# Analyse und Evaluierung von Vorgehensweisen zur Erstellung von Mixed Reality-Anwendungen

Diplomarbeit von

**Elham Allahyari-Jam**
Matrikelnummer: 1194902

September 2004

Johann Wolfgang Goethe-Universität Frankfurt am Main
Fachbereich Biologie und Informatik
Institut für Informatik
Professur für Graphische Datenverarbeitung

Eingereicht bei: Prof. Dr.-Ing. Detlef Krömker

Betreuer: Dipl.-Wirtsch.-Inform. Daniel F. Abawi

## Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass die vorliegende Diplomarbeit ohne unzulässige Hilfe und ausschließlich unter Verwendung der angegebenen Literatur angefertigt wurde.

*Frankfurt am Main, 28. September 2004*

Elham Allahyari-Jam

Das schönste Glück des denkenden Menschen ist,
das Erforschlische erforscht zu haben und
das Unerforschliche zu verehren.
*Johann Wolfgang von Goethe*

## Danksagung

An erster Stelle möchte ich mich bei meinem Betreuer Daniel F. Abawi bedanken, der mich in meinem Vorgehen unterstützt hat und mir durch seine hervorragende Betreuung immer mit Engagement und Rat durch viele Gespräche und Sitzungen zur Seite stand.

Mein spezieller Dank geht an Prof. Dr.-Ing. Detlef Krömker für die Bereitstellung der Ressourcen.

Ich bedanke mich auch für die nette Atmosphäre bei den Kollegen des AMIRE-Arbeitsteams: Sönke Dirksen, Michael Krauß, Björn Schmidt, André Miede, Joachim Bienwald und Martin Böss, die stets kompetente Ansprechpartner bei allen meinen Fragen waren.

Ganz herzlich bedanke ich mich bei meiner Familie für ihre Unterstützung und insbesondere bedanke ich mich bei meinem Bruder, Amir Mohammad Allahyari-Jam für seine Hilfe und Begleitung während meines Studiums.

Mein besonderer Dank gilt meinen Freunden: Majid Hamdouchi, Kambiz Assadi, Pouneh Khayat Pour und Bahareh Elahi. Sie haben mich immer motiviert, um die manchmal schweren Zeiten in den letzten Jahren bestmöglich zu verbringen.

Frankfurt am Main, 28. September 2004

Elham Allahyari-Jam

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Kapitel 1

# Einleitung

Dieses Kapitel gibt eine Einführung in das Thema dieser Arbeit. Zu Beginn wird die Motivation der Arbeit erklärt. Anschließend werden die Ziele beschrieben, die während dieser Arbeit erreicht werden sollen. Zum Schluss wird der Aufbau der Arbeit in Kurzfassung vorgestellt.

## 1.1 Motivation

Seit einigen Jahren existiert ein neuer Technologiezweig im Bereich Informatik, der eine Erweiterung der realen Welt durch eine virtuelle Welt ermöglicht. Bei dieser Erweiterung handelt es sich um Mixed Reality (MR), mit dessen Hilfe virtuelle Informationen wie 3D-Objekte, Graphiken, Texte, Audios und Videos in die Sicht eines Anwenders einblenden lassen. Durch die nahezu endlose Erweiterungsmöglichkeit der Realität und durch den Wegfall einer zeitlichen Verzögerung unterliegt die MR-Technologie fast keinen Einschränkungen bezogen auf ihr Einsatzgebiet. Allerdings bestehen zurzeit Probleme mit ihrer Komplexität. Diese Komplexität entsteht durch die unterschiedlichen Techniken, die bei der Entwicklung von MR-Anwendungen eingesetzt werden und dadurch den Produktionsumfang von MR-Anwendungen um eine meist unüberschaubaren Größe wachsen lassen. Zusätzlich dazu besteht eine MR-Anwendung meistens aus vielen verschiedenen multimedialen Objekten wie 3D-Modellen oder Videos, bei deren Erzeugung das Spezialwissen eines Designers, Regisseurs oder andere Experten unentbehrlich ist. Diese Experten besitzen meistens keine oder wenig Erfahrungen mit MR-Technologie und deren Implementierung. Daher sind sie bei der Erstellung von MR-Anwendungen auf das Spezialwissen von MR-Experten mit Programmierkenntnissen angewiesen. Diese Tatsache trägt einen erheblichen Beitrag zur Komplexität der Produktion und Wartung von MR-Anwendungen bei ([ZHHL03], [WGT$^+$02]).

Diese Komplexität wird ferner dadurch unterstützt, dass bisher kein Standardprozess für die Produktion einer MR-Anwendung existiert [ADG04]. Mit Standardprozess ist eine Vorgehensweise gemeint, die mittlerweile erfolgreich bei der Software-Technik eingesetzt wird. Die standardisierten Vorgehensmodelle zur Softwareentwicklung [Dro03] sind passable Beispiele dafür. Dieses Fehlen eines definierten Prozesses führt meistens

dazu, dass die Aufgabenbereiche der mitwirkenden Autoren einer MR-Anwendung nicht klar genug abgegrenzt werden und dies wiederum dazu, dass während der Entwicklung von MR-Anwendungen eine unregelmäßige und mit vielen Reibungsverlusten verbundene Kommunikation zwischen Autoren stattfindet. Es darf nicht unerwähnt bleiben, dass diese Problematik seit längerem bekannt ist und im Rahmen vieler Projekte eine Lösung dafür gesucht beziehungsweise entwickelt wird.

Zusammenfassend bestehen derzeit die größten Probleme bei der Entwicklung von MR-Anwendungen aus der Komplexität der Umsetzung, dem Vorraussetzen von Spezialwissen über MR-Technologie, dem schwer überschaubaren Umfang der einzusetzenden Techniken für die Entwicklung, dem Mangel an definierten Prozessen und der fehlenden klaren Trennung der Aufgabenbereiche mitwirkender Autoren einer MR-Anwendung.

Um diese Probleme zu beseitigen, sind Versuche unternommen worden, den Entwicklungsprozess von MR-Anwendungen zu vereinfachen. Ein bekannter favorisierter Lösungsansatz dafür ist der Einsatz von Autorensystemen, die in mehreren bekannten Bereichen der Informatik wie Multimedia und Entwicklung von Internet-Seiten erfolgreich verwendet werden. Bei Autorensystemen wird meistens ein Verfahren eingesetzt, das die Entwicklung von Anwendungen ohne Programmierung ermöglicht und jedem zugänglich macht. Zu diesem Zweck werden Autorensysteme für MR-Anwendungen entwickelt und eingesetzt, die genau dieses Ziel verfolgen. MR-Autorensysteme sollen die Aufgabenbereiche der mitwirkenden Autoren klarer abgrenzen und die Kommunikation zwischen ihnen erleichtern. Des Weiteren sollen MR-Autorensysteme die Definition eines Prozesses für die Erstellung von MR-Anwendungen ermöglichen. Inwieweit die Versuche der Vereinfachung dieses Entwicklungsprozesses gelungen sind, kann noch nicht definitiv beantwortet werden.

## 1.2 Zielsetzung der Arbeit

Die im vorigen Unterkapitel genannten Probleme, die zur Komplexität der Erstellung von MR-Anwendungen beitragen, werden in verschiedenen Projekten aus dem Bereich der Forschung und Industrie intensiv behandelt. Einige dieser Projekte haben mittlerweile konkrete Lösungsansätze, die ihrerseits eine Vereinfachung des MR-Erstellungsprozesses beanspruchen. Aber inwieweit ist diese Vereinfachung tatsächlich gelungen?

Des Weiteren existiert die Frage, ob sich MR-Anwendungen ähnlich wie Standardsoftware im Sinne der Software-Technik erstellen lassen. Lassen sich standardisierte Vorgehensmodelle zur Softwareentwicklung, wie das Wasserfallmodell, Evolutionäres Modell oder V-Modell ([Dro03], [Bal00]), die in verschiedenen Phasen wie Anforde-

rungsanalyse, Spezifikation, Realisierung und Validierung unterteilt sind, auf den MR-Erstellungsprozess übertragen?

Diese Arbeit geht den oben genannten Fragestellungen nach und setzt sich zum Hauptziel, eine Antwort auf diese Fragen zu finden.

Ein weiteres Ziel, was im Laufe dieser Arbeit verfolgt wird, ist die Erstellung einer typischen MR-Anwendung. Sie gilt als typisch, sofern sie unabhängig von ihrer Erstellungsmethode spezifiziert wird. Sie soll weiterhin repräsentativ für MR-Anwendungen sein, um als eine gute Vorlage für weitere Entwicklungen eingesetzt werden zu können. Sie gilt als eine repräsentative MR-Anwendung, wenn sie häufig benötigte Funktionen wie Orientierungshilfe und Benutzerführung beinhaltet.

Diese typische MR-Anwendung ist eine willkommene Gelegenheit, um zu überprüfen, ob ein standardisiertes Vorgehensmodell auf den MR-Erstellungsprozess übertragbar ist, indem sie auf Basis eines solchen Vorgehensmodells, wie das Wasserfallmodell, erstellt wird. Man spricht vom Wasserfallmodell, wenn die Aktivitäten in einem Projekt mit einem klar definierten Ereignis - einem so genannten Meilenstein - enden und jeder von ihnen beendet werden muss, bevor der nächste beginnt [Dro03]. Wenn die typische MR-Anwendung in den vier Phasen Anforderungsanalyse, Spezifikation, Realisierung und Validierung umgesetzt wird, könnte bei ihrer Umsetzung ähnlich wie beim Wasserfallmodell vorgegangen werden. Die gelungene Realisierung würde in diesem Fall die Übertragbarkeit vom Wasserfallmodell auf die Erstellung von MR-Anwendungen
bestätigen.

## 1.3 Gliederung der Arbeit

Im Folgenden werden die Inhalte der nachfolgenden Kapitel vorgestellt, um den Lesern einen Überblick über den Aufbau der gesamten Arbeit zu geben.

**Kapitel 2:**

Dieses Kapitel beinhaltet einige Grundlagen, die für das Verständnis des Gesamtthemas hilfreich sind. Es werden Begriffe wie Mixed Reality (MR), Virtual Reality (VR), Augmented Reality (AR) und Autorensysteme erläutert. Ferner werden allgemeine Begriffe wie Spezifikation oder Anforderungsanalyse erklärt und einige Anwendungsbereiche für MR beispielhaft genannt.

**Kapitel 3:**

Kapitel 3 analysiert die MR-Autorensysteme. Dazu werden zunächst Anforderungen an MR-Autorensysteme definiert. Diese werden in zwei Gruppen aufgeteilt, obligatorische und fakultative Anforderungen. Anschließend werden verschiedene MR-Autorensysteme vorgestellt und bewertet. Basierend auf dieser Bewertung wird ein MR-Autorensystem ausgesucht.

**Kapitel 4:**

Dieses Kapitel beschäftigt sich mit der Spezifikation einer typischen MR-Anwendung. Sie wird unabhängig von der Erstellungsmethode spezifiziert, ohne auf ihre Realisierung einzugehen. Dabei werden Ziele der MR-Anwendung definiert, ihre Funktionen beschrieben, die Anwendungsfälle erläutert und das Design für die Oberfläche festgelegt.

**Kapitel 5:**

Im Kapitel 5 wird eine Überführung der Spezifikation aus Kapitel 4 auf das ausgewählte MR-Autorensystem durchgeführt. Dabei wird beschrieben, wie die spezifizierte MR-Anwendung mit Hilfe des ausgesuchten MR-Autorensystems aus Kapitel 3 erstellt werden kann.

**Kapitel 6:**

In diesem Kapitel wird basierend auf der Spezifikation aus Kapitel 4 die MR-Anwendung realisiert. Zunächst werden die Benutzeroberfläche des ausgesuchten MR-Autorensystem aus Kapitel 3 und die Werkzeuge beschrieben. Anschließend werden die Vorgehensweise für die Umsetzung der Anwendung, die gesammelten Erfahrungen während der Produktion, die Testszenarien und Dokumentation der MR-Anwendung beispielhaft vorgestellt. Des Weiteren werden hier Verbesserungsvorschläge aufgelistet.

**Kapitel 7:**

In diesem Kapitel wird die Arbeit hinsichtlich ihrer Vorgehensweise und der Ergebnisse evaluiert.

**Kapitel 8:**

Dieses Kapitel beendet die Arbeit durch die Zusammenfassung aller erzielten Ergebnisse und durch einen zusätzlichen Ausblick auf Erweiterungsmöglichkeiten.

# Kapitel 2

# Grundlagen

Dieses Kapitel dient dazu, Lesern mit wenigen Erfahrungen bezüglich der Mixed Reality-Technologie einen leichteren Einstieg in das Thema der Arbeit zu verschaffen. Dazu werden im ersten Abschnitt Begriffe erläutert, die in der Arbeit verwendet werden. Im zweiten Abschnitt werden computergestützte und -generierte Realitäten wie Virtual Reality, Augmented Reality und Mixed Reality als Begriffe dargelegt. Im darauf folgenden Abschnitt werden die Anwendungsbereiche der Mixed Reality beispielhaft erwähnt. Am Ende dieses Kapitels werden Autorensysteme im Allgemeinen und die verschiedenen Arten von Autorensystemen vorgestellt.

