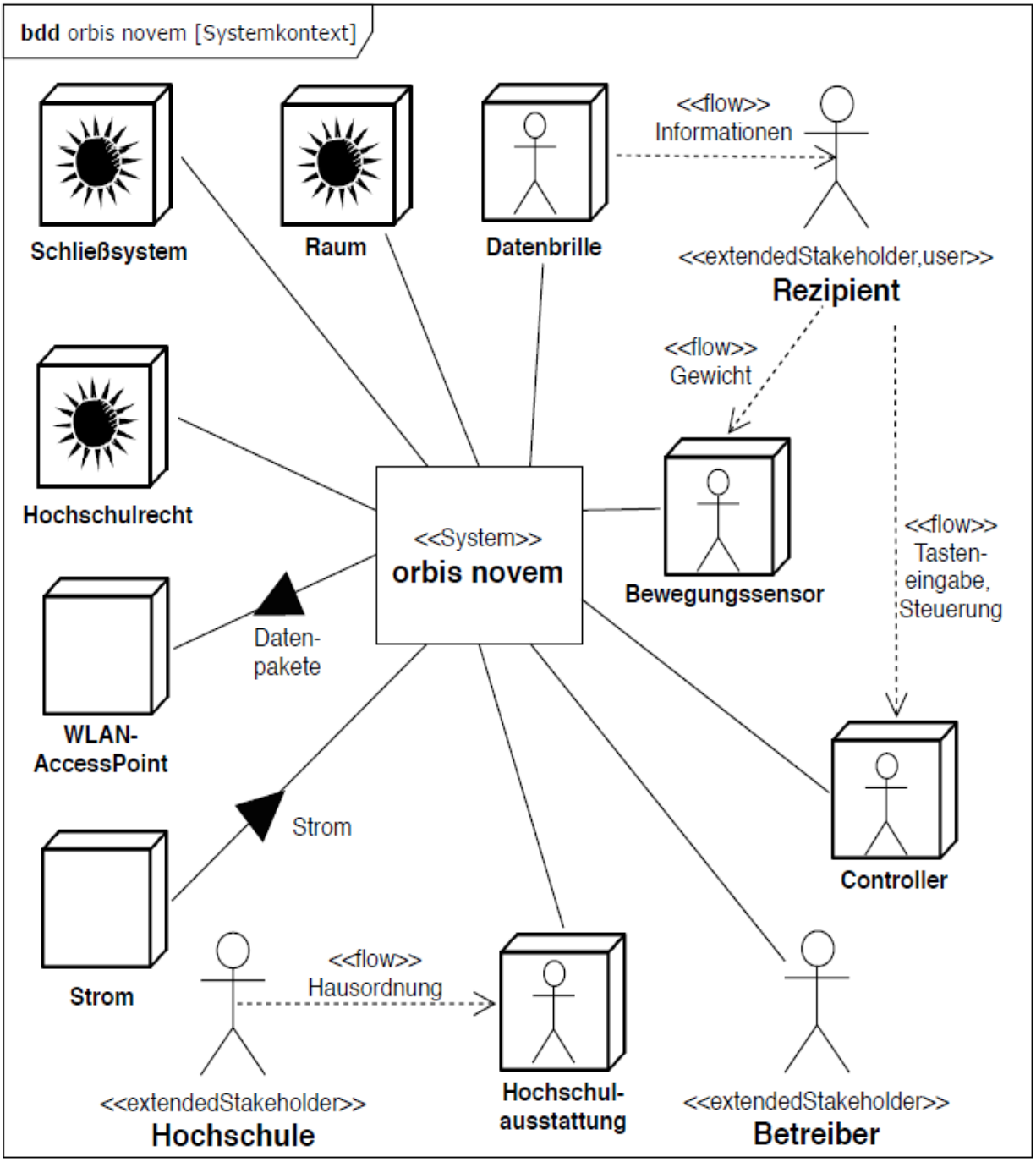

Abbildung 9: Systemkontext

Zunächst ist der Betreiber notiert, also die Rolle, die direkt mit dem System interagiert. Der Betreiber kann der Betreuer während der geführten Immersion sein, ein Professor, welcher das System für eine Demonstration erweitert oder ein INF-Service-Mitarbeiter, welcher ein Update auf einer der Komponenten aufspielt. Die Rolle Betreiber ist daher als Benutzer nach der SysML notiert.
Der Benutzer ist ein menschlicher Akteur und wird mit dem Standardsymbol für Akteure dargestellt. Ein Benutzer interagiert direkt mit dem System und kann und sollte direkt nach seinen Anforderungen an das System gefragt werden [5].

Zudem enthält die Abbildung 9 vier Benutzersysteme, namentlich Datenbrille, Bewegungssensor, Controller und die Hochschulausstattung.
Das Benutzersystem ist nach SysML eine Art Fremdsystem, das einem Benutzer als Medium dient, um mit dem System zu interagieren. Typische Benutzersysteme sind Tastatur, Anzeige oder ein Instrumentenbrett [5].
Es folgt eine kurze Beschreibung der vier erwähnten Benutzersysteme, und deren Interaktionspartner:

- Die Datenbrille, über welche kontinuierlich visuelle Informationen vom Steuerungscomputer zum Rezipient gesendet werden.
- Der Bewegungssensor des Laufbands, über welche kontinuierlich Informationen in Form von Gewichtsverschiebung des Rezipienten an den Steuerungscomputer gesendet werden.
- Der oder die Controller, über welche kontinuierlich Informationen wie Tasteneingaben und Steuerung (durch die Handhaltung) vom Rezipienten an das System gesendet werden.
- Die Hochschulausstattung, welche Informationen über deren Nutzung in Form der Hausordnung vorgegeben werden. Darunter fallen Einflussfaktoren wie die Stromversorgung, Hygiene, Raumnutzung, Verschlüsselungsstandards, Zugangs-Regelungen und Budget-Vorgaben.

Alle genannten Komponenten können als Medium genutzt werden, um das orbis novem System im aktiven und inaktiven Zustand zu beeinflussen und sind daher als Benutzersysteme nach [5] modelliert.
Zudem sind zwei Fremdsysteme modelliert, die je nach Art und Verwendungsweise auf das System einwirken. Diese sind hierbei die WLAN-Infrastruktur, konkreter der jeweilige WLAN-AccessPoint und die Stromversorgung im Raum. Ein Fremdsystem ist in der SysML ein System, das direkt mit dem zu modellierenden System interagiert. In der Rolle eines Interaktionspartners wird das Fremdsystem nur als Blackbox betrachtet und als leerer Quader dargestellt [5].
Für die genannten Fremdsysteme wurde zudem der Objektfluss modelliert, in diesem Fall

Datenpakete und Strom. Der Objektfluss wird modelliert, indem für jeden Akteur (ausgenommen Umgebungseinflüsse und mechanische Systeme, deren Einfluss sich in nicht-funktionalen Anforderungen wiederspiegelt) die relevanten Objekte, die dieser zum System sendet beziehungsweise vom System erhält, notiert werden. Der Fokus liegt auf der Objektflussrichtung, welcher in der SysML mit einem Dreieck, welches die Richtung im weitesten Sinne anzeigt, dargestellt wird [5].
Zu den notierten Umgebungseinflüssen zählt in der Abbildung 9 das Schließsystem des Raums in dem das System aktuell aufgebaut ist. Zudem kommen als Umgebungseinfluss die charakteristischen Gegebenheiten des jeweils genutzten Raumes selbst und das Hochschulrecht hinzu. Der letztgenannte Umgebungseinfluss ist besonders durch die Haftungsfrage gegeben.
Diese Faktoren aus der Umgebung, welche das System beeinflussen ohne eine direkte Interaktion auszuführen werden in der SysML als Umgebungseinfluss modelliert. Dieser wird als ein Quader mit Sonnensymbol notiert [5].
Auf die Modellierung mittels mechanischer Systeme wurde nach einiger Abwägung verzichtet, es soll aber hier erwähnt werden: Ein mechanisches System ist ein spezielles Fremdsystem, das aus Sicht unseres Systems nur mechanische Aspekte besitzt, beispielsweise ist das System daran befestigt. Es wird wie ein Fremdsystem als Quader mit zusätzlichem Werkzeugsymbol notiert [5].

3.5.6 Subsysteme

Nachdem zuvor der Objektfluss vom und zum System genauer betrachtet wurde, wird nun genauer auf die Subsysteme eingegangen. Um dies angemessen tun zu können, wird im Folgenden das Konzept Systems of Systems eingeführt. Wie zuvor betrachtet, besteht ein System stets aus mehreren System-Bausteinen, welche interagieren oder sich in einen Zusammenhang stellen lassen. Ein Systembaustein ist ein atomarer Baustein, welcher aus der Sicht des Systems nicht weiter zerlegbar ist. Wird der Systembaustein im Rahmen einer Betrachtung weiter zerlegt, verliert dieser seinen atomaren Status und stellt in Folge ein eigenes System dar. Ist dies der Fall wird von einem System of Systems gesprochen. Dabei handelt es sich also konkret um ein System, welches aus Systembausteinen besteht, welche nicht atomar sind, da sie wiederum in eigene Systembausteine zerlegt werden. Darauf aufbauend findet sich auch der Begriff System of Systems of Systems [8].
Das Gesamtsystem orbis novem als System of Systems besteht auf dieser Ebene der Modellierung aus fünf Subsystemen, dargestellt in der Abbildung 10 ‘Übersicht der Subsysteme‘.

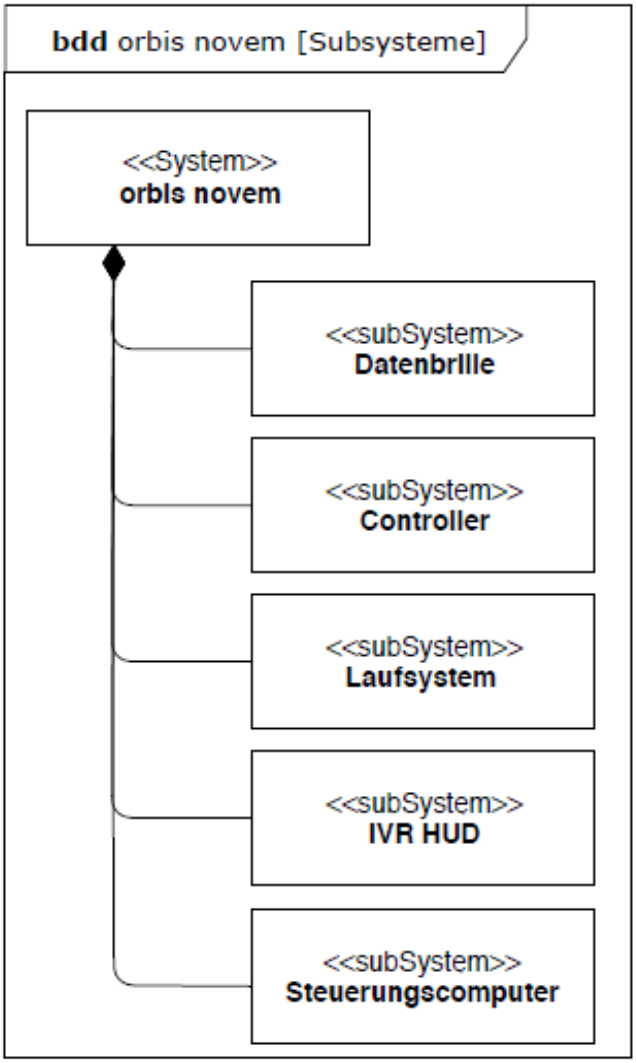

Abbildung 10: Übersicht der Subsysteme

Die dargestellten Subsysteme ergeben sich dabei unter anderem aus den zuvor beschriebenen Teilsystemen, die kombiniert werden sollen. Datenbrille und Controller könnten dabei als ein Subsystem angesehen werden, da Hardware für beide aktuell von HTC in einem Paket kommen.
Allerdings könnte auch eine andere Technologie oder Hersteller für die Controller verwendet werden, wie zum Beispiel der Leap Motion Controller. Auf die Erweiterung des Systems um Produkte der Firma Leap wird im Ausblick eingegangen. Außerdem erleichtert diese Unterteilung die weitere Modellierung, wie sich gleich zeigt.
Basierend auf dieser Modellierung werden Interaktionspunkte der Subsysteme modelliert. Diese Interaktionspunkte werden als sogenannte Ports dargestellt. Der Port in der SysML beschreibt einen Interaktionspunkt zwischen einem Systembaustein und der Umgebung. Der Port ist eine spezielle Eigenschaft des zugehörigen Systembausteins [5]. Die Abbildung 11 ‘Interaktionspunkte der Subsysteme mit Objektfluss‘ zeigt die jeweiligen Subsysteme und ihre Interaktionspunkte.