## 2.1 Terminologie

Im Folgenden werden alle wichtigen Begriffe definiert und erläutert, die in den nächsten Kapiteln häufig benutzt werden und das Verständnis der Themen dieser Arbeit erleichtern.

**Definition 1: Anforderung**

- Eine Anforderung ist eine Bedingung oder Fähigkeit, die von einer Person zur Lösung eines Problems oder zur Erreichung eines Ziels benötigt wird [Gli02].

- Anforderungen an ein Softwaresystem legen fest, was das System leisten soll und definieren Einschränkungen seiner Funktionen und Implementierung [Som01].

**Definition 2: Anforderungsanalyse**

- Die Beschreibungen der Dienste und Einschränkungen sind die Anforderungen an das System. Der Prozess des Herausfindens, Dokumentierens und Überprüfens dieser Dienste und Beschränkungen wird Anforderungsanalyse genannt [Som01].

**Definition 3: Spezifikation**

- Eine Spezifikation ist ein Text, der die Syntax und die Semantik eines bestimmten Bestandteiles beschreibt beziehungsweise eine deklarative Beschreibung, was etwas ist oder tut [GBB00].

- Im Allgemeinen kann eine Spezifikation als ein Abkommen zwischen Hersteller einer Dienstleistung und Verbraucher dieser Dienstleistung angesehen werden [GJM91].

**Definition 4: Anforderungsspezifikation**

- Hier handelt es sich um die Zusammenstellung aller Anforderungen an einer Software [Gli02].

**Definition 5: Softwarespezifikation**

- Die Softwarespezifikation soll festlegen, welche Funktionen von dem System verlangt werden und welchen Beschränkungen der Betrieb und die Entwicklung des Systems unterliegen [Som01].

**Definition 6: Mixed Reality-Autorenprozess**

- Ein Mixed Reality-Autorenprozess spezifiziert, wie eine Mixed Reality-Anwendung erstellt wird und welche Autorenrollen mit welchen Aufgaben und an welchen Schnittstellen zusammenarbeiten. Der Mixed Reality-Autorenprozess leitet die Autoren bei der Erfüllung der ihm zugewiesenen Aufgaben [ADG04].

## 2.2 Mixed Reality

Um Mixed Reality (MR) besser verstehen zu können, ist es hilfreich zunächst herauszufinden, was Virtual Reality (VR) ist. VR ist eine künstliche Welt, die durch die Computertechnologie entsteht. Sie vermittelt dem Anwender den Eindruck, ein Teil dieser Welt zu sein, sich in ihr bewegen und mit ihren Objekten in Echtzeit interagieren zu können [Hal02b]. Die erzeugte virtuelle Welt wird dann von Anwendern als wahr und realistisch empfunden, wenn der Anwender wie in der Realität mit den virtuellen Objekten interagieren kann. Noch besser wird diese Realitätsempfindung, wenn der Anwender zusätzlich durch akustische Sinnesreize in der künstlichen Welt kommunizieren kann.

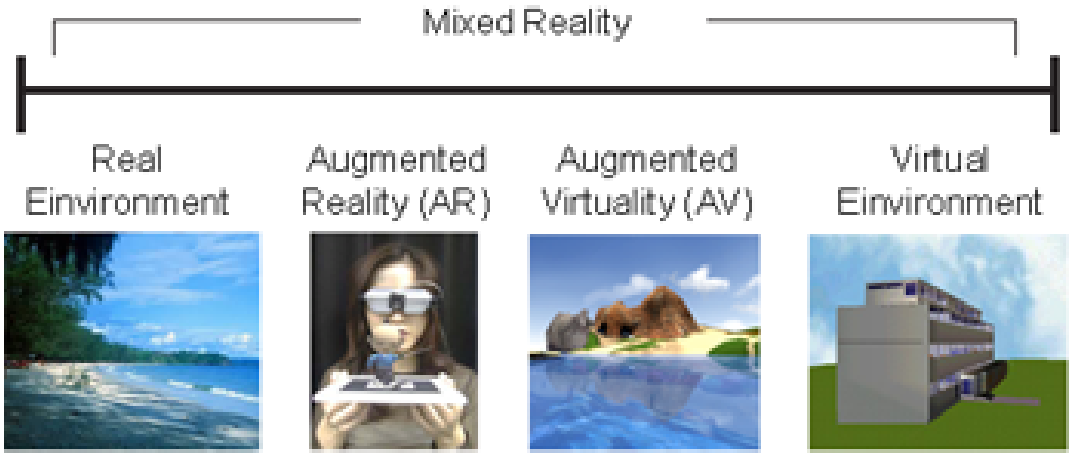


Abb. 2.1: Reality-Virtuality Continuum von Milgram (vgl. [MK94])

Anders als VR, die den Benutzer in die künstliche Welt aufnehmen und ihn vergessen lassen will, dass er ein Teil der realen Welt ist, versucht Augmented Reality (AR) dem Benutzer die Möglichkeit zu geben, die reale Welt überlagert oder vermischt mit virtuellen Objekten wahrzunehmen [Azu97]. Das heißt, in VR wird die reale Welt komplett durch eine virtuelle ersetzt, während in AR diese durch die virtuelle Welt erweitert wird. Azuma definiert AR-Systeme mit Hilfe von drei charakteristischen Eigenschaften wie folgt [ABB+01]:

- Kombination von realen und virtuellen Objekten in einer realen Umgebung
- Echtzeit-Interaktivität
- Ausrichtung von realen und virtuellen Objekten aufeinander

Diese Eigenschaften helfen den Anwendern, zusätzliche Informationen zur realen Umgebung zu bekommen, welche mit ihren Sinnen nicht oder nur schwer erfasst werden.

1994 haben Milgram und Kishino [MK94] das so genannte „Reality-Virtuality Continuum“ vorgestellt, in dem definiert wird, wie Realität mit der Virtualität vermischt werden kann. Ihr Ziel dabei war es, eine konsistente Definition der genannten Begriffe zu schaffen. In Abbildung 2.1 wird das „Reality-Virtuality Continuum“ dargestellt. Die linke Seite der Abbildung zeigt die Realität und die rechte Seite die Virtualität. MR ist der Bereich dazwischen, in dem die Mischung aus realen und virtuellen Objekten stattfindet. Wie aus dem Dokument von Abawi [ADG04] zu entnehmen ist, ist MR die gezielte Überblendung von realen Bildern mit computergenerierten Informationen. Reale Bilder sind zum Beispiel Videoaufnahmen oder Echtzeitkameraaufnahmen. Bei den computergenerierten Informationen handelt es sich zum

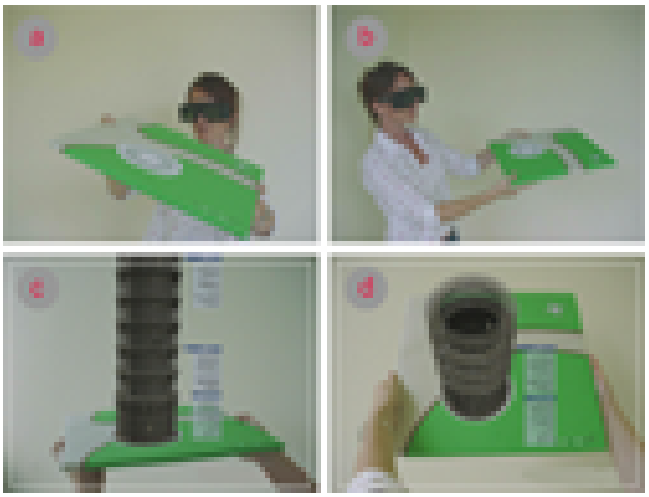

Abb. 2.2: Beispiel einer MR-Anwendung [ADG04]

Beispiel um Animationen, Texte oder Bemerkungen. Wie in Abbildung 2.1 zu sehen ist, tauchen hier noch zwei weitere Begriffe auf: AR, in der virtuelle Objekte zur realen Welt hinzugefügt werden und Augmented Virtuality (AV), in der reale Objekte zur virtuellen Welt hinzugefügt werden. Auf der rechten Seite der Abbildung ist die Umgebung virtuell, während auf der linken Seite die Umgebung real ist. In der Literatur werden die Begriffe MR und AR meistens als sinnverwandt verwendet, obwohl MR der Überbegriff von AR ist.

Abbildung 2.2 stellt ein Beispiel für die Anwendung dieser Technologie dar. Hier wird ein Benutzer gezeigt, der ein virtuelles Objekt in seiner realen Umgebung integriert sieht. Das Modell des schiefen Turms von Pisa als virtuelles Objekt scheint sich auf der realen Grundplatte zu befinden, die der Benutzer in der Hand hält. Abbildungen a und b zeigen, wie der Benutzer die Grundplatte bewegt, während in Abbildungen c und d die Situation aus der Sicht des Benutzers dargestellt wird.

Damit ein Anwender die MR-Technologie anwenden kann, benötigt er ein Gerät, das die Realität mit virtuellen Objekten kombiniert anzeigen kann. Für den Einsatz dieser Technologie werden verschiedene Ausgabegeräte verwendet, die hier näher betrachtet werden. Als Ausgabegeräte kommen einerseits so genannte „Head Worn Displays“ (HWDs), die auch unter „Head Mounted Displays“ (HMDs) bekannt sind, und andererseits herkömmliche und projektionsbasierte Displays zum Einsatz. Der Hauptunterschied liegt darin, dass die projektionsbasierten und herkömmlichen Displays nicht am Kopf des Benutzers befestigt sind, während sich die Anzeige bei HWD direkt vor den Augen des Benutzers befindet. Die HWDs sind wiederum in zwei Kategorien aufzuteilen. Die erste Kategorie ist die so genannte „Optical see-through HMD“ und die andere „Video see-through HMD“. Die „Optical see-through HMDs“

funktionieren so, dass ein halbdurchlässiger Spiegel direkt vor die Augen des Benutzers angebracht wird, so dass er durch sie durchsehen kann und seine reale Umwelt wie gewohnt wahrnehmen kann. Dabei werden die virtuellen Objekte auf den Spiegel projiziert und der Effekt der MR hervorgehoben. Abbildung 2.3 stellt dieses Konzept graphisch dar und Abbildung 2.4 zeigt ein Beispiel für ein solches HMD.

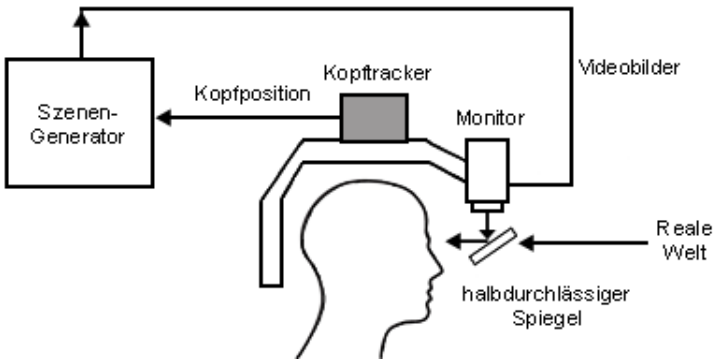


Abb. 2.3: Aufbau eines Optical see-through HMDs (vgl. [Azu97])

Abb. 2.4: Beispiel für Optical see-through HMDs [Sch04]

Die „Video see-through HMDs“ benutzen ein oder zwei am Kopf angebrachte Videokameras, die die reale Welt aufnehmen und auf die Monitore vor den Augen des Benutzers projizieren. Vor der Wiedergabe wird durch eine Echtzeitbearbeitung der Bilder die reale Welt mit virtuellen Objekten vervollständigt. Abbildung 2.5 visualisiert dieses Konzept und Abbildung 2.6 präsentiert ein Beispiel für ein solches HMD.

Herkömmliche Displays funktionieren ähnlich wie „Video see-through HMDs“, nur mit dem Unterschied, dass die Anzeige nicht direkt vor den Augen des Benutzers, sondern auf einem tragbaren oder fixierten Display wiedergegeben werden kann. Abbildung 2.7

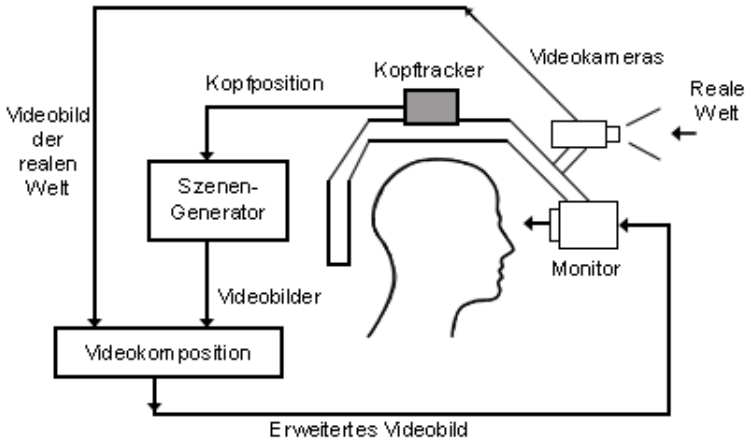


Abb. 2.5: Aufbau eines Video see-through HMDs (vgl. [Azu97])

Abb. 2.6: Beispiel für Video see-through HMDs [Sch04]

präsentiert dieses Konzept und Abbildung 2.8 zeigt ein Beispiel für monitorbasierte Displays.

Projektionsbasierte Anzeigegeräte bilden die virtuellen Objekte auf reale ab. Dies funktioniert folgendermaßen: Die realen Objekte werden mit einer retroreflektierenden oder hell gefärbten Oberfläche beschichtet und durch einen oder mehrere Projektoren mit virtuellen Objekten beleuchtet. Abbildung 2.9 stellt das Prinzip dieser Technologie graphisch dar und Abbildung 2.10 zeigt ein Beispiel dazu. Dabei stellen die realen Objekte, wie auf der linken Seite zu sehen ist, die Form des abgebildeten Stadtmodells dar. Sie werden durch die Bestrahlung mit dem Projektor mittels Anwendung von Farben und Mustern erweitert [RL01].

Details und technische Hintergründe, die diese Ausgabegeräte betreffen, können den Dokumenten [Azu97], [ABB+01] und [RL01] entnommen werden.

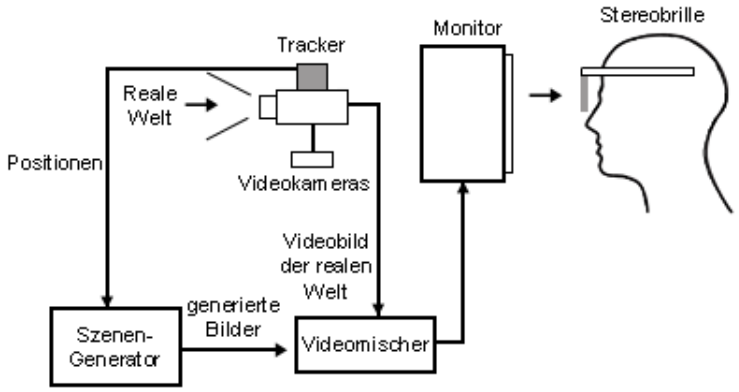


Abb. 2.7: Aufbau eines monitorbasierten Displays (vgl. [Azu97])

Abb. 2.8: Beispiel für monitorbasierte Displays [Azu97]

Damit in der MR-Anwendung die virtuellen Objekte abhängig von realen Objekten positioniert werden können, werden so genannte „Trackingsysteme“ eingesetzt. Es gibt zwei verschiedene Arten dieser Systeme: markerlose und markerbasierte. Markerlose Systeme eignen sich für Outdoor-Anwendungen und markerbasierte für Indoor.

Für diese Arbeit sind markerbasierte Trackingsysteme relevant, weil die in dieser Arbeit zu realisierende MR-Anwendung eine Indoor-Anwendung ist. Deswegen muss das zum Einsatz kommende MR-Autorensystem dementsprechend ein markerbasiertes Trackingsystem einsetzen. Bei solchen Trackingsystemen werden Marker in eine vorliegende Szene platziert, die als Referenzpunkte zwischen realer und virtueller Umgebung dienen und von einer Kamera erkannt und verfolgt werden. Dadurch wird die Position des Markers in Echtzeit berechnet und mit virtuellen Objekten verbunden. Von großer Bedeutung ist, dass die Form der Marker bei jedem System unterschiedlich sein kann, aber die Marker an sich eindeutig erkennbar sein müssen. Dadurch ist es dem System möglich, Berechnungsfehler bei der Zuordnung der virtuellen Objekte zu ihrem Marker zu vermeiden. Bei manchen Systemen werden verschiedene Kreise mit bunten Farben [NYC+99] oder Strichcodes [RN95] eingesetzt.

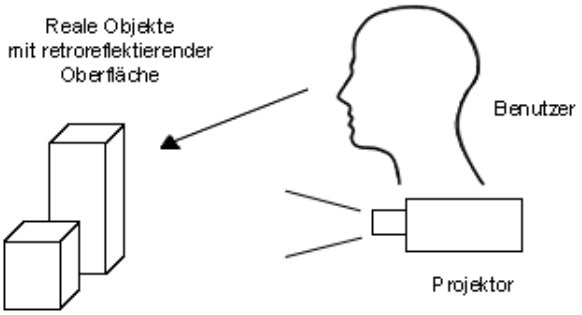


Abb. 2.9: Projektionsbasierte AR (vgl. [Sch04])

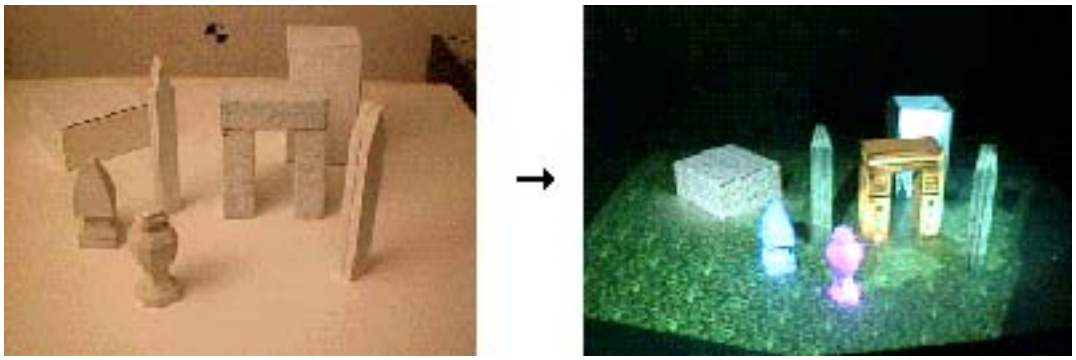

Abb. 2.10: Beispiel für projektionsbasierte Displays [RL01]

Beim „ARToolKit“ [KBP00] kommen wiederum schwarzweiße Quadrate als Marker zum Einsatz. Abbildung 2.11 zeigt ein Beispiel für Marker, die beim „ARToolKit“ verwendet werden. „ARToolKit“ ist eine C-Bibliothek, mit der MR-Anwendungen einfach erstellt werden können.

Im Abbildung 2.12 wird der Einsatz von markerbasierten Trackingsystemen anhand von zwei Beispielen graphisch dargestellt. Die linke Seite der Graphik stellt die Sichtweise eines Benutzers dar, der einen Marker in der Hand hält. Dabei wird der Marker von einem AR-System erkannt und durch eine virtuelle Geometrie erweitert. Die rechte Seite der Abbildung zeigt ein Beispiel vom Projekt „MusicAR“ [Hau02], das ein musikalisches Spiel mit Virtuellen Erweiterungen für Kinder entwickelt und „ARToolKit“ als Trackingsystem einsetzt.


Abb. 2.11: Beispiele für ARToolKit-Marker (vgl. [KBP00])

Abb. 2.12: Marker mit überlagerter Geometrie ([KBP00],[Hal02a])

## 2.3 Anwendungsbereiche

Wenn man sich etwas mit der MR-Technologie auseinandersetzt, ist gut zu erkennen, dass sie ein breites Spektrum an Einsatzgebieten abdecken kann. Um dem Leser eine Vorstellung davon zu geben, werden hier einige dieser Einsatzgebiete aufgelistet.

#### Film und Fernsehen

In diesem Bereich kann MR vielseitig eingesetzt werden. Beispielsweise können Live-Übertragungen von Sportereignissen mit computergenerierten Informationen erweitert und so für die Zuschauer informativer gestaltet werden, wie in Abbildung 2.13 zu sehen ist. Diese Objekte können zum Beispiel virtuelle Werbebanner sein. Sie können auch Informationen wie die Entfernung der Mauer beim Freistoß in einem Fußballspiel oder die Nationalflaggen in Schwimmbahnen anzeigen, wie damals in der Olympiade 2000 in Sydney. Auch in der Filmindustrie kann die MR-Technologie vorteilhaft eingesetzt werden. Zum Beispiel können Spezialeffekte bei der Produktion von Filmen und Videos eingesetzt werden.


Abb. 2.13: Fernsehübertragung eines Automobilrennens [ABB+01]

### Bibliotheken

Um die Regale einer Bibliothek mit aktuellen Informationen über deren Inhalt zu erweitern, kann eine MR-Anwendung zum Einsatz kommen. Beispielsweise können Informationen über vorhandene oder ausgeliehene Bücher ganz aktuell dem Benutzer zur Verfügung gestellt werden. In Abbildung 2.14 wird eine solche MR-Anwendung gezeigt, die in einem so genannten „augmented library“ eingesetzt wird. Dabei hilft das System dem Benutzer, ein Buch zu finden oder spezielle Fragen über Bücher zu beantworten. Nähere Details zu dieser MR-Anwendung können dem Dokument [RN95] entnommen werden.

Abb. 2.14: Beispiel für eine so genannte augmented library [RN95]

### Museen

In vielen Museen existieren bereits elektronische auditive Führer, die den Besuchern Zusatzinformationen zu den Ausstellungsobjekten zur Verfügung stellen. Diese Führer sind jedoch auf auditive Fähigkeiten beschränkt. Durch eine MR-Anwendung würden diese Führer um eine Vielzahl an Möglichkeiten erweitert werden können. Es können beispielsweise Videos zum Herstellungsprozess eines Kunstwerkes oder textbasierte Informationen über den Künstler verwendet werden. Außerdem können Navigationsinformationen eingesetzt werden, um in einem großen Museum den Benutzer zu seinem Ziel zu führen. In Abbildung 2.15 wird gezeigt, wie in einem Museum ein realer Dinosaurierschädel durch dreidimensionale und virtuelle Gewebeschichten Schritt für Schritt ergänzt wird [TBE03]. Auf der linken Seite der Abbildung werden die Knochen des Dinosaurierschädels mit Muskeln und auf der rechten Seite mit Haut überlagert.

Abb. 2.15: Einsatz der MR-Technologie in einem Museum [RAP04]

### Montage und Service

In vielen Industriezweigen ist es sehr wichtig, dass die Mitarbeiter in ihrem Arbeitsbereich geschult werden. Dies steht meistens in Bereichen von Montage, Wartung und Reparatur auf der Tagesordnung. Bei diesen Schulungen könnte eine MR-Anwendung den Mitarbeitern eine effektivere und einfachere Weiterbildung ermöglichen. Wenn in einer Fahrzeugfabrik zum Beispiel ein neues Modell eingeführt wird, können sich die Mitarbeiter während der Schulung durch Video-, Text- und Audio-Anweisungen die Montage einfach und übersichtlich anzeigen lassen. Dadurch ist es einfacher, eine Vorstellung von der noch zu leistenden Arbeit zu bekommen, was bisher nur durch komplizierte Handbücher möglich war. Dasselbe Prinzip gilt auch für die Industriezweige in Luft- und Raumfahrt sowie Maschinenbau. Steve Feiner's group in Columbia-Universität hat eine MR-Wartungsanwendung für Laserdrucker entwickelt [FMS93], die den Technikern während der Reparatur Anweisungen zur Vorgehensweise erteilt. Im linken Teil der Abbildung 2.16 wird ein Techniker gezeigt, der diese Anwendung benutzt

und im rechten Teil der Abbildung wird sein Sichtfeld visualisiert. Dabei wird anhand von virtuellen Objekten gezeigt, wie das Papierfach entfernt werden kann [Azu97].

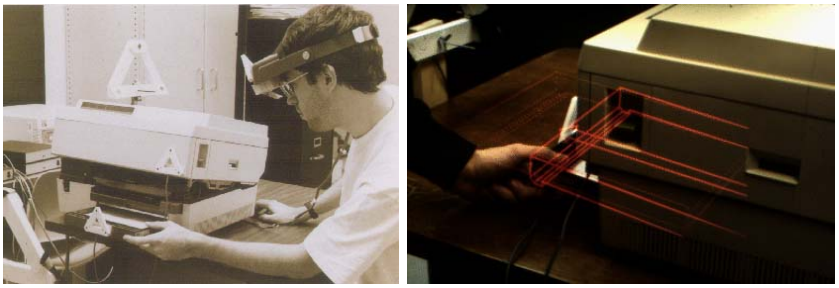

Abb. 2.16: Reparaturanleitung mit Hilfe von MR-Technologie [FMS93]

**Medizin**

Eines der interessantesten Einsatzgebiete für MR ist der medizinische Bereich. Einem Chirurgen wird zum Beispiel durch diese Technologie ermöglicht, stets mit neuesten in sein Blickfeld eingeblendeten Informationen versorgt zu werden. Diese Informationen können wichtige Parameter wie der aktuelle Blutdruck, Sauerstoffsättigung, Pulsfrequenz oder Röntgenbilder sein, die während einer Operation ständig beobachtet werden müssen. Diese Informationen können auf den Patienten projiziert werden. Zwei bekannte Projekte, die MR in medizinischen Bereichen einsetzt, sind ARSyS-Tricorder [ARS04] und Medarpa [MED04].

Der ARSyS-Tricorder ist ein System, welches in der Operationstechnik während eines chirurgischen Eingriffs den dreidimensionalen Operations- und Planungsweg auf dem Patienten einblenden soll. Der Hauptfokus dieses Projektes liegt auf der Unterstützung der Mund-Kiefer-Gesichtschirurgie. Beim ARSyS-Tricorder werden während der Operation anatomische Strukturen und Operationswege auf den Patienten dreidimensional über eine Anzeige eingeblendet. Dadurch erhält der Chirurg Informationen, die für die Operation von großer Wichtigkeit sind. Die Operation wird durch den zusätzlichen Einsatz einer MR-Anwendung vereinfacht. Es kann zum Beispiel der Verlauf einer Nervenbahn oder eines Blutgefäßes je nach Bedarf angezeigt werden [GTB+04]. Abbildung 2.17 zeigt das ARSyS-Tricorder Display System.