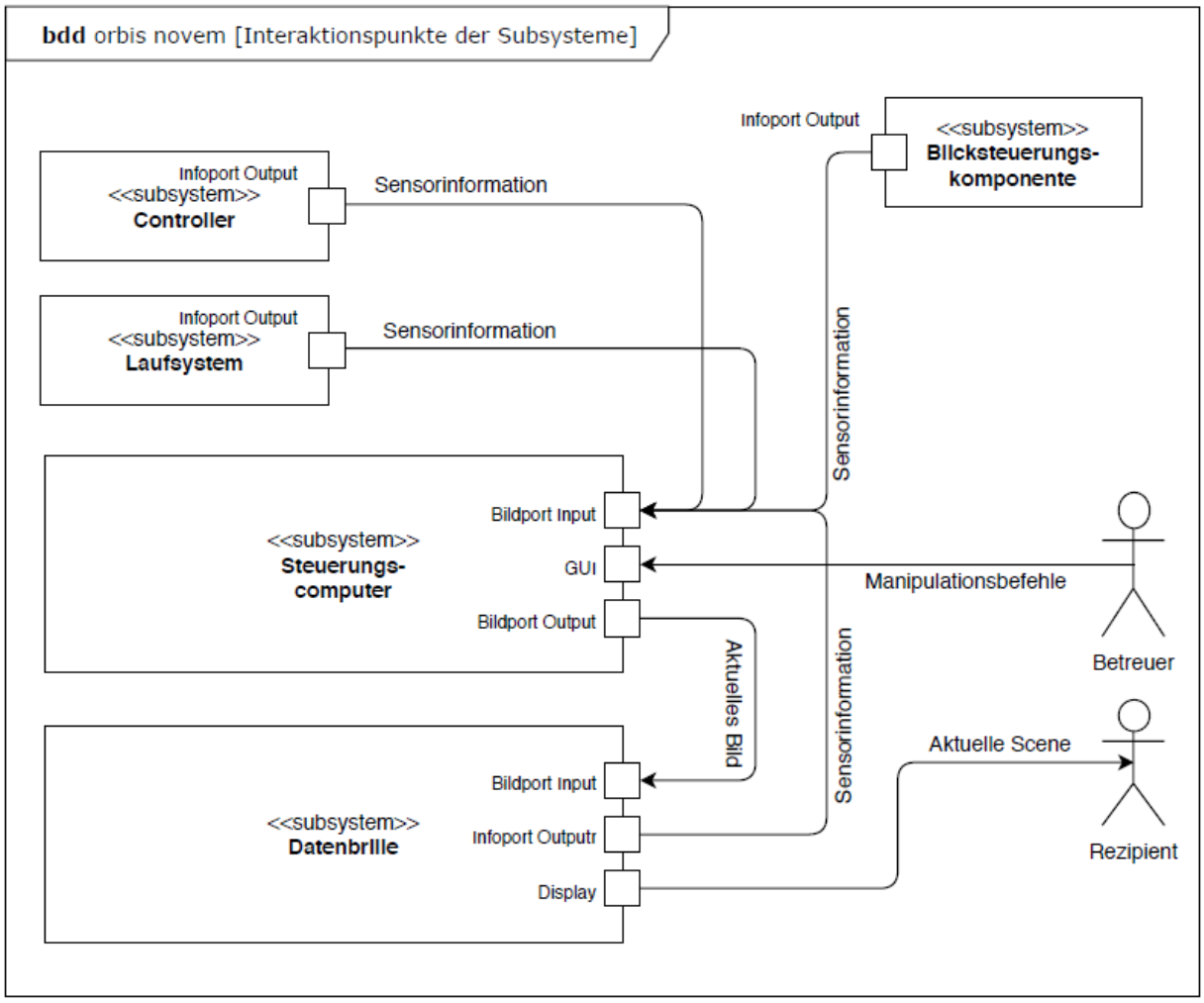

Abbildung 11: Interaktionspunkte der Subsysteme

Die beiden Interaktionspartner Rezipient und Betreuer sind an dieser Stell bewusst im Systemkontext, da beide aktiv den Systemzustand verändern können. Dies geschieht im Falle des Rezipienten wie folgt:

1. Über die Bewegung der Augen und bewusstes Blinzeln. Dieses Verhalten wird von der Blicksteuerungs-Komponente mittels Eyetracking-Technologie erfasst.
2. Über die Bewegung und Haltung des Kopfes. Dieses Verhalten wird mittels Sensorik in der Datenbrille erfasst.
3. Über die Verlagerung des Gewichts und gegebenenfalls Sprungbewegungen. Dieses Verhalten wird von der Sockelfläche des Laufsystems mittels Bewegungs-Sensorik erfasst.
4. Über die Bewegung der Controller in einer oder beiden Händen und dem Drücken deren Tasten. Dieses Verhalten wird von der Sensorik in den Controllern beziehungsweise den jeweiligen Tasten erfasst.

Basierend auf dieser Erläuterung wird an dieser Stelle genauer auf die Interaktionspunkte der Subsysteme Steuerungscomputer und Datenbrille nach Abbildung 11 eingegangen:

Alle erfassten Daten über die aktuellen Parameter des Rezipienten, wie zuvor beschrieben, werden an das Subsystem Steuerungscomputer übertragen. Dazu wird der Port Bildport Input genutzt. Der Steuerungscomputer errechnet anhand der über den Port eingehenden Daten kontinuierlich das aktuelle Bild, beziehungsweise die Szene. Diese ist abhängig von der genutzten VR-Umgebung, dem Bildausschnitt, aller aktuellen Parameter des Rezipienten und der internen Logik der virtuellen Realität (beispielsweise ob in der Szene gerade Tageslicht oder Nacht simuliert wird). Über den Port GUI werden Manipulationsbefehle vom Interaktionspartner Betreuer angenommen, das heißt über diesen Port hat der Betreuer explizit Zugriff zum orbis novem System. Über diesen Port würde im fertiggestellten System zum Beispiel das Starten, Beenden und Pausieren, sowie das selektive Abschalten einzelner Subsystems geschehen, falls dieses nicht vom Rezipient gewünscht sind oder ein Testszenario durchgeführt wird. Zudem ist der Bildport Output notiert. Über diesen Port wird das aktuelle Bild an den Bildport Input der Datenbrille gesendet. Insgesamt haben der Steuerungscomputer, sowie die Datenbrille drei Ports. Die Datenbrille erhält das aktuelle Bild über den Port Bildport Input und gibt über den Port Infoport Output die erfassten Daten an den Bildport Input des Steuerungscomputers. Ebenso ist der Port Display notiert, über den die aktuelle Szene dem Rezipienten angezeigt wird. Dieser Port ist essenziell für das Anbieten einer Immersion: Einzig über diesen werden dem Rezipienten die jeweiligen Ergebnisse seiner Verhaltensweisen, in Bezug auf die verschiedenen Interaktionsmittel, visualisiert.
Nachdem nun unter anderem die allgemeinen Anforderungen an das System und der Systemkontext erläutert wurden, werden diese Inhalte im Folgenden genutzt, um Use Cases abzuleiten [5].

3.6 Use Cases und Systemprozesse

In diesem Abschnitt wird auf Inhalte eingegangen, die sich in der Bearbeitung der vierten SYSMOD-Phase ergeben, namentlich *Definieren von Use Cases und Prozessen.*
Zunächst ist es hierbei nötig, relevante Use Cases (dt. Anwendungsfälle) für das System zu identifiziert. Diese Anwendungsfälle beschreiben die Dienstleistungen; welches das System erbringt, die von außen wahrgenommen und gefordert werden. Die jeweiligen Dienstleistungen, die ein System erfüllt, bestimmen seinen Sinn und Zweck [5].

In dieser Ausarbeitung wird der Begriff Anwendungsfall und Systemanwendungsfall synonym benutzt, soll aber hier formal eingeführt werden:
Ein Systemanwendungsfall hat immer mindestens einen Akteur, wird von genau einem fachlichen Auslöser gestartet und endet mit einem fachlichen Ergebnis [5].
In der Benennung der Anwendungsfälle wird die folgende Schablone als Orientierung verwendet: Aus Sicht des auslösenden Akteurs muss ein Satz der Form

[<Akteur> möchte <Anwendungsfall>]

sinnvoll klingen [5].
Für das System werden die folgenden, grundlegenden Systemanwendungsfälle modelliert, sowie in der Abbildung 12 dargestellt:

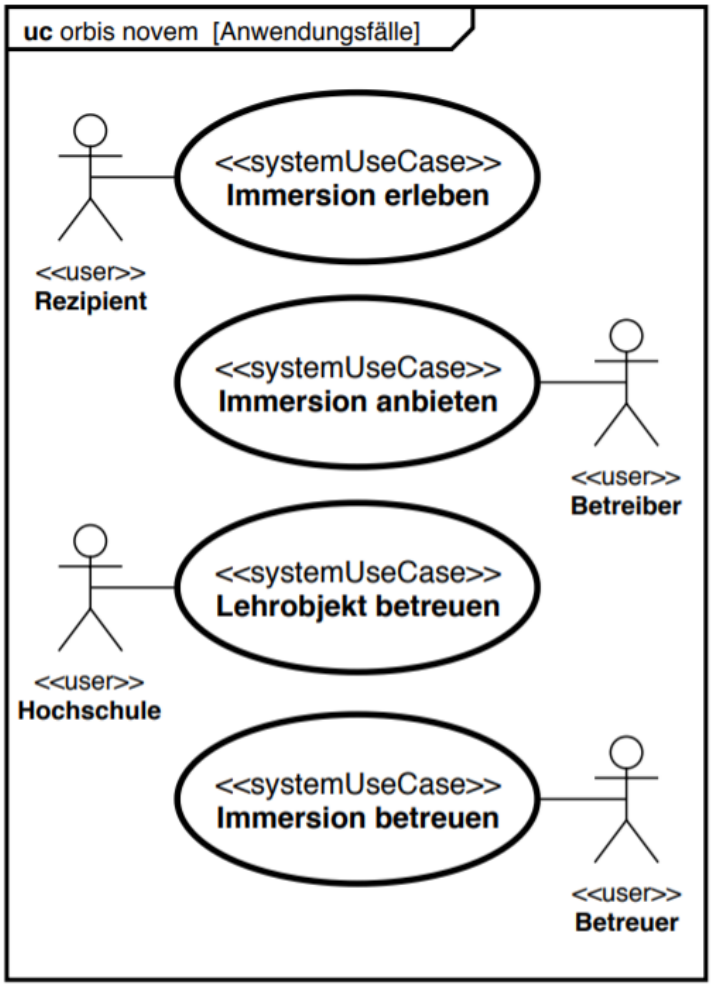

Abbildung 12: Anwendungsfälle

Diese Anwendungsfälle sich sehr allgemein gehalten, das heißt, es wurde noch kein konkreter Auslöser und ein fachliches Ergebnis modelliert. Sie dienen stattdessen der Übersicht über die Dienstleistungen des orbis novem Systems für die verschiedenen Akteure:

- Der Rezipient möchte eine Immersion erleben und der Betreiber mittels des Systems eine Immersion anbieten.
- Der Akteur Hochschule möchte das System in den eigenen Räumlichkeiten und mittels der zur Verfügung gestellten Ausstattung betreuen, um den eigenen Studierenden, Lehrpersonal und Mitarbeiter, sowie eventuell Besuchern ein Lehrobjekt zu bieten.
- Zuletzt möchte der Betreuer eine (geführte) Immersion als Dienstleistung des Systems betreuen.

3.6.1 Systemprozesse

Zuvor definierte Anwendungsfälle werden an dieser Stelle als Input für die Modellierung von Systemprozessen genutzt. Ein Systemprozess beschreibt dabei einen anwendungsfall-übergreifenden Ablauf. Dieser besteht wiederum aus einer Menge von Anwendungsfällen mit

einer sachlogische Ablauffolge [5].

Die Namensgebung an dieser Stelle ist verwirrend und soll daher kurz kommentiert werden. Die Begriffe Systemprozess und Systemanwendungsfall lassen sich leicht für Synonyme halten. Allerdings ist es wichtig, diese Begriffe konsequent zu unterscheiden, besonders da diese unterschiedliche Notationen in der SysML aufweisen und verschiedene Charakteristiken abbilden.

Die Abbildung 13 ‘Systemprozesse mit Essenz‘ wurde gemäß der Leitfrage *Welche Anwendungsfälle haben eine große fachliche Nähe?* modelliert [5].
In der Abbildung 13 werden dazu die beiden, für die Zielsetzung dieser Ausarbeitung relevanten Systemprozesse notiert und in enthaltene Systemanwendungsfälle aufgeschlüsselt.

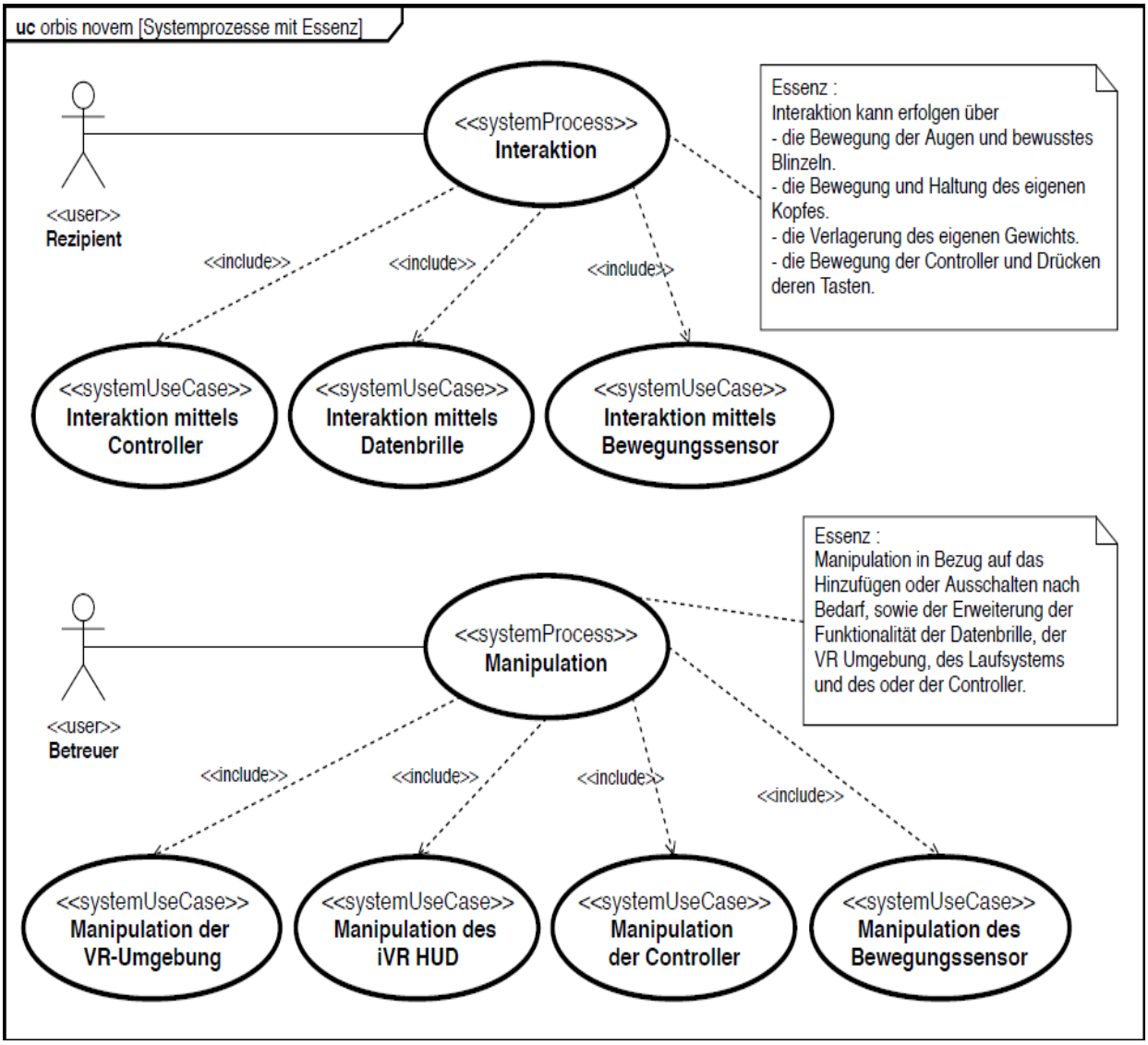

Abbildung 13: Systemprozesse mit Essenz

Ein Systemprozess wird in der SysML als ein spezieller Anwendungsfall mit dem Stereotyp <<systemProcess>> notiert. Um diesen Systemanwendungsfällen zu unterscheiden, wird eine andere Namenskonvention verwendet: Der Name ist ein Substantiv ohne Verb [5].

Damit ergibt sich für orbis novem der Systemprozess *Interaktion* von Seiten des Rezipienten und *Manipulation* von Seiten des Betreuers. Der Systemprozess Interaktion enthält dabei die Systemanwendungsfälle Interaktion mittels Controller, Interaktion mittels Datenbrille und Interaktion mittels Bewegungssensor. Der Systemprozess Manipulation dagegen enthält die Systemanwendungsfälle Manipulation der VR-Umgebung, Manipulation des iVR HUD, Manipulation der Controller und Manipulation des Bewegungssensor. Beide Systemprozesse sind gemäß ihrer grundlegenden Funktionsweise essenziell beschrieben. Auf die einzelnen essenziellen Anwendungsfallschritte jedes Anwendungsfalls wurde nicht eingegangen, da die Subsysteme auf dieser Modellierungsebene als Blackbox betrachtet werden.
An dieser Stelle ist zu erwähnen, dass die Grenze zwischen rein fachlichem Schritten und ersten Architektur-Entscheidungen sehr unscharf ist. Die Reihenfolge der Anwendungsfallschritte ist hier noch nicht relevant. Natürlich sind diese in einer sinnvollen Abfolge zu notieren. Aber oft ist die Reihenfolge technologieabhängig und spielt auf der fachlichen Ebene keine besondere Rolle. Später können die Anwendungsfälle detailliert und technologieabhängig beschrieben werden [5].
Zuletzt wird im Folgenden auf die Modellierung des Systemprozesses *Aktuelle Szene anzeigen* eingegangen. Damit ist der Prozess gemeint, bei dem das System die aktuelle Szene der VR-Umgebung gemäß der aktuellen Parameter auf der Datenbrille anzeigt. Es handelt sich bei diesem Systemprozess um einen von vielen potenziellen kontinuierlich stattfindenden Vorgängen des orbis novem Systems.
Daher wird dieser exemplarisch genutzt, um das folgende Konzept nach SysML zu betrachten: den kontinuierlichen Anwendungsfall. Dazu wird in der Abbildung 14 der Systemprozess Aktuelle Szene anzeigen mittels eines Zustandsautomaten modelliert.

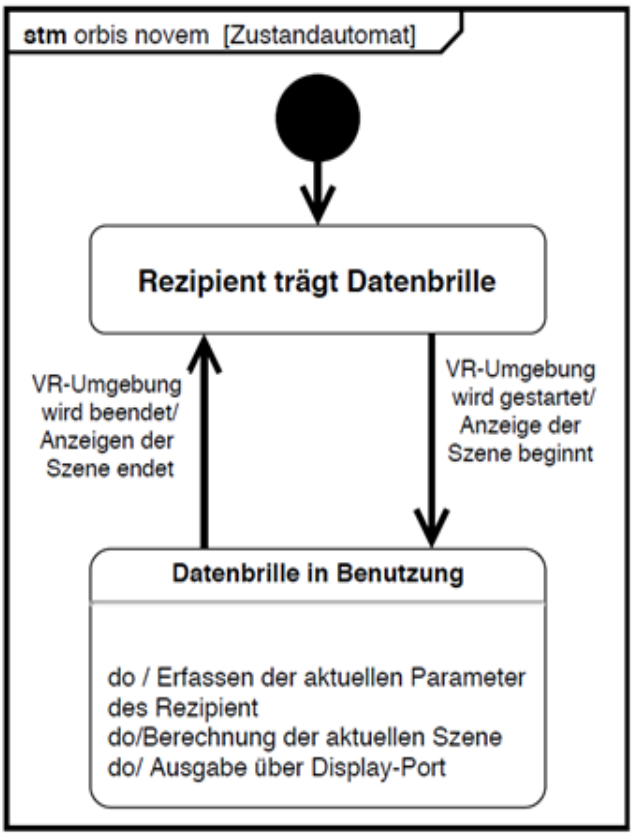

Abbildung 14: Kontinuierlicher Anwendungsfall Aktuelle Szene

Der kontinuierliche Anwendungsfall ist ein spezieller Anwendungsfall, der in einem definierten Systemzustand startet und kontinuierlich Ergebnisse liefert. Ein Endergebnis ist nicht erforderlich [5]. Sobald sich das orbis novem System im Zustand Nutzung befindet, beginnt die Anzeige der aktuellen Szene auf der Datenbrille. Der Anwendungsfall wird durch einen internen Zustandswechsel ausgelöst und das Verlassen des Zustandes beendet den Anwendungsfall. Der Bezug zu Systemzuständen lässt sich gut mit Systemprozessen beschreiben [5].
Die Modellierung mittels Zustandsautomaten wie in Abbildung 14 ist dabei wie folgt zu lesen: Die Vorbedingung ist, dass der Rezipient die Datenbrille trägt, die Auslöser der Anwendungsfälle sind die Auslöser der Transitionen, also das Ein- und Ausschalten der VR-Umgebung über den Steuerungscomputer durch den Betreuer. Die Aktivitäten der Anwendungsfälle sind das Verhalten an den Transitionen, als das kontinuierliche Berechnen und Ausgeben der aktuellen Szene, basierend auf den kontinuierlich erfassten Parameter zur Kopfhaltung und Blickrichtung des Rezipienten.

3.7 Systemstruktur und Fachwissen

An dieser Stelle werden die Ergebnisse der fünften SYSMOD-Phase beschrieben, namentlich *Modellierung der Systemstruktur*.
Innerhalb dieses Abschnittes werden Anwendungsfallabläufe modelliert, das heißt internen Abläufe und Abhängigkeiten der Teilsysteme im System werden analysiert [5].