Abb. 2.17: ARSyS-Tricorder Display System ([GTB+04], [ARS04])

Das Medarpa-Forschungsprojekt [MED04] verfolgt ähnlich wie das ARSyS-Tricorder-Projekt das Ziel, den Chirurgen während der Operation zu unterstützen, indem dem Chirurgen Bilddaten des Patienten direkt in sein Sichtfeld eingeblendet werden. Dabei kommt hier ein transparentes Display zum Einsatz, welches patientenbezogene Daten im direkten Sichtfeld des Chirurgen anzeigen kann. Das hat den Vorteil, dass der Chirurg während der Operation seinen Blick nicht von der Operationsstelle abwenden muss [SRSS04]. Wie eine solche Operation mit Hilfe von Medarpa aussehen könnte, zeigt die Abbildung 2.18.

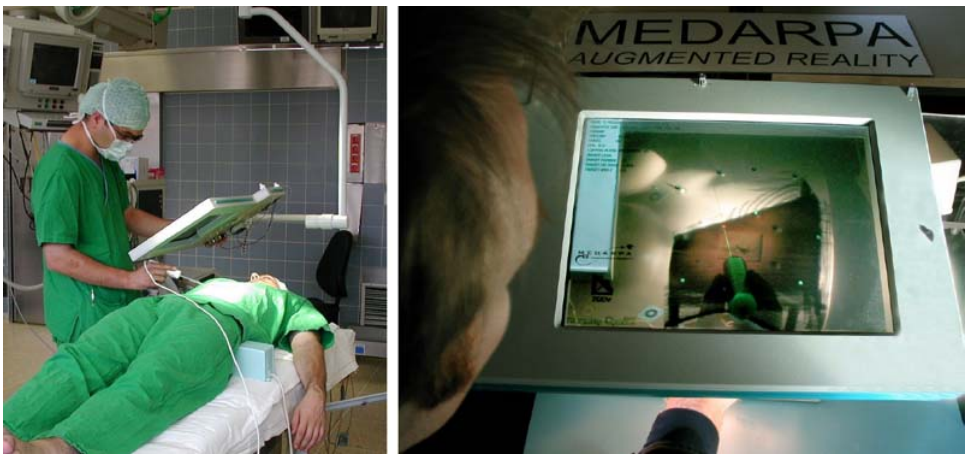


Abb. 2.18: Einsatz des MEDARPA-Systems [SRSS04]

### Militär

Seit einigen Jahren werden im Militärbereich Technologien eingesetzt, die der MR-Technologie sehr ähnlich sind. Piloten von Kampfhubschraubern und Kampfjets tragen beispielsweise so genannte „Head Up Displays“ (HUDs) und „Helm Mounted Sights“ (HMSs) [Azu97]. Mit Hilfe dieser Geräte können taktische Informationen und Maschinendaten visualisiert werden. Diese Visualisierung kann durch MR-Technologie mit der realen Umgebung in Verbindung gebracht werden und kann somit die Sicht der Piloten um detaillierte Informationen erweitern. Zum Beispiel können virtuelle Rollbahn-Markierungen oder ein Ausgleich von schlechten Sichtverhältnissen direkt in das Blickfeld des Piloten eingeblendet werden. Für den Einsatz der Bodentruppen finden ähnlichen Entwicklungen statt. In Abbildung 2.19 wird ein Soldat gezeigt, der im Rahmen des Projektes „Land-Warrior“ [LW04] der US-Armee ein HMD trägt, auf dem zusätzliche Informationen zum Einsatzgebiet angezeigt werden können. Diese Informationen können eine Karte vom Einsatzgebiet mit Anzeige der Position von feindlichen Zielen und freundlichen Einheiten sein.

Abb. 2.19: Einsatz der MR-Technologie beim Militär [LW04]

### Archäologie

Durch MR wird dem Anwender ermöglicht, während archäologischen Betrachtungen sichtbare Objekte durch konstruierte virtuelle Objekte zu ergänzen. Dadurch wird eine bessere Vorstellung von dem fertigen Objekt geschaffen. ARCHEOGUIDE (Augmented Reality-based Cultural HEritage On-site GUIDE) [ARC02] ist ein bekanntes Projekt, das die Rekonstruktion von historischen Ruinen vor den Augen der Betrachter durch die MR-Technologie ermöglicht. Ein Beispiel für eine solche Rekonstruktion zeigt Abbildung 2.20. Darin wird der Hera-Tempel im alten Olympia im Griechenland virtuell rekonstruiert und in das Blickfeld des Betrachters eingeblendet. Das linke Bild

zeigt die reale Umgebung und das rechte Bild stellt die erweiterte Realität aus Sicht des Benutzers dar.

Abb. 2.20: Einsatz der MR-Technologie in der Archäologie [VKT+01]

### Spiele

In der Spielindustrie kann MR eine große Rolle spielen. Es könnte zum Beispiel ein altes klassisches Spiel wie Pacman so entwickelt werden, dass die Spielfiguren Pacman und die Geister menschliche Spieler wären. Das Spielfeld wäre dabei ein reales Labyrinth, und die zu sammelnden Kugeln wären virtuelle Objekte. Das Human Pacman-Projekt [CFG+04], das dieses Spiel in der beschriebenen Form umsetzen soll, wird zurzeit in der Universität von Singapur entwickelt. Abbildung 2.21 zeigt, wie dieses Spiel aussehen könnte.

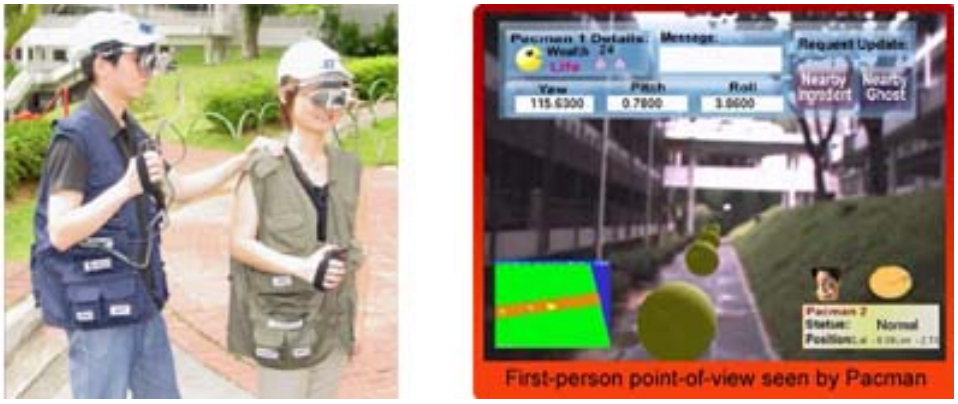


Abb. 2.21: Human Pacman-Projekt [HP04]

Ein weiteres Spiel wird im Projekt GEIST [GEI04] realisiert. Es geht dabei um ein interaktives Lernspiel, mit dessen Hilfe die Geschichte von Heidelberg zurzeit des 30jährigen Krieges spielerisch erlebt werden kann. Während des Spiels können sich die Spieler frei in der Stadt bewegen und dabei verschiedene Aufgaben lösen. Während dessen treffen sie verschiedene Geister, die ihnen bei der Lösung ihrer Aufgaben helfen. Dieses Projekt verfolgt das Ziel, die Geschichte der Stadt Heidelberg den Besuchern und den Bewohnern der Stadt spielerisch näher zu bringen. Abbildung 2.22 zeigt einen Anwender beim Test der Anwendung.

Abb. 2.22: GEIST-Projekt [HS04]

Weitere Details zu diesen und andere Einsatzbereiche von MR-Technologie können den Vorlesungsunterlagen von Krömker [Krö04] und Dokumenten von Azuma ([Azu97], [ABB+01]) entnommen werden.

## 2.4 Autorensysteme

Ein Kernpunkt dieser Diplomarbeit handelt von so genannten MR-Autorensystemen. Was aber sind Autorensysteme und seit wann existieren sie? Die Entwicklung von Autorensystemen hat bereits in den 80er Jahren begonnen. Für die 90er Jahre war es entscheidend, neue Technologien auch mit Hilfe von Software steuern zu können [RKHF02].

Autorensysteme sind meistens graphisch interaktive Werkzeuge zur Erstellung von Multimedia-Inhalten. Diese erlauben die Erstellung von Multimedia-Dokumenten, welche bei den Endbenutzern mehr oder weniger interaktiv dargestellt werden [RP02]. Dabei soll die Komplexität des Systems für den Autor unbemerkbar bleiben, das heißt mit Hilfe von Autorensystemen werden auch Nichtprogrammierer in der Lage sein, An-

wendungen zu erstellen.

Die Verwendung von Autorensystemen hat nicht nur wirtschaftliche Vorteile. Für die Erstellung der Anwendung wird unter anderem wenig Spezialwissen benötigt und die Autoren können sich intensiver mit der Gestaltung der Benutzerschnittstelle der Anwendung befassen. In einem Autorensystem können verschiedene Informationsobjekte, wie zum Beispiel Texte, Graphiken, Animationen, Audio- oder Videodateien in einer graphischen Benutzeroberfläche miteinander verbunden werden. Dadurch wird der Herstellungsprozess einer Anwendung vereinfacht. In der Regel werden diese Informationsobjekte mit unterschiedlichen Werkzeugen erstellt und später in das Autorensystem integriert. Die Hauptaufgaben des Autorensystems liegen darin, die Informationsobjekte miteinander zu verbinden und interaktive Elemente zu erstellen [Mül01]. Die chronologischen Beziehungen dieser Objekte legen dabei den Ablauf der Anwendung fest. Hinsichtlich der Art und Weise, wie diese Beziehungen zwischen Informationsobjekte existieren, können die Autorensysteme in drei Klassen aufgeteilt werden [Bol95b]:

- timeline-basierte Autorensysteme
- frame-basierte Autorensysteme
- flowchart-basierte Autorensysteme

Im Folgenden werden drei weit verbreitete Autorensysteme erwähnt, die als Vertreter dieser Klassen rudimentär vorgestellt werden, um eine etwas genauere Vorstellung von Autorensystemen zu vermitteln: Macromedia Flash [Fla04], Macromedia Authorware [Aut04] und Microsoft PowerPoint [Pow04]. MR-Autorensysteme sind speziell für die Erstellung von MR-Anwendungen konzipiert und unterscheiden sich prinzipiell wenig von anderen Autorensystemen.

#### Timeline-basierte Autorensysteme

In timeline-basierten Autorensystemen werden die Informationsobjekte symbolisch auf einer Zeitachse angeordnet, die den chronologischen Verlauf der Anwendung festlegt. Die räumliche Anordnung der Objekte auf dem Bildschirm findet über Menüs statt [Bol95a]. Macromedia Flash [Fla04] ist ein gutes Beispiel für diese Klasse von Autorensystemen. Es ist ein Autorensystem, das die Erstellung von Animationen bis hin zu komplexen interaktiven Web-Anwendungen ermöglicht. Dabei unterstützt dieses Werkzeug die Autoren darin, eine einfache Anwendung mit Hilfe von Drag & Drop zu erstellen. Ferner ist durch die integrierte Sprache „ActionScript“ eine umfangreiche Programmierung möglich [Fla04].

In Abbildung 2.23 wird die graphische Benutzeroberfläche von Macromedia Flash vorgestellt. Es befindet sich links eine Werkzeugleiste, in der die wichtigsten Erstellungs-

und Bearbeitungswerkzeuge angebracht sind. Die Menüleiste befindet sich, ähnlich wie bei Windows-Standardsoftware, am oberen Rand des Fensters. In der Zeitleiste wird der Ablauf der Animation festgelegt. Die Informationsobjekte werden durch Symbole auf der Zeitleiste dargestellt. Hier werden die einzelnen Bildzustände ausgewählt und können dann in der Arbeitsfläche bearbeitet werden. Wie ein Film kann die Anwendung abgespielt werden und eventuell verändert oder angepasst werden.