Im Rahmen der Bearbeitung dieser Phase wurde zunächst der Ablauf einer geführten Immersion untersucht, da das Verständnis dieser sehr wichtig für das System ist. Dazu wurden alle nötigen Interaktionen mit den Subsystemen, sowie zwischen den Systemakteuren Rezipient und Betreuer analysiert und festgehalten. Das relevanteste Ergebnis dieser Phase ist dabei die Abbildung 2 ‘Aktivitäten für eine geführte Immersion‘, welche bereits in Kapitel 2 eingeführt wurde. Da dort bereits die wichtigsten Erkenntnisse beschrieben sind, wird an dieser Stelle nicht nochmals oder weiter auf diese Phase eingegangen.
An dieser Stelle wird zudem der Vollständigkeit halber auch kurz die sechste und letzte SYSMOD Phase *Fachwissen sammeln und formal beschreiben* kurz betrachtet. Zu diesem Zweck können Anforderungen und Anwendungsfälle genutzt werden, um Strukturen und fachliche Begriffe des Systems zu definieren. Innerhalb dieser Aktivität können fachliche Begriffe des Projektes in einem Glossar beschrieben, was zum Ziel hat, dass alle Projektbeteiligten dasselbe Verständnis der für die Entwicklung elementaren Begriffe haben [5].
Für das Verständnis des orbis novem System ist besonders ein einheitlicher Immersionsbegriffs notwendig, welcher zuvor definiert wurde. Für das Verständnis dieser Ausarbeitung als Dokumentation sind die Begriffe der erfolgreichen und der geführten Immersion relevant.
Eine weitere Bearbeitung dieser Phase wird für die Zielsetzung dieser Arbeit als nicht relevant erachtet. Die Analyse der zentralen Systemaspekte lässt sich der vorliegenden Dokumentation in ihrer Gesamtheit entnehmen.
Mit dem BSCW verfügt das Masterprojekt VR Lab glücklicherweise um ein Verwaltungssystem, das rege genutzt wird um Fachwissen zu sammeln und zu archivieren. Damit bietet es Zugang zu einer unverzichtbaren Wissensbasis für Teilnehmer und Interessierte.

4 Tests

In diesem Kapitel wird ergänzend zu den Phasen der systemischen Modellierung auf die Möglichkeiten des Testens eingegangen. Dazu wird auf die Gestaltung einer Nutzerstudie eingegangen. Ebenso werden drei verschiedene Test-Arten betrachtet und innerhalb dieser Tests-Szenarien für das orbis novem System skizziert.
Dabei werden Integrations-, System- und Abnahmetests betrachtet, wie sie unter anderem in [4], [6] und [10] beschrieben werden. Besonders das V-Modell des Testens nach [6] diente hierbei als Orientierung und kann in dieser Quelle nachgelesen werden.
Um die Bereitstellung ausgewählter funktionaler Anforderungen sicherzustellen und die

Akzeptanz des Rezipienten zu testen, wird ein Systemtest skizziert. Ferner wird ein Abnahmetest skizziert, um die Akzeptanz des Systembetreibers sicherzustellen. Komponententests nach [4] wurden dabei nicht explizit betrachtet. Zum einen konnten aus Zeitgründen keine Szenarien für Komponententests mehr festgelegt werden, zum anderen dienen die anderen drei Test-Szenarien als Validierung der Komponenten auf einem übergeordneten Level.

4.1 Integrationstest

Mittels Integrationstests werden die einzelnen Teilsysteme in ihrem Zusammenspiel getestet. Nachdem die Subsysteme zusammengesetzt also integriert wurden, muss das einwandfreie Zusammenspiel mit den anderen Subsystemen überprüft werden. Der Integrationstest verfolgt das Ziel, Fehlerzustände in Schnittstellen zwischen integrierten Komponenten zu finden [10].

Hierbei sind bei orbis novem Integrationstests für die folgenden Subsysteme von besonderem Interesse:

1. Zum einen die Integration der beiden Softwarekomponenten iVR HUD und Neue Welt 9 Umgebung. Ein Testszenario muss sicherstellen, dass virtuelle Elemente innerhalb der VR-Umgebung sich tatsächlich durch Blick-Steuerungen manipulieren lassen, wenn diese erkannt werden.
2. Zum anderen muss die Integration zwischen der HTC Vive Datenbrille und der Softwarekomponente iVR HUD angemessen durchgeführt und getestet werden. Für aufbauende Funktionalitäten ist daher sicherzustellen, dass die Information darüber, dass ein Blinzeln- oder Blick-Befehl des Rezipienten erkannt wurde korrekt an den Steuerungscomputer weitergegeben wird. Diese Information muss dann korrekt verarbeitet und ausfallsicher zurück an die Datenbrille gesendet werden, damit die jeweilige Reaktion des manipulierten Elements ausgelöst werden kann.

Die notwendigen Grundlagen für weitere und konkretere Integrationstests in Bezug auf die Subsysteme iVR HUD und Datenbrille lassen sich dabei [1] entnehmen.

Weitere Testszenarien für Integrationstests lassen sich zudem aus den zuvor beschriebenen, funktionalen Anforderungen F1 bis F5 ableiten. Diese können je nach aktuellem Technologiewissen in den zukünftigen Iterationen konkretisiert werden, beziehungsweise ihrerseits wiederum auf spezialisierten Komponententestfällen basieren.

4.2 Systemtest

Ein Systemtest meint die Validierung des gesamten Systems durch einen Benutzer, nachdem die Integration der einzelnen Systembausteine beendet wurde. Bei der Durchführung eines Systemtests ist darauf zu achten, dass jede, für den Test genutzte Test-Umgebung, der späteren Produktivumgebung so ähnlich wie möglich ist. Dazu sollte die später eingesetzte Hard- und Software eingesetzt werden, um den Test so realistisch wie möglich zu halten [10].
Diese Anforderung ist bei der Durchführung des beschriebenen Einsatzszenarios (eine erfolgreiche, geführte Immersion) mittels des orbis novem Systems gegeben. Damit ist der Systemtest eine naheliegende Test-Art, wenngleich diese mit spezialisierten Tests kombiniert werden sollte. Jede Durchführung des Einsatzszenarios ist in diesem Kontext ein oberflächlicher, aber angemessener Systemtest durch den Rezipienten.

Im Folgenden wird ein konkreter, denkbarer Systemtest vorgestellt: Dieser könnte darin bestehen, einen Rezipienten eine virtuelle Tür in der Neuen Welt 9 öffnen zu lassen. Dieser muss dazu mittels des Laufbandes und den Controllern zu einer bestimmten Tür navigieren. Dann muss er oder sie diese Tür mittels eines Blinzeln-Befehls anklicken. Anschließend muss sich die Tür öffnen und der Systemtest ist beendet, wenn der Rezipient den zugehörigen Raum betreten hat. Dabei wird zwar keine vollständige Immersion geprüft, welche subjektiv erlebt wird und daher anders geprüft werden muss. Aber indirekt wird mit diesem Testszenario geprüft, ob eine jeweils angemessene Reaktion für jedes Element eingebaut wurde. Ebenso ist indirekt eine Prüfung dessen enthalten, ob der Datenaustausch zwischen allen Komponenten möglich ist. Die Überprüfung dessen, ob diese Übertragungen mit der zuvor als angemessen festgelegten Latenz passiert, ist dabei Betrachtungsgegenstand des Abnahmetest.

4.3 Abnahmetest

Der Abnahmetest stellt die Einhaltung der vereinbarten Leistungsmerkmale sicher. Im Gegensatz zum Systemtest, der mit dem Rezipienten durchgeführt wird, prüft der hier konzipierte Abnahmetest die Akzeptanz durch den Systembetreiber, sowie die vertragliche Akzeptanz [6].
Zu diesem Zweck wird an dieser Stelle angenommen, dass der Systembetreiber der Stakeholder Professor, wie zuvor beschrieben ist. Zudem wird angenommen, dass die Akzeptanz des Systembetreibers ausschließlich davon abhängig ist, ob das System die Anforderungen nach Latenz erfüllt. Mit Latenz ist hierbei explizit die Verzögerung in der Übertragung von Daten in der geführten Immersion gemeint. Die Latenz ist daher ein so zentrales Merkmal des Testens, da bereits eine einzige Übertragung mit zu hoher Latenz den immersiven Irrtum für den Rezipienten aufheben kann. Ebenso ist eine nicht konstante oder vernachlässigbare Latenz bei der Exploration von VR-Umgebungen sehr frustrierend. Ebenso kann dies die Auftritts-Wahrscheinlichkeit von Virtual Reality Sickness erhöhen. Bei Virtual Reality Sickness handelt es sich um die häufig auftretende ‚Seekrankheit' in der Nutzung von

VR-Umgebungen durch Rezipienten. Diese äußert sich zum Beispiel in Kopfschmerzen oder Übelkeit, je nach Konstitution des Rezipienten.

Basierend auf diesen Annahmen muss die vernachlässigbare Latenz (Soll-Latenz) festgehalten werden. Diese sei ein Wert kleiner als der Wert X in Millisekunden. Anhand dieser Soll-Latenz kann dann ein Testszenario konzipiert werden, bei dem die Ist-Latenz des orbis novem System mittels automatisierter Zeitnahme gemessen wird. Die Daten zur Ist-Latenz werden dann bei einer Abnahme mit der Soll-Latenz verglichen.

4.4 Nutzerstudie

Nachdem zuvor Systemtest und Abnahmeteste betrachtet wurden, geht dieser Abschnitt ergänzend auf eine mögliche Nutzerstudie ein. Eine Nutzer-Befragung oder Nutzerstudie lässt sich gut mit den genannten Test-Arten kombinieren. Bei der Gestaltung einer Nutzerstudie für das orbis novem System sind die folgenden Hinweise hilfreich:
In einer Nutzerbefragung könnte das subjektive Immersions-Erleben einzelner Rezipienten mittels digitaler oder analoger Fragebögen abgefragt werden. Dabei sollte das Alter, Geschlecht und die Vorerfahrung mit VR-Umgebungen abgefragt werden, da diesen Merkmalen ein Einfluss auf das wahrscheinliche Immersions-Erlebnis zugeschrieben wird. Diese sind auch ein Einfluss in Bezug auf Virtual Reality Sickness.

Die konzipierten Fragebögen sollten zudem die Likert-Skala nutzen, um den Befragten die nötige Ausdrucksfreiheit innerhalb ihrer Antworten zu bieten. Die Likert-Skalierung ist eine bipolare Skalierungsmethode, die entweder die positive oder negative Reaktion auf eine Anweisung misst. Manchmal wird eine gerade Punkteskala verwendet, bei der die mittlere Option " Weder Zustimmung noch Widerspruch " nicht verfügbar ist. Die neutrale Option kann als eine einfache Option angesehen werden, die man wählen kann, wenn ein Befragter unsicher ist, und so ist es fraglich, ob es sich um eine echte neutrale Option handelt. [11]

Dabei können Fragen in der Form von Aussagen gestellt werden, der ein Befragter zustimmen oder sie negieren soll. Das Format eines typischen fünfstufigen Likert-Elements nach [11] als Antwortmöglichkeit könnte beispielsweise so aussehen:

Aussage: „Die Latenz (Verzögerungszeit) in der Anzeige der VR-Elemente war vernachlässigbar, d.h. sie hat mein Immersions-Erlebnis nicht beeinträchtigt“

- Ich bin absolut nicht einverstanden
- Ich bin nicht einverstanden
- Weder Zustimmung noch Widerspruch
- Ich bin einverstanden

- Ich bin völlig einverstanden

Freitext: [_____]

Jede Frage sollte auch mit Kommentarmöglichkeiten kombiniert werden, also Freitextfeldern für die Kommentierung der Antwort durch den Befragten.

Gerade bei der Entwicklung von Anwendungen, die explizit mehrere Formen der Interaktion erfordern, sollten durchgängig darauf getestet werden, ob alle diese Funktionalitäten dem Rezipienten zu jedem Zeitpunkt korrekt zur Verfügung stehen. Der Erfolg dieser Interaktionsmöglichkeiten kann ebenfalls durch eine Nutzerstudie validiert werden.

5 Aktueller Stand

5.1 Erste Iteration

5.1.1 iVR-HUD v0.1.a Prototyp

An dieser Stelle soll kurz auf den iVR-HUD Prototypen nach [1] eingegangen werden, bei dessen Entwicklung sich die Idee zum orbis novem Projekt ergab.
Ziel des iVR-HUD v0.1.a Prototypen war die Erstellung eines klassischen HUD in VR, hier zunächst noch ohne Interaktivität jeglicher Art. Unter einem klassischen HUD sind UI-Elemente zu verstehen, die üblicherweise unabhängig von Kamerabewegungen am Bildschirmrand liegen und immer sichtbar sind, also über alle anderen Spiel- oder Anwendungsobjekte gerendert werden.
Ein Bildschirmrand lässt sich in VR zwar technisch definieren, die subjektive Wahrnehmung dessen kann durch Probleme in der Darstellung (die Anzeige am technischen Bildschirmrand der HTC Vive scheint von schlechterer Qualität zu sein) und der Wahrnehmung (Objekte im Blickfeldrand sind schwerer zu erkennen und zu fokussieren) aber stark abweichen. Daraus entsteht im vorliegenden Kontext eine zusätzliche Anforderung an die Sichtbarkeit. Somit soll hier ein HUD entstehen, bei welchem die UI-Elemente sowohl in Relation zu anderen Objekten als auch in Relation zur Kamera- / Kopfbewegung immer sichtbar sind. Da eine Interaktion zu diesem Zeitpunkt noch nicht vorausgesetzt wird, genügt die Anzeige von statischen Texten im UI.

Das orbis novem System lässt sich, wie zuvor modelliert, implementieren durch die Nutzung des zugehörigen Plugins aus [1]:
Unter der Annahme, dass mit dem Produkt von Pupil Labs weitergearbeitet werden soll, braucht es, neben der Unity Engine das zugehörige Plugin von Pupil Labs. Zudem das erwähnte iVR HUD Plugin. Dieses kann im BSCW runtergeladen (https://bscwserv.reutlingen-

university.de/bscw/bscw.cgi/11378953) und vollständig in das vorbereitete VR-Projekt importiert werden. Hier bietet es sich an, die (vermutlich überwiegend leere) Default-Szene von Unity mit dem Eyetracker zu starten, um sicherzustellen, dass alles funktioniert. Falls ja ist das die erste Grundlage, um Eyetracking in VR nutzen zu können. Für andere Projekte, die sich hier nur über das Eyetracking informieren möchten, muss stattdessen das Pupil Gaze Unity Plugin von deren Webseite heruntergeladen und integriert werden. Die ursprünglich genutzte Version findet sich als [pupil_v0912_windows_x64.zip] im BSCW. [1]

5.2 Zweite Iteration

5.2.1 Aktueller Stand und Inbetriebnahme

Das Projekt NeueWelt9 in Verbindung mit VR funktioniert nur über die Unity-Version „**Unity 2018.3.11f1**“. Um eine neuere Version nutzen zu können, müssen dementsprechend Änderungen an NeueWelt9 vorgenommen werden. Aktuelle Versionen zeigen verschiedene Fehler auf, welche mit Hilfe vom NeueWelt9 Team behoben werden sollten.

Verwendete Plug-Ins

Folgende Plug-Ins sind notwendig, um das Projekt zum Laufen zu bekommen. Nicht obligatorische Plug-Ins werden markiert.

CybSDK (nicht obligatorisch) zur Erweiterung:

Dieses Plug-In wird für die Nutzung vom Virtualizer genutzt. Es ist nicht im Unity-Store erhältlich, sondern kann nur von der offiziellen Webseite https://developer.cyberith.com/login?next=%2F (Login Daten sind im BSCW hinterlegt) oder vom BSCW heruntergeladen werden (https://bscwserv.reutlingen-university.de/bscw/bscw.cgi/10829015).

Der Virtualizer ermöglicht das Laufen auf der Stelle in der wirklichen Welt, während sich gleichzeitig ein 3D Objekt in der virtuellen Welt bewegt. Die Orientierung geschieht durch die Vive.

Abbildung 15: Der Virtualizer

Photon Unity Networking:

Ermöglicht das Multiplayer-Spiel mit zwei oder mehr Spielern. Hierfür wird ein Account bei Photon benötigt. Ein Account existiert bereits. Um das Spiel auf zwei Rechner laufen zu lassen, wird eine PUN benötigt.

1. Photon Webseite abrufen. Https://dashboard.photonengine.com/en-US/publiccloud
2. Anmeldedaten sind unter folgendem Link zu finden: https://bscwserv.reutlingen-university.de/bscw/bscw.cgi/12605243?op=preview&back_url=10940761
3. Die PUN wird dann im Reiter App „OrbisNovem“ unter „App ID“ angezeigt. Es können, nach Bedarf, auch mehrere PUN’s erstellt werden (siehe Abbildung 16).

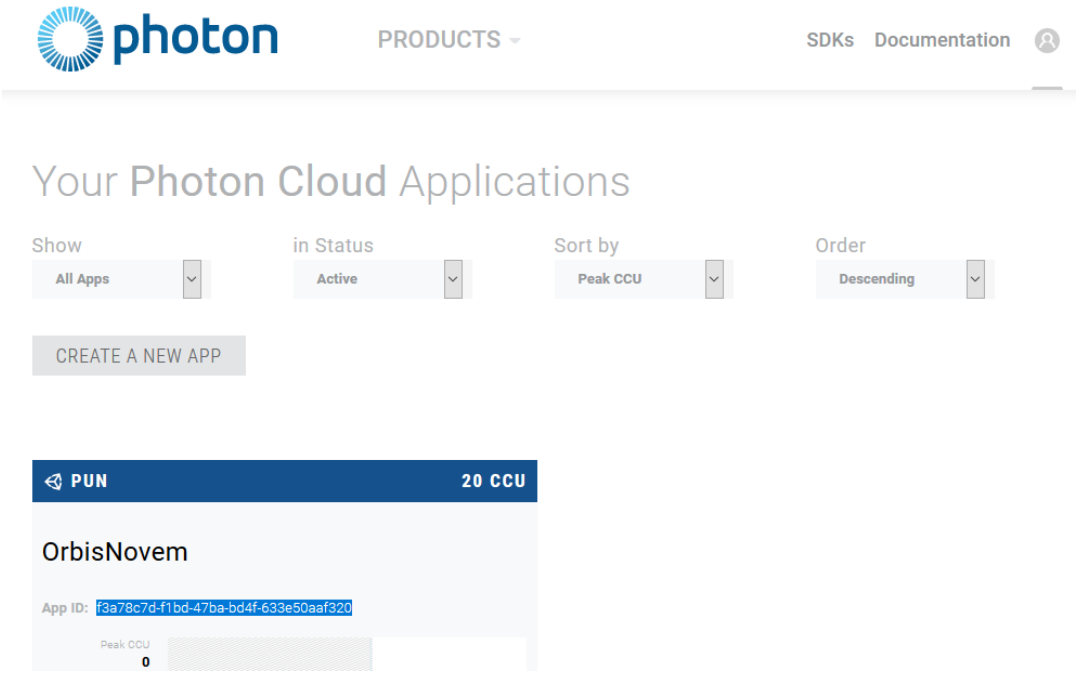

Abbildung 16: Dashboard Photon

SteamVR:

Dieses Plug-In ermöglicht eine 3D Ansicht in der Virtuellen Welt in Verbindung mit dem Vive-Set. Erhältlich ist dieses Plug-In im Unity Store. Folgende Hardware wird hierfür benötigt:

- Zwei Basestations (hängen am Gerüst)
- Vive Headset
- Vive-Controller

Abbildung 18: Logo Steam VR

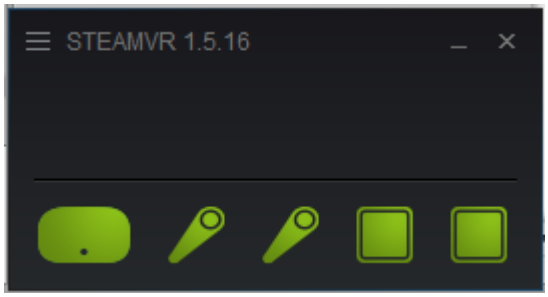

Abbildung 17: Anzeige des Vive-Sets (Alles verbunden und aktiv)

Anmeldedaten für SteamVR sind im BSCW zu finden. https://bscwserv.reutlingen-university.de/bscw/bscw.cgi/12605243?op=preview&back_url=10940761

VRTK

VRTK ermöglich die Interaktion mit den Controllern. Dieses wird für folgendes genutzt.

- Controller Interaktionen
- Teleportieren
- Elemente manipulieren
- Laufen ermöglicht durch Interaktion mit dem Touchpad

Abbildung 19: Logo VRTK

Inbetriebnahme

Ziel dieses Projektes ist es, eine Multiplayer-Funktionalität anzubieten. Hierfür wurde die NeueWelt9 als virtuelle Umgebung genutzt. Zwei Rezipienten, mit jeweils einer Vive, können sich gemeinsam in der virtuellen Umgebung fortbewegen und sehen sich als 3D-Objekt in dieser Umgebung. Des Weiteren ist es möglich, sich zu Teleportieren, Gegenstände aufzuheben und sich mit dem Touchpad vom Controller zu bewegen.

Der erste Schritt ist es, die benötigten Plug-Ins in der richtigen Unity-Version herunterzuladen und zu Installieren. Das Plug-In für den Virtualizer wird nur benötigt, wenn auch der Virtualizer genutzt wird. Wird der Virtualizer nicht genutzt, aber das Plug-In ist installiert, treten Serverprobleme bei der Verbindung auf. Damit zwei Rezipienten in einer Szene gemeinsam interagieren können, muss das Unity Projekt auf zwei Rechnern laufen. Heißt, es muss eine Kopie des Projektes auf einem anderen Rechner existieren.

Für die Serververbindung wird Photon genutzt. Mit der PUN von Photon könne sich zwei oder mehrere Spieler mit derselben App-ID in einer Szene finden. Die App-ID wird dann an beiden Rechnern in Unity eingetragen und muss identisch sein. Bei der Erstinstallation wird direkt nach der App-ID gefragt. Diese kann aber auch im Nachhinein eingetragen werden.

Dafür gehe wie folgt in Unity vor:

- Window
- Photon Unity Networking
- PUN Wizard
- Klicke auf „Locate PhotonServerSettings“

Trage nun die App-ID in die dafür vorhergesehene Zeile ein (siehe Abbildung 20)

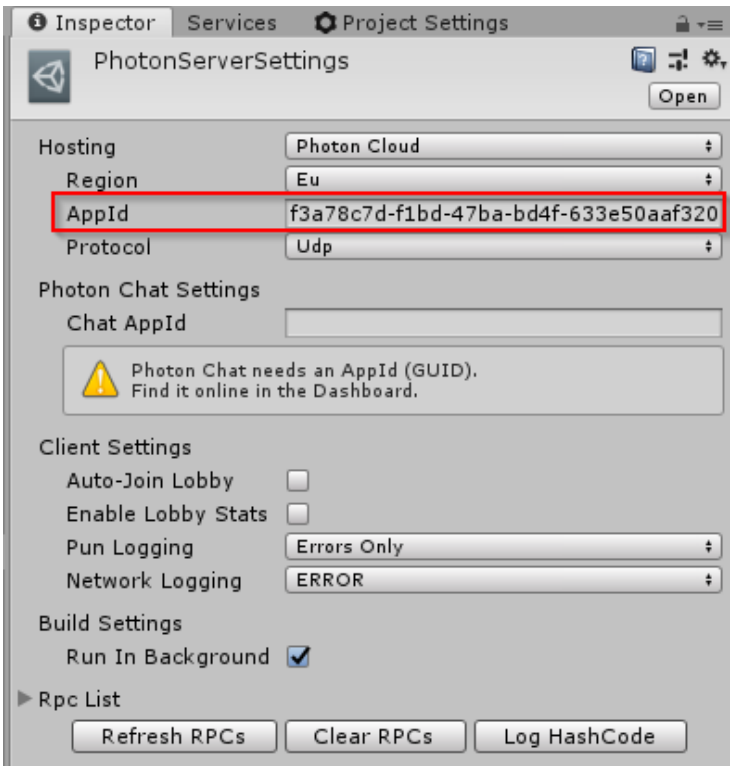

Abbildung 20: PhotonUnitySettings für die Eintragung der App-ID

Das Skript für die Netzwerkverbindung und andere relevante Skripte sowie alle Plug-Ins werden im Ordner „Assets“ im Reiter „Project“ zu finden sein (siehe Abbildung 21).