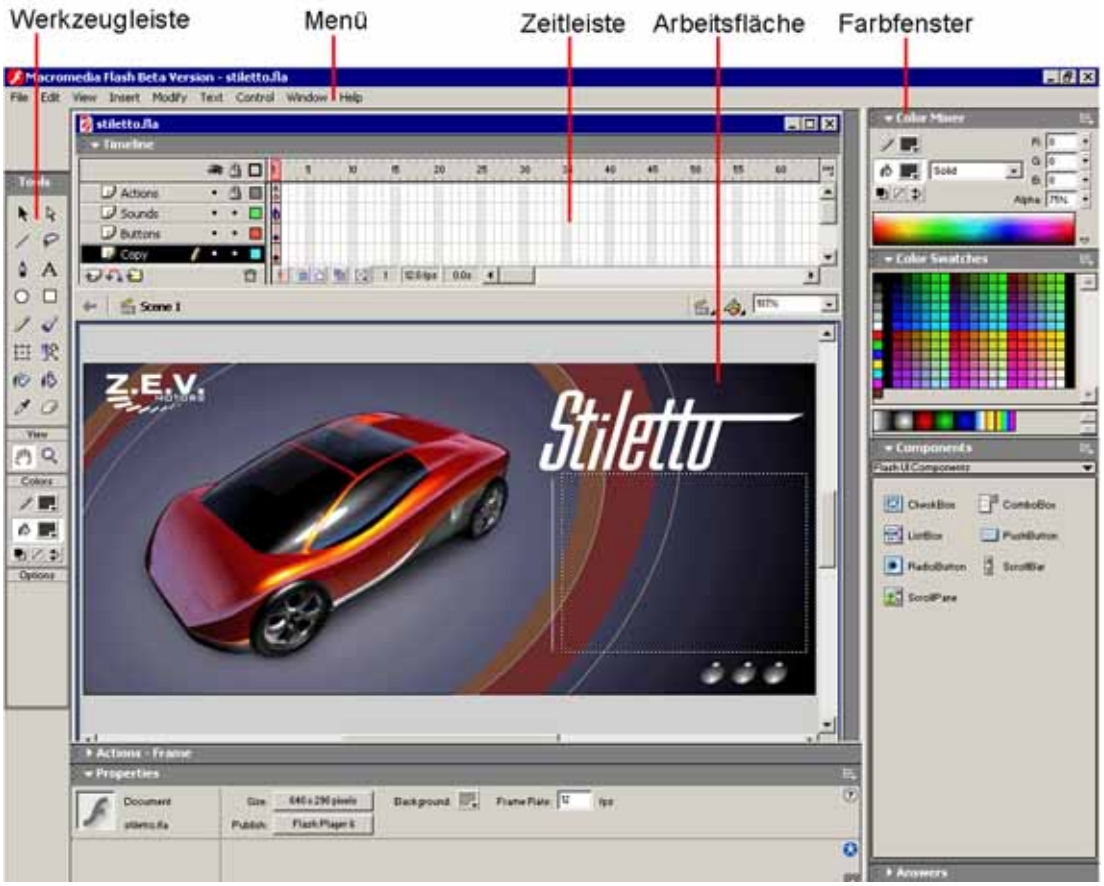


Abb. 2.23: Macromedia Flash [Fla04]

### Frame-basierte Autorensysteme

Bei frame-basierten Autorensystemen werden die Informationsobjekte auf Flächen angeordnet, die als Frames bezeichnet werden. Die Anwendung besteht aus einer Menge solcher Frames. Sie werden während der Vorführung für einen Zeitraum und in einer Reihenfolge angezeigt, die auch interaktiv bestimmt werden kann [Mül01]. Microsoft Power Point [Pow04] gehört zu dieser Klasse von Autorensystemen und

wird hier als Beispiel rudimentär vorgestellt. Es ist ein Hilfsmittel zum Erstellen von multimedialen Präsentationen. Dabei können verschiedene Informationsobjekte wie Texte, Graphiken, Audios und Videos in eine Präsentation integriert werden. Diese Informationsobjekte werden auf Frames angeordnet. Ferner ist es möglich, diesen Informationsobjekten eine Animation zuzuweisen. In Abbildung 2.24 wird die Oberfläche von Microsoft Power Point dargestellt. Sie besteht aus folgenden Hauptbereichen: Titelleiste, Menüleiste, Symbolleiste, Folienfenster, Folienübersicht und ein Bereich für Notizen zu jeder Folie, wie in Abbildung 2.24 zu sehen ist.

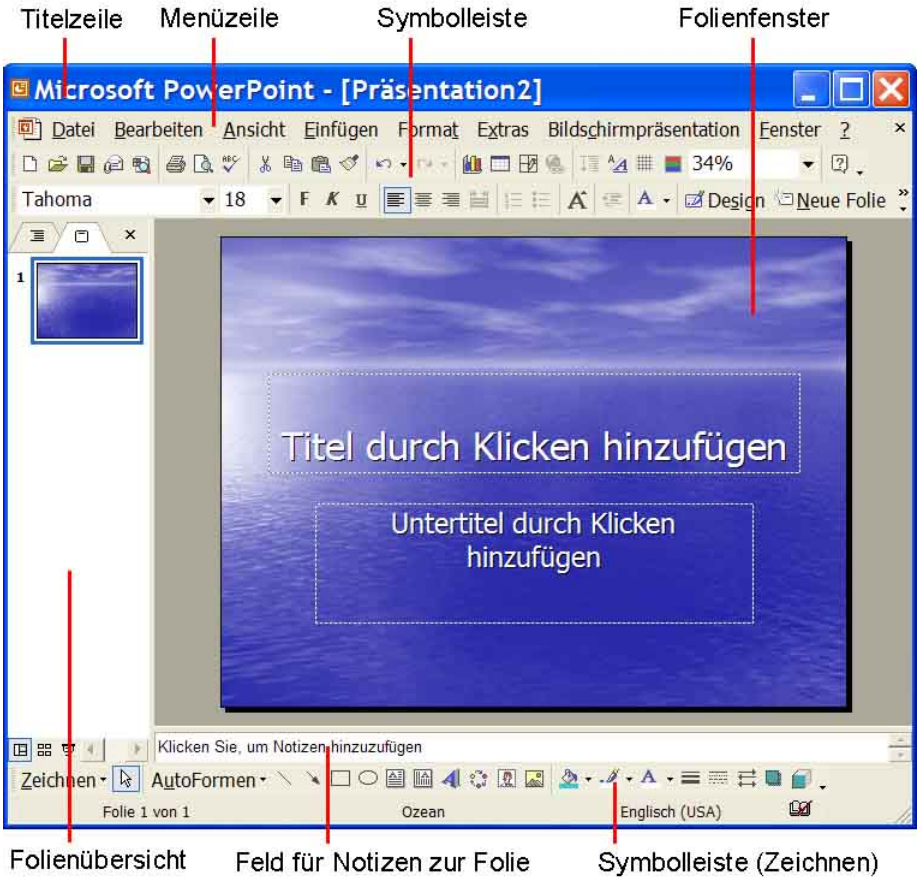


Abb. 2.24: Microsoft Power Point [Pow04]

### Flowchart-basierte Autorensysteme

Flowchart-basierte Autorensysteme sind dadurch gekennzeichnet, dass die Informationsobjekte als ikonisierte Symbole in Diagrammen dargestellt und durch Kanten miteinander verbunden werden. Durch diese Verbindungen wird der mögliche Ablauf der Präsentation festgelegt [Mül01]. In strukturierten flowchart-basierten Autorensystemen ist eine hierarchische Strukturierung der Diagramme möglich, das heißt, bestimmte logisch beziehungsweise funktional zusammenhängende Teile können zusammengefasst und in einem externen Diagramm ausgelagert werden. Sie bilden ein komplexes Medienobjekt. Die externen Diagramme werden durch eine spezielle Ikone in ihrem übergeordneten Diagramm referenziert [Bol95b]. Macromedia Authorware [Aut04] gehört zu dieser Klasse der Autorensysteme und wird hier kurz vorgestellt.

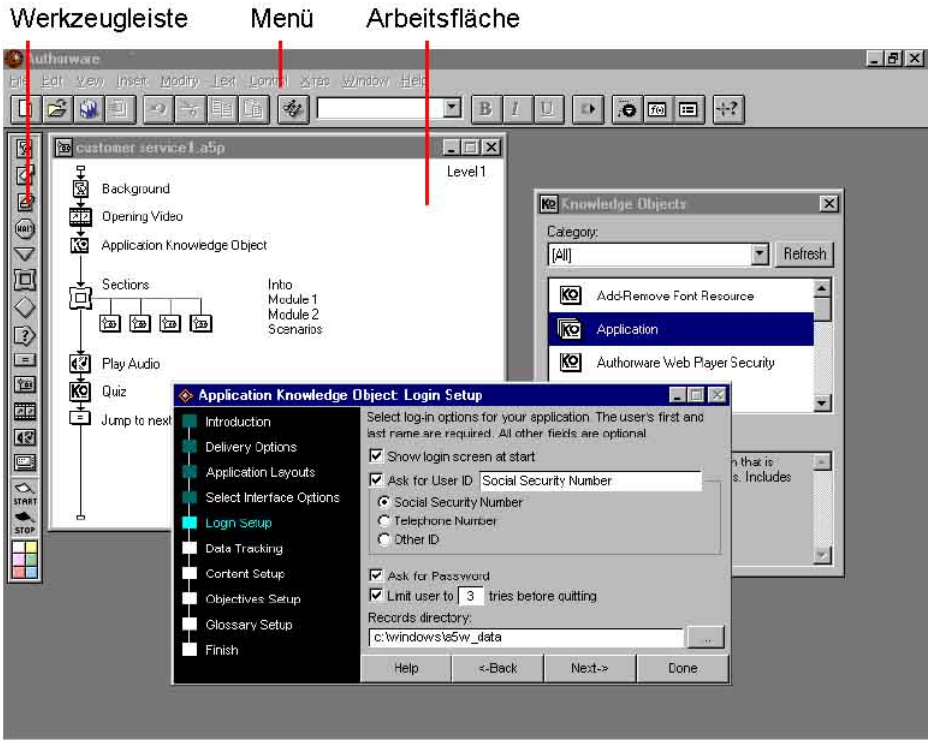


Abb. 2.25: Macromedia Authorware [Aut04]

Es bietet verschiedene Werkzeuge zur Erstellung interaktiver Lernanwendungen, bei denen Videos, Audios, Animationen, Texte und Graphiken zum Einsatz kommen. Die Anwendungen werden in Form von Flussdiagrammen erstellt. Mittels Drag & Drop werden vorgefertigte Icons, die verschiedene Funktionen repräsentieren, in die Anwendung eingefügt und vom Autor mit Inhalt gefüllt. Abbildung 2.25 zeigt die Oberfläche

von Macromedia Authorware. Links befindet sich eine Werkzeugleiste, die verschiedene Ikonen zur Erstellung der Anwendung enthält. Oben ist die Menüleiste angezeigt, die die allgemeinen Funktionen wie Speichern und Öffnen beinhaltet.

In diesem Kapitel wurden die Grundlagen der MR-Technologie und einige Grundbegriffe erläutert. Dazu sind die Anwendungsbereiche von MR rudimentär beschrieben worden. Anschließend wurden Autorensysteme mit Beispielen aus bekannter Standardsoftware vorgestellt. Mit Hilfe dieser Grundlagen können im nächsten Kapitel MR-Autorensysteme analysiert werden. Dabei werden verschiedene MR-Autorensysteme vorgestellt, aus denen ein System ausgewählt wird.

# Kapitel 3

# Analyse

In diesem Kapitel wird eine Anforderungsanalyse für MR-Autorensysteme durchgeführt. Danach werden verschiedene MR-Erstellungsmethoden untersucht. Die Anforderungsanalyse kann als erste Phase im Entstehungsprozess jeder Software gesehen werden. Sobald die Idee für ein Projekt entsteht, kann die Anforderungsanalyse beginnen und sie endet, wenn keine weiteren Anforderungen an das Projekt gestellt werden. Die Anforderungsanalyse ist eine besonders kritische Phase des Softwareerstellungsprozesses, da Fehler zu späteren Problemen beim Entwurf und der Implementierung des Systems führen. Diese Fehler würden unnötigerweise in den nachfolgenden Phasen des Projektes, wie zum Beispiel Spezifikation oder Realisierung, zu sehr hohen Kosten führen. Fehlende, unklare oder fehlerhafte Anforderungen führen oft zu fehlerhaften Entwicklungen in Projekten oder sogar zum Scheitern von Entwicklungsvorhaben.

Abbildung 3.1 stellt die Steigerung der Kosten abhängig von der Entwicklungsphase eines Systems graphisch dar. Wie aus der genannten Abbildung zu entnehmen ist, können Fehler bei der Spezifikation und beim Entwurf mit viel weniger Kosten korrigiert und aufgehoben werden als in den späteren Phasen der Entwicklung. Wenn man Anforderungsanalysen ausführlich durchführt, sind die Einsparungen bei den Fehlerkosten höher als die entstehenden Kosten für den Aufwand einer Anforderungsanalyse. Dies betont den Vorteil der Wirtschaftlichkeit bei dieser Analyse [Gli02].

Das Anforderungsdokument legt die Funktionalität des zu implementierenden Systems zwischen Auftraggeber und Auftragnehmer fest. Der Inhalt der Anforderungsdefinition dient als Grundlage einerseits für die Abnahme des Systems durch den Auftraggeber, andererseits für die Realisierung des Systems durch den Auftragnehmer. Es ist zu beachten, dass bei vielen internen Projekten Auftraggeber und Auftragnehmer als Rollen gespielt werden.

In der Anforderungsanalyse werden Anforderungen aus der Sicht des Anwenders erstellt. Dabei ist die technische Betrachtungsweise des Entwicklers unerheblich. Bei einer guten Anforderungsanalyse sollen Anforderungen nach der Definition von Balzert [Bal00] bestimmte Qualitätsziele verfolgen:

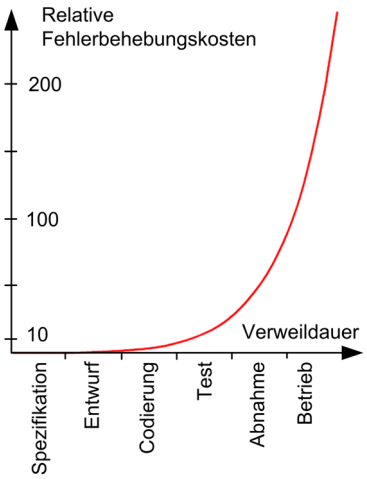


Abb. 3.1: Kosten für die Fehlerbehebung (vgl.[Boe81])

- Vollständigkeit, die die Beschreibung aller notwendigen Funktionen für den Einsatz des Produktes bedingt.
- Konsistenz, die die Widerspruchslosigkeit der Anforderungen untereinander erforderlich macht.
- Eindeutigkeit, die den Interpretationsspielraum auf eine einzige mögliche Interpretation der Anforderung beschränkt.
- Durchführbarkeit, die die Realisierbarkeit der Anforderung voraussetzt.