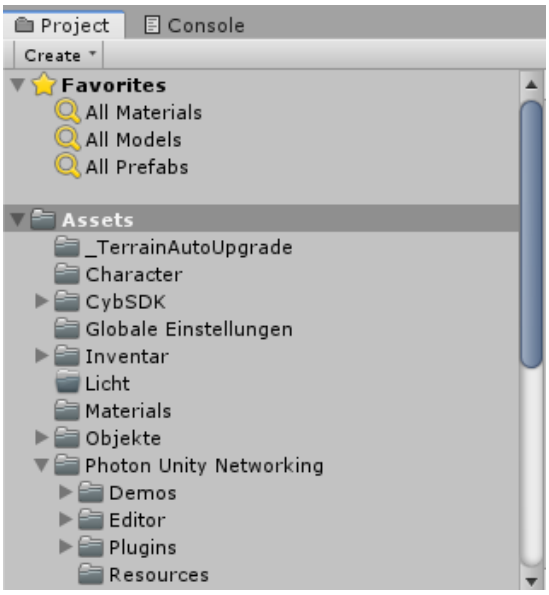

Abbildung 21: Projektassets

Das komplette Projekt befindet sich in der „Multiplayer_NW9“ Szene, welche auch unter den Assets gespeichert ist. Die „GameObjects“ befinden sich dann in dieser Szene Hierarchisch

untergliedert. Der „NetworkManager“ befindet sich im übergeordneten Ordner „[VRTK_SDKManager]“ (siehe Abbildung 22). Das Skript befindet sich direkt Im Übergeordnet Ordner „Assets“. Der Code im Skript wird von Photon selber bereitgestellt und wurde durch drei GameObjects erweitert.

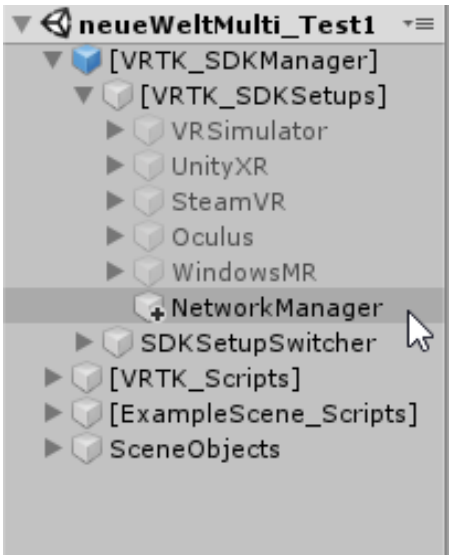

Abbildung 22: Hierarchie der Szene

Zudem wurde im Code beim Betreten eines virtuellen Raums die Position und die Rotation jedes GameObjects hinzugefügt (siehe Abbildung 23) welches von der Vive selber ausgeht. Hierfür wurde ein weiteres Skript Namens „ViveManager“ für die Instanziierung der Position und der Rotation erstellt. Dieses Skript sorgt für den Zugang zu unserem Kopf, zu unserer linken und rechten Hand.

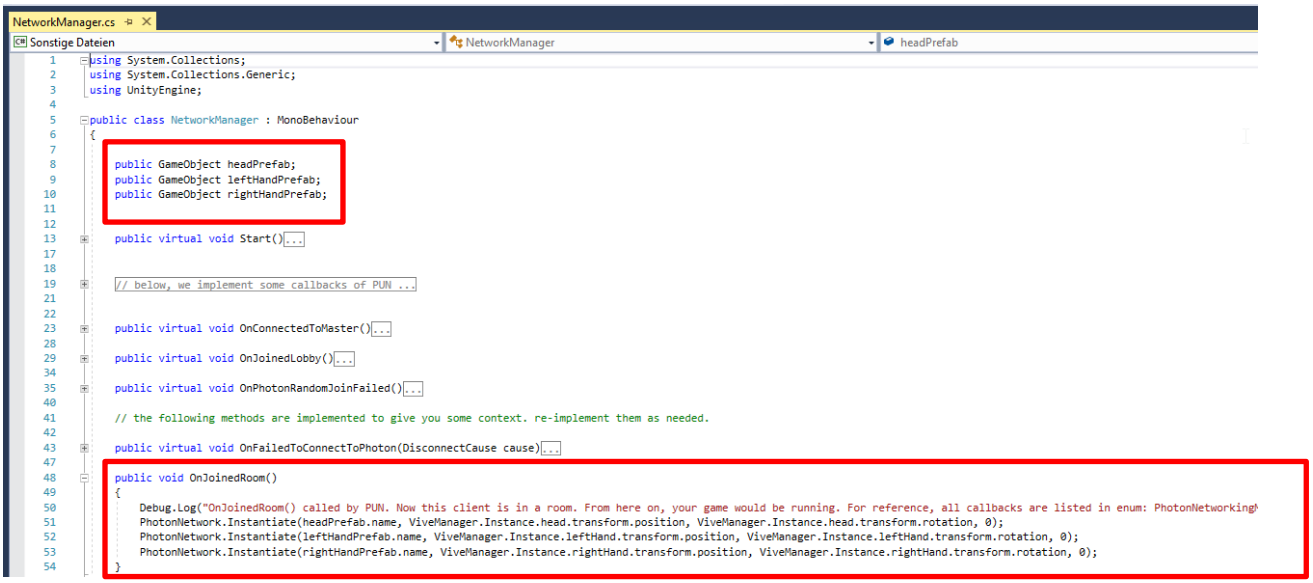

Abbildung 23: Ergänzungen im NetworkManager

Das Skript wird dann in der „CameraRig“ eingefügt und den jeweiligen GameObjects „head“, „leftHand“ und „rightHand“ werden die Kamera, der rechte Controller und der linke Controller übergeben (siehe Abbildung 24).

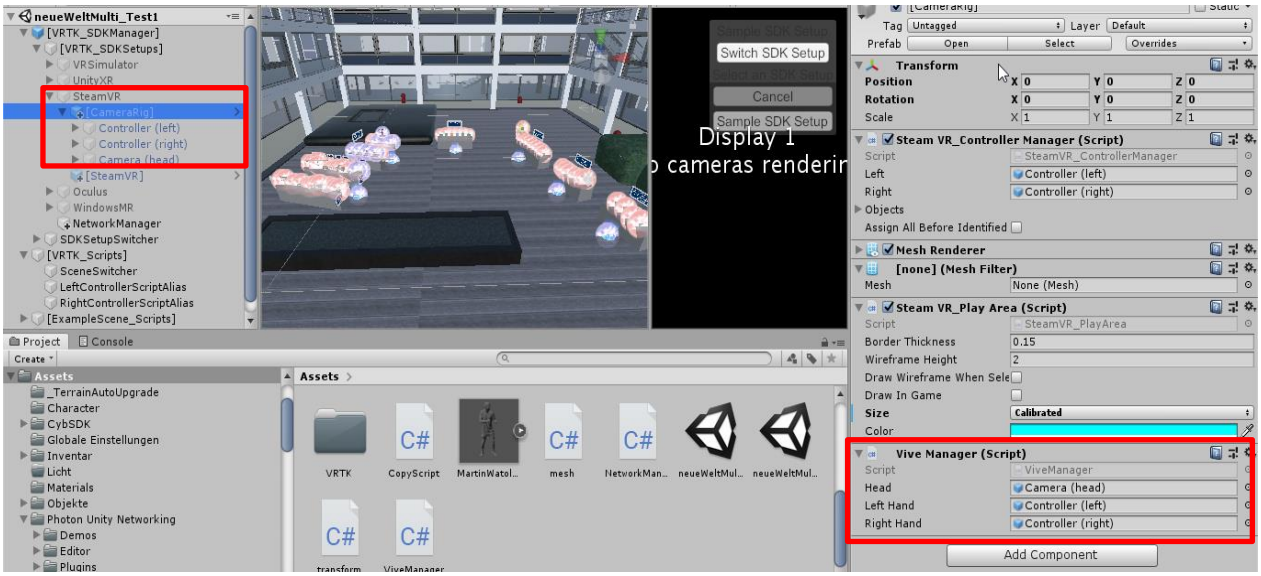

Abbildung 24: Zugang zur Vive

Die Prefabs, welche den GameObjects „Head Prefab“, „Left Hand Prefab“ und „Right Hand Prefab“ für den NetworkManager übergeben werden, befinden sich im übergeordneten Ordner „Assets“ unter „Prefabs“ und „Resources“.
Diese Ordner dürfen unter keinen Umständen umbenannt werden, da Unity sonst den Pfad zu den Objekten nicht finden kann, und die Objekte dann nicht in der Szene angezeigt werden. Um ein GameObject einem Prefab zu übergeben, wird dieser per Drag&Drop in das jeweilige Prefab-GameObject vom Ordner „Resources“ gezogen (siehe Abbildung 25).

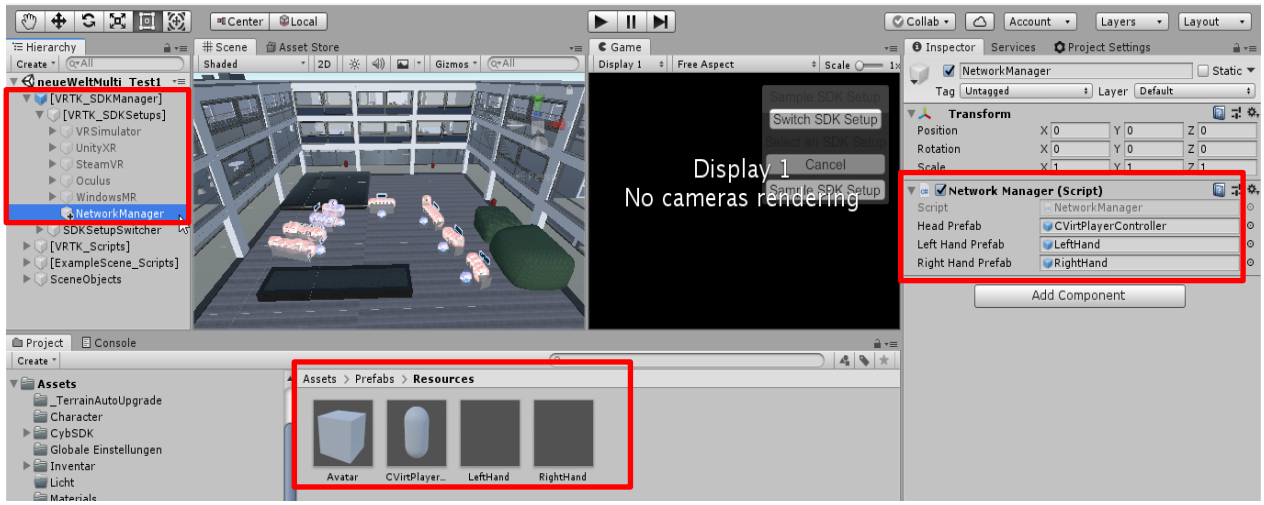

Abbildung 25: Prefabs werden an das GameObject übergeben

Es ist auch möglich die Prefabs durch andere auszutauschen, sodass der Avatar eine andere Gestalt annimmt. In Abbildung 25 roter Kasten oben links, werden außer dem NetworkManager, noch andere „GameObjects“ angezeigt. Da wir mit SteamVR arbeiten ist nur dieses für uns relevant und wird vom VRTK-Plug-In zur Verfügung gestellt. Alle anderen sind nicht aktiv, können aber genutzt werden sobald es nötig ist. Damit die Kamera von Unity dem Avatar folgt, wurde das Skript „CopyScript“ erstellt, welches im übergeordneten Ordner „Assets“ zu finden ist und dem Avatar als Komponente übergeben. Zudem müssen die von Photon zur Verfügung gestellten Skripte „Photon View und Photon Transform View“ hinzugefügt werden, um zu gewährleisten, dass sich beide Spieler in derselben Szene sehen (siehe Abbildung 26). Diese drei Komponenten müssen auch dem linken und rechten Controller hinzugefügt werden.