Des Weiteren ist es hilfreich, Anforderungen einfach zu formulieren. Dabei sollte in den Formulierungen auf den Gebrauch technischer Begriffe möglichst verzichtet werden. Sollten diese doch genannt werden, ist es empfehlenswert, sie in einem separaten Abschnitt im Anhang des Dokumentes (Glossar) aufzuführen. Wenn möglich, sollten die Anforderungen mit Hilfe eines Beispiels verdeutlicht werden.

Bevor in dieser Arbeit genauer auf eine Anforderungsanalyse und Spezifikation einer MR-Anwendung eingegangen wird, gibt dieses Kapitel einen groben Überblick über das gesamte Themengebiet wieder. Dazu werden im Unterkapitel 3.1 die allgemeinen Anforderungen an Softwaresysteme vorgestellt, welche dann weiter im Unterkapitel 3.2 in Bezug auf MR-Autorensysteme untersucht werden. Laut Drobnik [Dro03] und Sommerville [Som01] sollten Softwaresysteme gewisse Anforderungen erfüllen, um bestimmten Qualitätsansprüchen gerecht zu werden. Dafür existiert schon eine Anzahl an allgemeingültigen Anforderungen für Softwaresysteme, die zum Teil leicht modifiziert auf MR-Autorensysteme übertragen werden können.

Alle an MR-Autorensysteme gestellten Anforderungen werden in zwei Hauptkategorien aufgeteilt. Diese werden im Unterkapitel 3.2.1 als obligatorische und im Unterkapitel 3.2.2 als fakultative Anforderungen aufgeführt. Die obligatorischen Anforderungen bestehen hauptsächlich aus Anforderungen, die jedes Softwaresystem erfüllen muss, während die fakultativen Anforderungen das Ziel der Verbesserung des Systems verfolgen. Es ist allgemein üblich, obligatorische Anforderungen mit „sollen“ und wünschenswerte Anforderungen mit „sollten“ zu formulieren.

## 3.1 Allgemeine Anforderungen an Softwaresysteme

An jede Software unabhängig von ihrem Funktionsumfang werden Anforderungen gestellt. Wenn man die Ansammlung vieler Anforderungen an verschiedene Softwaresysteme untersucht, kann man bestimmte geforderte Eigenschaften finden, die diese Systeme alle gemeinsam haben. Nach Ansicht von Drobnik [Dro03] sollte ein Softwareprodukt unter anderem folgende Eigenschaften besitzen: Benutzbarkeit, Effizienz, Korrektheit, Wartbarkeit und Zuverlässigkeit. Diese Eigenschaften werden im Folgenden genauer erläutert. In jedem Softwareprodukt sollten die oben genannten Anforderungen möglichst genau und vollständig erfüllt werden. Dies kann jedoch nicht immer gewährleistet werden, da sich diese Merkmale manchmal widersprechen. Dieser Widerspruch wird weiter unten anhand eines Beispiels dargelegt.

#### Benutzbarkeit

Eine gute Software soll möglichst hohe Benutzerfreundlichkeit aufweisen. Die Benutzerfreundlichkeit enthält alle Aspekte der Interaktion zwischen Mensch und Maschine. Damit die Software die Anforderungen des Benutzers erfüllen und sich an die Bedürfnisse des Anwenders anpassen kann, muss für ihre Realisierung viel Aufwand investiert werden. Als ein einfaches Beispiel kann der Einsatz einer GUI bei einem Betriebssystem dienen. GUI ist eine Abkürzung für „Graphical User Interface“, wörtlich übersetzt „Graphische Benutzerschnittstelle“. Eine Benutzerschnittstelle ist derjenige Teil einer Software, der mit dem Benutzer kommuniziert, das heißt die Eingaben des Benutzers entgegen nimmt und die Ausgaben an den Benutzer weitergibt.

Beim Vergleich von zwei bekannten Betriebsystemen kann das Beispiel für den Einsatz der GUI verdeutlicht werden. Zunächst wird die GUI eines Microsoft Windows Betriebssystems der Version XP [Mic04] betrachtet. Windows XP stellt für den Einsatz vieler Funktionen so genannte Assistenten zur Verfügung, die den Benutzern helfen, die Funktionen des Betriebssystems kennen zu lernen und bei Bedarf zu nutzen. Der Benutzer kann mit Hilfe der Maus so gut wie alle Funktionen des Betriebssystems nutzen. Diese Vorgehensweise hilft Benutzern, die wenig Erfahrungen mit Computern haben, sich in der Microsoft Windows XP Umgebung rudimentär zurechtzufinden. Ganz an-

ders funktioniert Microsoft DOS [Mic04]. Microsoft DOS als Betriebssystem stellt für Benutzer wenig Hilfe zur Verfügung und meistens ist die Hilfe zu technisch oder schwer verständlich. Dadurch sind nur wenige Benutzer in der Lage, die Funktionen des Systems zu nutzen. An dieser Stelle ist es erkennbar, dass eine benutzerfreundliche Oberfläche viel zur Akzeptanz und Arbeitsfreude der Benutzer beitragen kann. Im Gegensatz zu einem Betriebsystem ohne GUI, wie Microsoft DOS, werden Betriebssysteme mit einer einfachen Benutzerführung und einer leicht bedienbaren Oberfläche schneller erlernt und häufiger von Benutzern eingesetzt.

#### Effizienz

Ein gut entworfenes Softwareprodukt soll eine hohe Effizienz nachweisen. Effizienz ist eine wichtige Eigenschaft, weil sie während der Softwareproduktion mit geringstem Aufwand die höchste Nutzbarkeit zu erreichen versucht. Dabei sollte der Aufwand an Arbeitszeit und Kosten für die Herstellung von Software sowohl während der Entwicklung als auch während dessen Einsatz möglichst klein sein. Laut Sommerville [Som01] gibt es einen weiteren Aspekt der Effizienz, der nicht unbeachtet bleiben sollte. Eine Software sollte nicht verschwenderisch mit Systemressourcen wie Speicher und Prozessorkapazität umgehen.

In der Frühzeit der Entwicklung von Computern waren ihre Benutzer hauptsächlich Programmierer und Systementwickler, die häufig mit folgenden Problemen konfrontiert worden sind:

- unverständliche Meldungen
- sehr schwer erkennbare Bedienungselemente
- unauffindbare Funktionen
- komplizierte Dialoge
- zeitaufwändige Funktionssuche in den Referenzhandbüchern

Als ein typisches Beispiel dienen die früheren Versionen des Betriebssystems Microsoft DOS, da bei diesem Betriebssystem viele dieser Probleme auftreten. Die rasante Entwicklung im Bereich der elektronischen Datenverarbeitung hat vielen Menschen mit unterschiedlichen Kenntnissen ermöglicht, Computeranwender zu werden. Sie können in zwei Hauptgruppen aufgeteilt werden:

- Programmierer, die die Anwendungsprogramme entwickeln
- Benutzer, die diese Programme anwenden

Die Programmierer erstellen Anwendungen für verschiedene Gebiete der Wirtschaft, Verwaltung und Forschung, die von ihnen hauptsächlich aus technischer Sicht betrachtet werden. Dabei kann es vorkommen, dass die Anforderungen der Benutzer außer Acht gelassen werden, da sie sich auf der Entwicklungsebene befinden. Um diesem Problem entgegen zu wirken, sollten die Benutzeranforderungen möglichst genau in Form einer Spezifikation definiert werden. Zum Beispiel sollte eine Software von deren Benutzern verstanden und leicht eingesetzt werden können. Dazu kann eine klar formulierte Hilfe beziehungsweise Dokumentation hilfreich sein.

Um bei der Spezifikation die Benutzeranforderungen genauestens zu erfassen, wurde von Karat [Kar98] vom Thomas J. Watson Research Center der IBM ein Leitfaden, genannt „User's Bill of Rights", erstellt. Sie arbeitet als Sozialpsychologin und Benutzerschnittstellendesignerin im genannten Forschungszentrum. Sie hat aus den Erfahrungen der Benutzer heraus die so genannten „Grundrechte" der Anwender der Datenverarbeitung wie folgt formuliert:

- „Der Anwender hat immer Recht. Existiert ein Bedienungsproblem, dann ist das System das Problem und nicht der Anwender.
- Der Anwender hat das Recht auf einfach zu installierende Software und Hardware.
- Der Anwender hat das Recht auf ein System, das genau die Leistung besitzt, die vom Hersteller versprochen wird.
- Der Anwender hat das Recht auf einfache Bedienungsanleitungen, die helfen, das System zu verstehen und anzuwenden.
- Der Anwender hat das Recht auf eine Kontrolle des IT-Systems und muss entsprechende Informationen erhalten.
- Der Anwender hat das Recht, präzise über die gerade laufenden Aufgaben und Prozesse bis zur Beendigung informiert zu sein.
- Der Anwender hat das Recht, Klarheit über die Systemanforderungen für einen erfolgreichen Software- oder Hardwareeinsatz informiert zu werden.
- Der Anwender hat das Recht, die Grenzen der Systemleistungen zu erfahren.
- Der Anwender hat das Recht, mit dem Technikanbieter zu kommunizieren und eine hilfreiche Antwort zu erhalten.
- Der Anwender sollte der Herr über Hardware und Software sein, nicht umgekehrt. Produkte haben natürlich und intuitiv zu sein" [Hel00].

### Korrektheit

Eine immer wichtigere Funktion in der Softwareentwicklung nimmt die Korrektheit ein. Korrektheit ist sehr entscheidend für den erfolgreichen Einsatz der Software. Eine Software ist dann nützlich, wenn sie die spezifizierte Funktionalität erfüllt und während des Einsatzes die vorhandenen Ressourcen effizient einsetzt. Die Software muss die gestellte Aufgabe korrekt lösen. Ein Programm, das diesen Anforderungen nicht genügt, ist im Allgemeinen wertlos oder kann sogar große wirtschaftliche Schäden anrichten. Als Beispiel für ein fehlerhaftes Softwaresystem kann die Abrechnungssoftware für die Berechnung der Sozialhilfe in amerikanischen Behörden genannt werden [Dro03]. Ein Fehler in der Abrechnungssoftware der Behörden war über 16 Jahre unentdeckt geblieben. Die Konsequenz war, dass 478 Millionen amerikanische Dollar an die rechtmäßigen Sozialhilfeempfänger nicht ausgezahlt wurden. Ein anderes Beispiel ist ein Fehler im Steuerprogramm einer indischen Rakete, der zu ihrem Absturz in den Ozean führte. Dabei entstand ein Schaden in Höhe von 144 Millionen amerikanische Dollar.

Damit überhaupt geklärt werden kann, ob ein Produkt korrekt ist und die gegebenen Anforderungen korrekt lösen kann, müssen diese Anforderungen in einer Spezifikation und nicht etwa in Form eines flüchtigen Gesprächs schriftlich beschrieben werden. Außerdem muss ein Verfahren entwickelt werden, das diese Spezifikation mit dem fertigen Produkt vergleichen kann.

### Wartbarkeit

Der Begriff Wartbarkeit hat seine Wurzeln im Warten im Sinne von Pflegen und Betreuen und wird im softwarespezifischen Sinne für die Weiterentwicklung der Softwaresysteme benutzt. Es gibt kaum Softwaresysteme, die niemals aktualisiert oder konvertiert, mit anderen Worten gewartet werden müssen. Manchmal kann dies mit Hilfe von Programmen und Skripten automatisch geschehen, aber meistens wird ein Entwickler diese Änderungen durchführen müssen. In diesen Fällen kann ein überschaubares und klar strukturiertes System vorteilhaft sein.

Die Wartbarkeit von Software ist eine äußerst wichtige Eigenschaft in der Softwareentwicklung und gibt an, mit welchem Aufwand Softwaresysteme geändert werden können. Der Grund dafür, warum wartbare Systeme eine hohe Qualität haben, liegt darin, dass sich diese Systeme an die Kundenbedürfnisse leichter anpassen lassen. Die Wartbarkeit kann sich zu einem entscheidenden Merkmal entwickeln, weil eine Änderung von Kundenbedürfnissen unvermeidlich ist. Systeme müssen über ihren gesamten Verwendungszeitraum hinweg gepflegt werden. So sollten zum Beispiel Fehler leicht und schnell beseitigt und Änderungen einfach und rückwirkungsfrei durchgeführt werden können. Um dies einfach gewährleisten zu können, sollten diese Systeme wart-

bar sein. Die Wichtigkeit der Wartbarkeit für ein Softwaresystem steigt proportional zur Verwendungsdauer der Software und zur Anzahl der Personen, die diese Software benutzen.