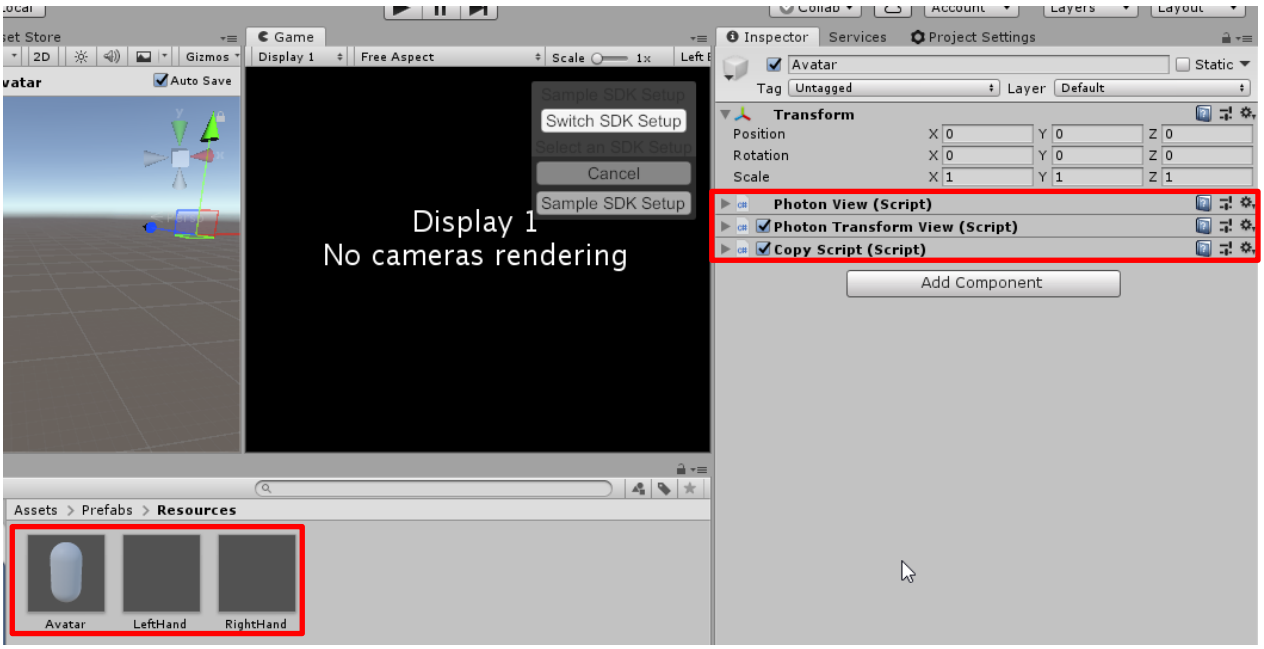

Abbildung 26: Copy Script wird an die Ressourcen übergeben

Im „[VRTK-Scripts]“ GameObject-Ordner werden dem „LeftControllerScriptAlias“ und „RightControllerScriptAlias“ Controller, die Skripte für die Zuordnung der Interaktionen im Controller in der virtuellen Welt übergeben. Diese Skripte werden vom VRTK-Plug-In geliefert und sind im übergeordnete Assets-Ordner unter „VRTK – Source – Scripts“ zu finden.

Im „[ExampleScene_Scripts]“ GameObject-Ordner werden dann dem „LeftController“ und „RightController“ in den beiden vorhandenen Komponenten das „LeftControllerScriptAlias“ und das „RightControllerScriptAlias“ übergeben (siehe Abbildung 27). Beide Komponenten werden vom VRTK-geliefert und zeigen die Interaktion mit dem Controller. Hier die Touchpad- und Pointeraktionen wenn mit dem Controller auf etwas gezeigt wird.

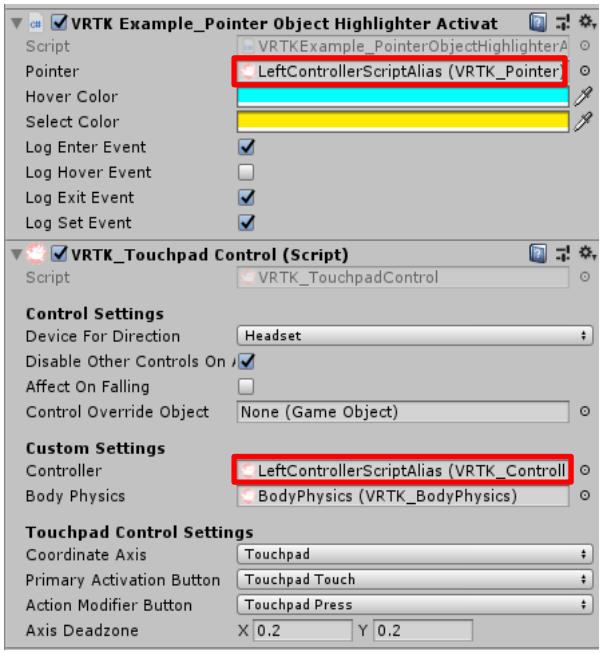

Abbildung 27: Linker Controller Interaktionsbereich
(Rechter Controller Interaktionsbereich identisch)

Der „SceneObjects“ Ordner weist alle Objekte auf, die in der Szene zu sehen sind. Dazu gehört NeueWelt9 und die Tische vor Gebäude 9 die zum Testen der Interaktionen genutzt wurden. Es ist möglich mit dem Controller Gegenstände aufzuheben und mit Ihnen zu interagieren. Damit die Person, welche das Objekt aufhebt, auch der Besitzer des Objektes ist, wird zu jedem interaktiven Objekt, heißt jedes Objekt, das ich aufheben möchte oder in jeglicher Art mit interagieren kann, ein Skript Namens „ChangeOwner“ übergeben. Zudem muss jedem interaktiven Objekt auch das Skript „Photon View und Photon Transform View“ übergeben werden, damit beide Spieler die gleichen Objekte und sich gegenseitig in einer Szene sehen (siehe Abbildung 26 als Beispiel). Nachdem das Skript dem interaktiven Objekt (z. B. Würfel) hinzugefügt wurde, muss das Skript „VRTK_Interactable Object“ im „ChangeOwner“ Skript per Drag&Drop übergeben werden.

Zudem muss im „Photon View“ Skript der „Owner“ auf „Takeover“ gesetzt werden, damit auch wirklich diejenige Person, welches das Objekt aufhebt, auch der Besitzer ist (siehe Abbildung 28). Nun kann mit dem Controller und mit Objekten in der Szene interagiert werden.

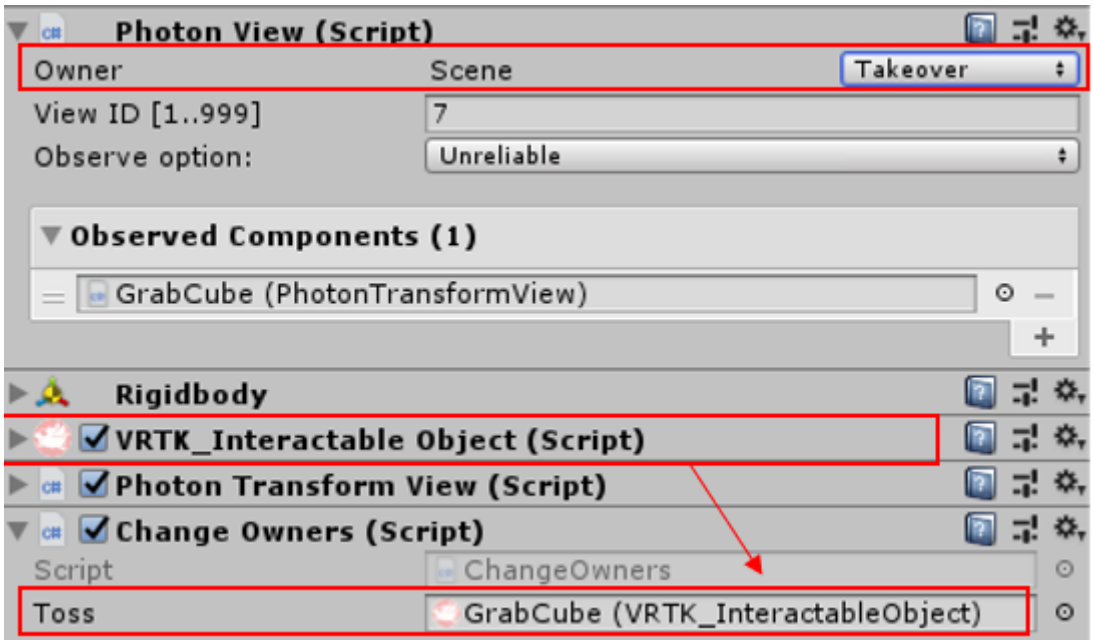

Abbildung 28: Skripte einfügen, um mit einem Objekt interagieren zu können

5.2.2 Schritt für Schritt Inbetriebnahme

Der folgende Abschnitt skizziert die Inbetriebnahme gemäß dem aktuellen Stand.

1. Unity Version **„2018.3.11f1“** öffnen
2. Ist das Projekt nicht unter den Projekten von Unity zu finden, muss es unter „Dokumente – OrbisNovem – neueWelt9“ geöffnet werden
3. Nun müssen alle Plug-Ins unter „Asset Store“ installiert werden.
4. Als nächstes muss die PUN-ID eingetragen werden.
5. In manchen Fällen wird eine Fehlermeldung angezeigt, die besagt, dass OpenVR installiert werden muss. Hierfür unter Window – Package Manager gehen und den OpenVR Desktop installieren.
6. Sichergehen, dass das Projekt auch auf einem anderen Rechner vorhanden ist und auch auf diesem öffnen.
7. SteamVR wird in der Regel automatisch geöffnet.
8. Sichergehen, dass alle Basestations an sind (insgesamt vier, zwei für jede Vive).
9. Sichergehen, dass beide Vives jeweils an einen Rechner angeschlossen sind. Die Vive über den Laptop funktioniert nur über den Mini-Port Eingang und nicht über den HDMI Eingang.
10. Es muss darauf geachtet werden, dass im SteamVR Fenster, zwei Basestations, ein Head (Vive) und zwei Controller für jeweils beide Rechner grün angezeigt werden.
11. Das Spiel auf beiden Rechnern starten

6 Fazit

In diesem Abschnitt erfolgen abschließend eine Zusammenfassung der Arbeit, sowie ein Ausblick bezüglich der künftigen Nutzung. Dies soll vor allem den Beitrag der vorliegenden Ausarbeitung veranschaulichen.