#### Zuverlässigkeit

Auch die Zuverlässigkeit spielt in einem Softwareprodukt eine wichtige Rolle. Zuverlässige und sichere Systeme zu realisieren ist eine große Herausforderung für Softwareentwickler. Zuverlässigkeit für ein Softwaresystem bedeutet dabei grob betrachtet eine genaue Ausführung jeder angeforderten Funktion. Bei einer Fehlfunktion darf ein Softwaresystem keinen physikalischen oder wirtschaftlichen Schaden anrichten.

Als ein Beispiel für einen wirtschaftlichen Schaden kann die Explosion der Rakete Ariane5 am 4. Juni 1996 genannt werden [Dro03]. Ihr Absturz wurde durch eine fehlerhafte automatische Messung der Horizontalgeschwindigkeit verursacht. Die Rakete explodierte 40 Sekunden nach ihrem Start und verursachte einen Schaden von circa einer halben Milliarde DM. Die Software, die den Fehler verursachte, wurde für die Ariane4 entwickelt und ohne Änderungen auf die Ariane5 portiert, obwohl Ariane5 andere Flugeigenschaften aufwies.

Ein weiteres Beispiel für einen wirtschaftlichen Schaden passierte in Denver (Colorado), International Airport [Dro03]. Das System sollte Gepäckstücke automatisch vom Flugzeug zum Terminal transportieren. Es sollte mit Hilfe von Induktionsmotoren und Strichcodeaufklebern, die von Scannern gelesen wurden, funktionieren. Aus verschiedenen Gründen wie zum Beispiel Software- und Hardwareproblemen konnte das System jedoch nicht rechtzeitig fertig gestellt werden. Dies führte zu einer verzögerten Eröffnung des Flughafens und zu einem finanziellen Schaden in Millionenhöhe.

Die Verbesserung der Zuverlässigkeit eines Systems kann mit sehr hohen Entwicklungskosten verbunden sein, wie das in Abbildung 3.2 graphisch dargestellt wird.

Von den oben genannten Eigenschaften widersprechen sich typischerweise Effizienz und Benutzbarkeit. Bei der Gegenüberstellung der oben genannten Beispiele benötigt die Entwicklung von Microsoft Windows XP mehr Aufwand, um den Anforderungen der Benutzerfreundlichkeit gerecht zu werden, als Microsoft DOS. Und während des Einsatzes benötigt Microsoft Windows XP mehr Ressourcen als Microsoft DOS.

Es gibt eine Reihe von weiteren Eigenschaften, die abhängig vom Einsatzgebiet des Softwaresystems von diesem angefordert werden können. Einige dieser Anforderungen sind zum Beispiel: Geschwindigkeit, Skalierbarkeit, Wiederverwendbarkeit und Portierbarkeit.

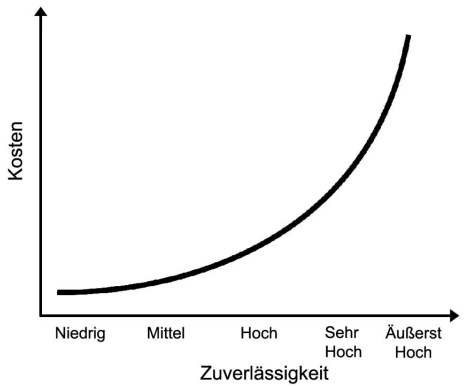


Abb. 3.2: Verhältnis zwischen Kosten und Zuverlässigkeit (vgl. [Som01])

Geschwindigkeit ist wichtig für den Einsatz eines Softwaresystems, da eine unzureichende Geschwindigkeit die Betriebskosten der Software erhöhen kann.

Auch Skalierbarkeit ist entscheidend für ein Softwareprodukt. Skalierbarkeit ist die Fähigkeit eines Systems, mit Datenmengen von unterschiedlicher Größe umzugehen. Das heißt, für ein System dürfte keine Grenze existieren, bei deren Überschreitung es nur noch sehr langsam oder mit einem unakzeptablen hohen Aufwand betrieben wird. Dieses Problem taucht beispielsweise bei Datenmengen auf, die größer sind als der Hauptspeicher. Ein System ist gut skaliert, wenn es für die Verarbeitung der zehnfachen Datenmenge circa das Zehnfache an Ressourcen beansprucht. Und das System ist schlecht skaliert, wenn dieses schon bei der Verarbeitung der doppelten Datenmenge circa das Zehnfache an Ressourcen belegt. Der Einsatz einer Software kann nur dann über längerer Zeit gewährleistet werden, wenn sie die ständig wachsenden Datenmengen verarbeiten kann.

Durch die Wiederverwendbarkeit reduzieren sich die Entwicklungs- und die Betriebskosten einer Software mittels einer sinnvollen Wiederverwendung von vorhandenen Softwareprodukten. Beim Softwareentwurf werden keine grundlegend neuen Ideen mehr erzeugt, sondern man erinnert sich an alte, gute Lösungen und verwendet diese. In der Zukunft bedeutet dies weniger Arbeitsaufwand und somit eine Kostenreduzierung auf längere Sicht. Jedoch ist zu beachten, dass die Kostenreduzierung nur dann erheblich sein wird, wenn für die vorhandene Softwareprodukte keine Lizenzgebühren anfallen. Hohe Lizenzgebühren können Kostensteigerungen verursachen.

Portierbarkeit erweitert das Einsatzgebiet der Software und macht diese dadurch für ein breiteres Kundenspektrum verfügbar. Es erhöht aber gleichzeitig den Entwicklungsaufwand.

## 3.2 Anforderungen an Mixed Reality-Autorensysteme

Die bisher aufgeführten Anforderungen an Softwaresysteme sind zum größten Teil allgemein gültig und somit auf MR-Autorensysteme übertragbar. Bei einer Übertragung müssen jedoch nicht nur die Anforderungen bekannt sein, sondern auch die Details für das System, an das die Anforderungen gestellt werden. Bei den Details handelt es sich um die Funktionen, Umgebungen und Einschränkungen des zu untersuchenden Systems. Im Unterkapitel 3.2.1 wird diese Übertragung durchgeführt, worin unter dem Aspekt der allgemeingültigen Anforderungen die Funktionen des Systems betrachtet und daraus obligatorische Anforderungen an das System abgeleitet werden. Dieses Unterkapitel ist ein Teil der vorliegenden Anforderungsanalyse. Ein anderer Teil besteht aus dem Unterkapitel 3.2.2, der zum größten Teil spezielle und nicht übertragene Anforderungen an MR-Autorensysteme beinhaltet. In ihren Formulierungen werden Anforderungen beschrieben, die unter anderem das Ziel der Optimierung des Systems verfolgen, also die programmierungslose Softwareerstellung von MR-Anwendungen.

### 3.2.1 Obligatorische Anforderungen

Die hier aufgeführten obligatorischen Anforderungen werden wie schon erläutert von allgemeinen Anforderungen an Softwaresysteme auf MR-Autorensysteme abgeleitet.

Aufgrund des geringen Bekanntheitsgrades der MR-Technologie spielt die Benutzerfreundlichkeit für ein MR-Autorensystem eine entscheidende Rolle. Je benutzerfreundlicher ein solches System ist, umso leichter wird dieses zu verstehen, zu bedienen und zu erlernen sein. Dabei ist beispielsweise eine umfassend indizierte Onlinehilfe mit einer Schnelleinführung und einer Suchfunktion sehr hilfreich. Dadurch wird es Autoren mit keiner oder wenig Erfahrung im Bereich Mixed Reality vereinfacht, einen Einstieg beziehungsweise Umstieg auf die MR-Technologie zu erreichen. Daher ist es von Vorteil, wenn das MR-Autorensystem über eine angemessene Bedieneroberfläche verfügt.

Die Oberfläche des MR-Autorensystems sollte Namen und Symbole verwenden, die für die Benutzer aufgrund ihrer Erfahrungen mit ähnlichen Werkzeugen bekannt sind, damit sie diese Erfahrungen weiterhin anwenden können. Zusätzlich sollten dem Benutzer folgende Funktionen zur Verfügung stellen:

- Drag & Drop, womit das Ziehen und Ablegen von Objekten in die Oberfläche ermöglicht wird.
- Kopieren, Ausschneiden und Einfügen, wodurch die Arbeit mit Hilfe der Zwischenablage des Betriebssystems vereinfacht wird.
- Rückgängig und Wiederherstellen, die jede Aktion des Benutzers rückgängig machen können.

- Vorschau, um die erstellte Arbeit vor dem endgültigen Abschluss darstellen zu können.
- Autokorrektur, um eventuelle Fehleingaben automatisch zu erkennen und beseitigen zu können.

Des Weiteren sollen Fehlermeldungen aussagekräftig und für Benutzer verständlich sein. Sie sollten Vorschläge zur Korrektur des Fehlers oder zumindest einen Verweis auf die Onlinehilfe des Systems anbieten.

Das MR-Autorensystem soll genügend Hilfsmittel, Werkzeuge und Dokumentationen besitzen, um Autoren bei der Erstellung ihrer Anwendungen zu unterstützen. Eine Benutzerdokumentation sollte hilfreiche Informationen über alle Systemdienste beinhalten. Die Benutzerdokumentation kann Autoren bei Kaufentscheidungen helfen.

Weiterhin ist es wichtig, eine detaillierte Beschreibung der Installation und konfigurationsabhängiger Dateien in der Benutzerdokumentation aufzuführen. Die Installationsanweisung sollte den Einsatz auf unterschiedlichen Betriebsystemen berücksichtigen und den Autoren ausreichende Informationen zur Systempflege zur Verfügung stellen. Nach einer Installationsanweisung sollte die Benutzerdokumentation eine schnelle Einführung in das System ermöglichen. Dadurch fällt es Autoren leichter, sich einen Einstieg in das MR-Autorensystem zu verschaffen. Da es während des Einsatzes des MR-Autorensystems zu Fehlfunktionen kommen kann, sollten mögliche Fehler in der Benutzerdokumentation aufgelistet werden. Auch ein Vorschlag zur Fehlerbehebung und eine Beschreibung der vermutlichen Ursachen dieser Fehler zählen zu den Merkmalen einer guten Benutzerdokumentation.

Eine Anforderung, die in mancher Hinsicht wie im Unterkapitel 3.1 in Konflikt zur Benutzerfreundlichkeit steht, ist die Effizienz des MR-Autorensystems, welche den Hauptfokus auf Reduzierung der Betriebskosten des Systems richtet. Die Effizienz eines MR-Autorensystems wird dadurch erreicht, dass das System die ihm zur Verfügung stehenden Ressourcen, wie Arbeitsspeicher, Festplattenspeicher oder Prozessorzeit, nicht unnötig beansprucht. Der optimale Einsatz dieser Ressourcen wird durch Verbesserung der Reaktions-, Verarbeitungszeit und der Speichernutzung verdeutlicht.

Korrektheit ist eine weitere wichtige Eigenschaft, die ein gutes MR-Autorensystem besitzen sollte. Ein MR-Autorensystem ist funktional korrekt, wenn es sich genauso verhält, wie es in der Spezifikation definiert ist. Genauso kann dieses System als unbrauchbar bezeichnet werden, wenn es im Sinne der Spezifikation nicht korrekt ist.

Die letzte obligatorische Anforderung an ein MR-Autorensystem bezieht sich auf die Zuverlässigkeit des Systems. Eine Fehlfunktion des MR-Autorensystems darf beispielsweise keine Fehlfunktion des Betriebssystems zur Folge haben. Weiterhin dürfte ein

Bedienungsfehler nicht eine Neuinstallation der Arbeitsstation verursachen. Damit ist also gemeint, dass ein MR-Autorensystem keine physikalischen oder wirtschaftlichen Schäden verursachen darf. Des Weiteren sollte das MR-Autorensystem ein ausgeprägtes Toleranzverhalten gegenüber Fehleingaben oder Bedienungsfehlern aufweisen. Mit anderen Worten sollte das System nicht bei jedem kleinen Bedienungsfehler abstürzen. Darüber hinaus wird ein MR-Autorensystem erst dann zu einem zuverlässigen System, wenn es sich in einem bestimmten Zeitraum so verhält, wie es von ihm erwartet wird.

### 3.2.2 Fakultative Anforderungen

Die nun folgenden Anforderungen verfolgen wie bereits erwähnt das Ziel der Verbesserung und Optimierung von MR-Autorensystemen und tragen zur Wirtschaftlichkeit dieser Systeme erheblich bei. Ferner ist noch zu erwähnen, dass die ersten vier der folgenden Anforderungen für MR-Autorensysteme sehr wichtig sind, da ihre Nichterfüllung die Qualität der Software gravierend verschlechtern würde. Die anderen Anforderungen besitzen niedrigere Priorität und müssen daher nicht zwingend erfüllt werden. Ein Grund dafür könnten zu hohe Kosten sein, die durch ihre Erfüllung verursacht werden könnten.

#### Vorschau

Die MR-Technologie bietet eine Kombination von realen und virtuellen Umgebungen. MR-Autorensysteme müssen während der Entwicklung von Anwendungen mit dem realen Einsatzgebiet der MR-Anwendung agieren. Dafür existieren Werkzeuge, wie Kameras, die idealerweise vom MR-Autorensystem unterstützt werden sollten. Das System sollte eine Möglichkeit zur Vorschau anbieten, damit der Autor an seinem Arbeitsplatz die MR-Anwendung entwickeln und testen kann, ohne sich an dem jeweiligen Ort des Endnutzers befinden zu müssen.