6.1 Zusammenfassung

Der Beitrag der Arbeit wird an dieser Stelle zusammengefasst. Die Basis der vorliegenden Ausarbeitung ist eine systemische Betrachtung des geplanten Immersions-Systems orbis novem, welches im Rahmen des Wahlfachs VR Labor von Studierenden entwickelt wird. Diese Betrachtung und Modellierung wurde in Kapitel 3 nach dem SYSMOD Prozess aus [5] durchgeführt und die relevanten Aspekte, die eine Immersion unterstützen, untersucht und dokumentiert. Dabei wurden die einzelnen Phasen vorgestellt und deren Ergebnisse in Bezug auf das orbis novem System, beziehungsweise orbis novem System of Systems dokumentiert. Die Entwicklung von System of Systems, also Systeme die aus mehreren sekundären Systemen bestehen, bedeutet immer, dass bei der Erstellung das Wissen aus mehreren, verschiedene Disziplinen zusammen kommt [3].
Diverse Aspekte des Systems wurden in Bezug auf ihre Unterstützungsfähigkeit einer Immersion ausgewählt, analysiert und abstrahiert. Diese wurden zudem in der vorliegenden Dokumentation mittels Modellen und Diagrammen nach der Modellierungssprache SysML modelliert. Das indirekte Ziel dieser Ausarbeitung ist dabei das Erstellen einer Dokumentation für Personen, welche das orbis novem Gesamtsystem oder dessen Subsysteme. Die Strukturen, Anforderungen und das Verhalten des Systems werden mittels einer standardisierten Sprache erklärt und unterstützen damit beliebig geartete Weiterentwicklungen. Besonders die erstellten Modelle sollen zukünftige Teilnehmer des VR Labors dabei unterstützen, das System und / oder dessen involvierte Teilsysteme besser zu verstehen, korrekt zu bedienen und als Basis eigener Projekte zu nutzen.
Zudem wurden im Kapitel 4 mögliche Testszenarien für Integrations-, System- und Abnahmetests und eine Nutzerstudie vorgestellt. In Kapitel 5 wurde der, zum Zeitpunkt der Verfassung dieser Dokumentation, aktuelle Stand des Projektes betrachtet.
Für das Verständnis dessen wird dringend geraten, die zugehörigen Dokumentationen der Teilsysteme zu lesen, besonders [1] für das iVR HUD.

Zuletzt wird noch auf mögliche Erweiterungen und die zukünftige Nutzung des orbis novem Systems, beziehungsweise dessen Teilsysteme eingegangen.

6.2 Erweiterungen

Im Laufe der Betreuung der Informatics Inside kam es zu der Idee für die folgende Erweiterung des orbis novem Systems: Eine Motion Controller der Firma Leap, wie er in der Abbildung 29 zu sehen ist.

Abbildung 29: Leap Motion Controller und Box

Darauf wird nun kurz eingegangen, wobei alle technischen Angaben nicht vom Projektteam validiert werden können, sondern [12] entnommen sind: Die Soft- und Hardwareplattform von Leap Motion bringt laut eigenen Angaben die bloßen Hände (ohne zusätzliche Hardware oder Handschuhe) direkt in die VR oder AR Anwendung. Die Leap-Motion-Technologie verfolgt dabei die Bewegung der Hände und sogar einzelner Finger. Der Leap Motion Controller verspricht hochgenaues Hand-Tracking mit extrem niedriger Latenz und ein unkompliziertes Setup: Nach dem Download der Leap Motion Software lässt sich das Gerät per USB-Anschluss anbinden. Zudem können Apps für Desktop und virtuelle Realität in der Leap Motion Gallery heruntergeladen werden. Leap bietet dazu ein SDK über eine API im C-Stil namens LeapC für den Zugriff auf Tracking-Daten aus dem Leap Motion-Service. Die Integrationen für die Unity und Unreal Engines basieren auf dieser API. So können Nutzer LeapC direkt in einem C-Programm verwenden, aber die Bibliothek ist in erster Linie für die Erstellung von Verknüpfungen zu übergeordneten Sprachen gedacht. Ältere Bindings für C++, C#, Java, JavaScript, Python und Objective-C bleiben verfügbar, werden aber nicht mehr aktiv unterstützt. Um den Einstieg zu erleichtern findet sich ein allgemeiner Überblick über die zur Verfügung gestellten Technologie- und Designrichtlinien, sowie die Unterschiede zwischen den

aktuell existierenden Softwareversionen V2, V3 und V4. [12]

Mit dem DeveloperKit können Rezipienten des orbis novem Systems also direkt mit den Händen in der Neue Welt 9 Umgebung arbeiten. Das DeveloperKit enthält dabei den Leap Motion Controller und die universelle VR-Entwicklerhalterung für die Datenbrille (siehe folgende Abbildung 30). Diese Halterung ist kompatibel mit, unter anderem, der HTC Vive und der vergleichbaren Datenbrille Oculus Rift. Der Sensor hat laut Hersteller 135-Grad-Sichtfeld um einen möglichst großen interaktiven Raum zu schaffen. [12]
Durch die Platzierung des Sensors auf der Datenbrille ist der Rezipient theoretisch nicht in der Exploration der VR-Umgebung gestört, unabhängig von seiner Gestik.
Auch das Abnehmen der Controller durch den Betreuer würde bei dieser Erweiterung entfallen, da keine Controller in der Hand getragen werden würden.

Abbildung 30: Leap Motion Controller und universelle VR-Entwicklerhalterung

Laut [12] ergeben sich bei der Nutzung die folgenden Systemanforderungen (Minimum System Requirements):

- Windows 7 SP1 64 bit or newer
- Intel® Core™ i5-4590 equivalent or greater
- 8GB+ RAM
- 3x USB 3.0 port
- NVIDIA GTX 970 / AMD R9 290 equivalent or greater with compatible HDMI 1.3 video output

6.3 Zukünftige Nutzung

In diesem Abschnitt erfolgt zuletzt ein Ausblick auf die zukünftige Nutzung. Zunächst ist zu erwähnen, dass Immersion ein komplexes Phänomen ist, das mehrere Ebenen neuropsychologischer Beteiligung wie Wahrnehmung, Aufmerksamkeit und Emotion erfordert. Die zugehörigen Mechanismen sind bei weitem noch nicht vollständig erforscht und

verstanden [7].
Das fertiggestellte orbis novem System bietet sich an, um als Experimentiergrundlage für verschiedene Forschungsprojekte im VR Labor zu fungieren. Dabei wäre zum Beispiel die Untersuchung von Virtual Reality Sickness, wie zuvor beschrieben, denkbar. Ebenso bietet es einen schnellen Einstieg in die verwendete Hardware. Das vollständig implementierte System ließe sich zudem für die Bildung von Studenten im Erst- oder Auslandssemester nutzen: Diese könnten sich zeitnah, ohne viel Aufwand und / oder sogar remote mit dem Gebäude 9 vertraut machen. Dazu müssten diese Studierenden nur die Immersion nutzen, um das Gebäude zu explorieren und sich mit dessen spezifischen, detailliert nachgebildeten Räumlichkeiten vertraut machen. Zuletzt bietet das System einen einfach zu benutzenden Proof-of-Concept an, den zukünftige Studierende relativ einfach erweitern können.
Die mit dieser Arbeit entstandene Dokumentation liefert eine Kommunikationsgrundlage. Diese kann genutzt werden, um mit anderen Stakeholdern des Systems wie zum Beispiel Lehrpersonal oder INF Service-Mitarbeiter charakteristische Merkmale des Systems einfacher darzulegen. Zudem bietet diese Dokumentation, neben der Wissensbasis für zukünftige Teilnehmer des VR Labors, auch einen ganz praktischen Nutzen: Wie die Abbildung 2 zeigt ist es für viele Schritte innerhalb der geführten Immersion dringend notwendig, einen Betreuer als Interaktionspartner zur Seite zu haben. Die explizite Darstellung dieses Vorganges kann damit als Anleitung genutzt werden um, beispielsweise auf Messen, Unfällen durch eine fehlerhafte Nutzung vorzubeugen.
Zudem soll diese Ausarbeitung als Dokumentation für Leser dienen, welche das System basierend auf der geleisteten Modellierung implementieren möchten. Diese Ausarbeitung kann zudem genutzt werden, um weitere systemische Betrachtungen mit der genutzten Vorgehensweise und einem Fokus auf verwandte Aspekte vorzubereiten.
Die Autoren der aktuellen Iteration würden sich sehr freuen, wenn das Projekt in Zukunft Anklang findet und Interessenten den Einstieg in die verwendeten Technologien erleichtern kann. Gleiches gilt, wenn das orbis novem System im Rahmen einer Bachelor-Theis, Master-Thesis oder von regulären Teilnehmern des Masterprojektes genutzt und weitergeführt werden würde.

6.4 Danksagung

Wir möchten uns an dieser Stelle ganz herzlich für die Unterstützung durch die Mitarbeiter des INF Service der Fakultät Informatik bedanken. Diese standen wirklich immer mit Rat und Tat zu Seite, haben für das VR Lab die ein oder andere Krise bewältigt und hätten zum Abschluss des Semesters (wie eigentlich in jedem Semester) einen Danke-Kuchen verdient ;).

6 Quellen

[1] Benjamin Batt, Claudiu Bräuer; 2018; Virtual Reality Laboratory, Dokumentation: iVR HUD; Hochschule Reutlingen, Reutlingen, DE.

[2] Staffan Björk, Sus Lundgren, Jussi Holopainen. 2004. Patterns In Game Design. Charles River Media. p. 206. ISBN 1-58450-354-8.

[3] Tiago Costa, Alberto Sampaio, Gustavo Ribeiro Alves. 2009. Using SysML in Systems Design, pp. 615–618. DOI: 10.1109/ICIII.2009.607.

[4] Thomas Grechenig, Mario Bernhart, Roland Breiteneder, Karin Kappel, 2009. Softwaretechnik. Mit Fallbeispielen aus realen Entwicklungsprojekten. München: Pearson Studium. ISBN 978-3-8632-6677

[5] Tim Weilkiens, 2014. Systems Engineering mit SysML/UML. Anforderungen, Analyse, Architektur. 3. Auflage, dpunkt, Heidelberg. ISBN: 978-3864900914

[6] Reinhard Höhn , Stephan Höppner, 2008. Das V-Modell XT. Grundlagen, Methodik und Anwendungen. eXamen.press. Springer-Verlag, Berlin Heidelberg. ISBN: 978-3540302490

[7] Chenyan Zhang, Sebastian Arndt, Andrew Perkis. 2017; Spatial immersion versus emotional immersion, which is more immersive? In QoMEX (Ed.): 2017 Ninth International Conference on Quality of Multimedia Experience (QoMEX). May 29-June 2, 2017, Erfurt, Germany. International Conference on Quality of Multimedia Experience; QoMEX. [Piscataway, NJ]: IEEE, pp. 1–6.

[8] Andrea Bondavalli, Sara Bouchenak, Hermann Kopetz 2016; Cyber-Physical Systems of Systems; Springer Verlag; 2016; ISBN 978-3-319-47590-5

[9] Ioanna Iacovides, Anna Cox, Richard Kennedy, Paul Cairns, Charlene Jennett, 2015. Removing the HUD: The impact of non-diegetic game elements and expertise on player involvement. In Annual Symposium on Computer-Human Interaction in Play (CHI PLAY '15) Conference; 2015

[10] Andreas Spillner, Tilo Linz. 2012. Basiswissen Softwaretest. Aus- und Weiterbildung zum Certified Tester; Foundation Level nach ISTQB-Standard. Safari Tech Books Online. dpunkt.verlag GmbH; 6. Auflage, Heidelberg. ISBN: 978-3864905834

[11] https://en.wikipedia.org/wiki/Likert_scale . Zuletzt aufgerufen am 07.07.2019

[12] https://developer.leapmotion.com/documentation. Zuletzt aufgerufen am 07.07.2019