Ferner sollte jede MR-Anwendung auch als so genannte „Stand-Alone-Anwendung“ erstellt werden können. Damit ist gemeint, dass die MR-Anwendung auf jedem beliebigen Computer ausgeführt werden kann, ohne spezielle oder zusätzliche Software installieren zu müssen.

#### Klar definierte Autorenrollen

Wie Abawi [ADG04] erläutert, gibt es bei der Erstellung einer MR-Anwendung viele Aufgaben, die erledigt werden sollten. Einige dieser Aufgaben sind zum Beispiel die Erstellung von 2D-Objekten wie Texte, Graphiken oder die Erstellung und Animation von 3D-Modellen. Diese Objekte müssen dann in das System integriert werden.

Normalerweise ist kaum ein Autor in der Lage, alle dieser Aufgaben zu erledigen. Das heißt, dass unterschiedliche Autoren mit verschiedenen Kenntnissen an der Erstellung der MR-Anwendung gemeinsam arbeiten sollten.

Ferner ist noch zu erwähnen, dass man für die Erstellung einer MR-Anwendung nicht immer von Autoren auf Expertenniveau ausgehen kann. Daher sollte zum Beispiel unterschieden werden zwischen Anwendungsautoren und anderen Autoren, die für die Programmierung und die technische Realisierung verantwortlich sind. Das MR-Autorensystem sollte die Rolle der Autoren und Entwickler klar definieren, so dass die Anwendungsautoren ohne Programmieren und die Entwickler durch Programmieren an der Gesamtentwicklung arbeiten können. Wenn mehrere Autoren an der Erstellung einer MR-Anwendung arbeiten, sollte das System genügend Hilfsmittel für eine reibungslose Zusammenarbeit anbieten, weil es bei fehlender Kommunikation zwischen den Autoren zur Parallelentwicklungen mit eventuell groben Fehlern kommen kann. Daher sollten diese Hilfsmittel die Autoren unterstützen, um beispielsweise eine problemlose Anpassung ihrer Arbeitsergebnisse oder Versionen durchzuführen.

#### Dedizierte Werkzeuge

Das MR-Autorensystem sollte Schnittstellen zur Kommunikation mit Standardsoftware anbieten. Dadurch ist es möglich, benötigte Objekte für die Erstellung von MR-Anwendungen einerseits innerhalb des MR-Autorensystems und andererseits trotzdem mit Hilfe von professionellen Softwaresystemen zu erstellen. Dabei sollten diese Softwaresysteme als Werkzeuge in das MR-Autorensystem eingebunden werden. Das heißt, der Autor sollte in der Lage sein, während der Erstellung der MR-Anwendung mit dem Autorensystem die beschriebene Software aufzurufen und seine Arbeit nahtlos fortsetzen zu können. Es ist ineffizient, wenn existierende Schnittstellen zur Standardsoftware nicht genutzt werden, sondern stattdessen neue Technologien entwickelt werden, um dasselbe Ergebnis zu erzielen. Viele der einsetzbaren Standardsoftware sind den meisten Benutzern vertraut. Für das Erlernen dieser Systeme haben diese Benutzer zum Teil sehr viel Zeit investieren müssen. Es ist nicht effektiv, wenn diese Systeme neu gebaut werden müssen, weil dann nicht nur die Entwicklung, sondern auch die Schulung der Autoren in diesen neuen Systemen mit großem Aufwand verbunden sein kann. Daher sollte das MR-Autorensystem solche Standardsoftware unterstützen und Schnittstellen nutzen, die die Kommunikation zwischen ihnen sicherstellen.

Da für MR-Anwendungen 3D-Modelle zum häufigen Einsatz kommen, sollte eine Schnittstelle zu einer 3D-Modellierungssoftware wie Alias Maya [May04] oder 3D Studio max [3dS04] erstellt werden. Ferner ist es hilfreich, wenn zusätzlich noch eine Prozess-Modellierungssoftware, wie zum Beispiel Microsoft Visio [Vis04] oder ARIS Toolset [ARI04], in das MR-Autorensystem eingebunden wird. Dabei sollen nicht nur sche-

matische Zeichnungen und Diagramme erstellt, sondern gleich in die MR-Anwendung umgesetzt werden. Normalerweise erstellt ein Autor ein Diagramm für seine Anwendung und setzt dieses Diagramm mit Hilfe des Autorensystems um. Bei der Umsetzung können dem Autor Fehler passieren. Ein Automatismus würde hierbei dem Autor helfen, Zeit zu sparen und die Anzahl der Fehlerquellen zu reduzieren. Hinzukommend kann ein Autor mit wenigen Erfahrungen in Bezug auf MR-Autorensysteme aber mit fundierten Kenntnissen im Bereich der Prozessmodellierungssoftware seine Stärken in der Erstellung der MR-Anwendung besser einbringen.

#### Unterstützung externer Dateien

Die MR-Autorensysteme sollten die Verwendung von externen Dateien unterstützen, die mit Hilfe von Standardsoftware erstellt werden und in den meisten Softwaresystemen zum Einsatz kommen. Als Hilfsfunktionen kommen beispielsweise für die meisten auf Windows basierenden Systemen so genannte kompilierte Hyper Text Markup Language (HTML)-Hilfedateien (*.chm) zum Einsatz. Bei den kompilierten HTML-Hilfsdateien handelt es sich um indizierte Hilfsdateien, die auch als Referenzhandbücher eingesetzt werden können. Die Implementierung einer solchen Hilfsfunktion kann sehr aufwändig werden, wenn sie durch eigene Entwicklungen realisiert werden soll.

Weitere externe Dateien, die unterstützt werden sollten, sind Audio-Dateien. Dabei sollten mindestens die zwei gängigsten Formate WAV und MPEG Audio Layer 3 (MP3) unterstützt werden. WAV-Format ist ein Dateiformat für digitale Audio-Dateien, welches unkomprimiert gespeichert wird, während MP3 ein komprimiertes Format von Audio-Dateien ist, womit eine zehnfache Komprimierung erreicht werden kann. Dabei sollte ein Werkzeug zum Abspielen dieser Dateien in das MR-Autorensystem implementiert werden. Es ist für die Aufnahme von Audio-Dateien nicht notwendig, dass ein entsprechendes Werkzeug im MR-Autorensystem existiert.

Ebenfalls wichtig ist die Unterstützung verschiedener Graphikformate. Es sollten gängige Formate wie BMP (Bitmap), JPEG (Joint Photographic Experts Group) und GIF (Graphics Interchange Format) unterstützt werden. Ein Werkzeug zur Bearbeitung dieser Graphikformate sollte nicht zwingend in das MR-Autorensystem integriert werden. Jedoch ist ein Werkzeug zur Ansicht der Formate für die Autoren sehr hilfreich.

Video-Formate sind wichtige Formate, die von einem MR-Autorensystem unterstützt werden sollten. Die gängigsten Formate sind Audio Video Interleaved (AVI) und Moving Pictures Experts Group (MPEG). Sie sind Multimedia-Dateiformate, mit denen Videos gespeichert werden können. Auch für Video-Formate sollten Werkzeuge nur zur Wiedergabe implementiert werden.

### Berücksichtigung aller Szenenänderungen

Bei der Entwicklungen der MR-Anwendung sollte das MR-Autorensystem alle Szenenänderungen berücksichtigen, speziell Szenen, in denen virtuelle Objekte von realen überdeckt werden. Dadurch kann die Szene realistisch aussehen. Wie in Abbildung 3.3 dargestellt, kann man sich die Situation so vorstellen, dass sich ein virtuell erstellter Ball gleichmäßig von Punkt A nach D entlang des rot markierten Pfades bewegt.

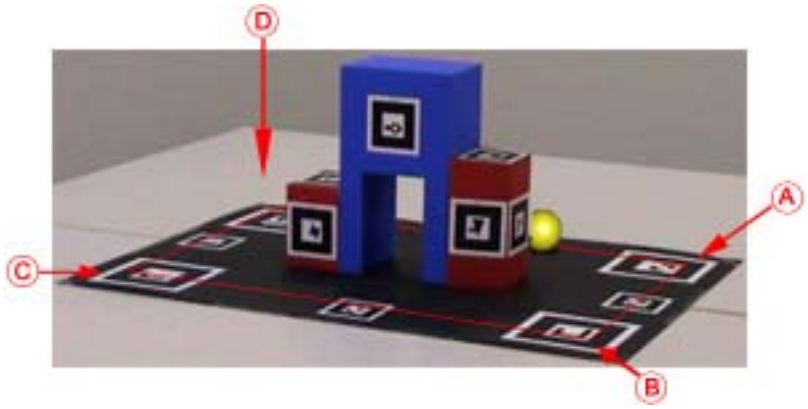


Abb. 3.3: Beispiel für das Verdeckungsproblem (vgl. [ARD04])

Des Weiteren sind in Abbildung 3.3 alle sichtbaren Objekte real und einzig der Ball virtuell. Die rotfarbige Markierungen A bis D und die dazugehörigen Pfeile dienen allein zur Verdeutlichung der Situation und gehören nicht zur Szene. Sobald der Ball den Punkt D passiert, bewegt er sich zum Punkt A weiter und genau hier wird er von den würfelähnlichen Objekten in der Mitte des Bildes verdeckt. Zumindest sollte in der MR-Anwendung die Illusion hervorgerufen werden, dass der Ball verdeckt wäre, um die Szene so realistisch wie möglich zu gestalten. Eine solche Eigenschaft sollte vom MR-Autorensystem zur Verfügung gestellt werden.

### Schattierung virtueller Objekte

Eine weitere Eigenschaft, die ein MR-Autorensystem bereitstellen sollte, ist die Schattierung virtueller Objekte in der Szene, um eine Erhöhung der realistischen Optik zu erreichen. Das heißt, ein Autor sollte mit Hilfe des MR-Autorensystems in der Lage sein, mit wenig Aufwand authentische Schatten für virtuelle Objekte in seiner Anwendungen zu erstellen. Virtuelle nicht schattierte Objekte sehen in der Umgebung von schattierten realen Objekten künstlich und unrealistisch aus. Deswegen ist die Schattierung eine essentielle Funktion, die bei der Realisierung einer authentischen Optik hilfreich sein kann. Zur Veranschaulichung dieser Gedanken sind folgende Abbildungen gut geeignet.

Abbildung 3.4 zeigt auf der linken Seite einen virtuellen Becher ohne Schatten, der künstlich wirkt. Auf der rechten Seite wird der gleiche Becher mit Schatten dargestellt, der somit eine Illusion der Zugehörigkeit zu seiner realen Umgebung beim Betrachter erzeugt.

Abb. 3.4: Darstellung von Schatten in einer MR-Anwendung [SKT03]

Ein weiterer Vorteil bei der Verwendung von Schatten ist die Verdeutlichung der dreidimensionalen Darstellung von virtuellen Objekten. In Abbildung 3.5 werden verschiedene dreidimensionale Figuren als bunte Bälle in einer Szene hinzugefügt. Auf der linken Seite ist es für den Betrachter unmöglich zu erkennen, welcher Ball am nächsten zur Kamera steht, weil die Objekte keinen Schatten im Bild werfen. Dieses Problem hat der Betrachter auf der rechten Seite nicht.

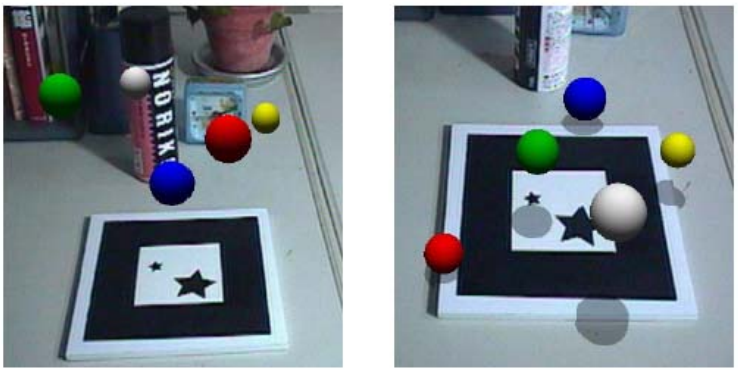

Abb. 3.5: 3D-Darstellung von virtuellen Objekten [SKT03]

### Abwärtskompatibilität

Ein MR-Autorensystem wird sicherlich im Laufe seines Einsatzzeitraumes Verbesserungen durchleben, die eine gravierende Veränderung der Erstellungsprozesse von MR-Anwendungen bewirken kann. Diese Veränderung kann die Datenstruktur in erstellter MR-Anwendung beeinflussen, in manchen Fällen sogar erneuern. Trotz dieser Veränderungen sollte das MR-Autorensystem die Möglichkeit anbieten, Anwendungen, die mit älteren Versionen erstellt wurden, stets zu importieren und weiter zu entwickeln, um abwärtskompatibel zu sein. Ein Computerprogramm wird als abwärtskompatibel bezeichnet, wenn es die mit einer älteren Version des Programms erzeugten Daten bearbeiten kann.

### Skriptunterstützung

Ein MR-Autorensystem sollte Autoren die Möglichkeit bieten, die Erstellung von Anwendungen mit Hilfe von Skriptsprachen zu beschleunigen. Dabei liegt der Vorteil in einer wesentlich größeren Flexibilität und Leistungsfähigkeit. Da die Skriptsprachen einfacher strukturiert sind, werden sie häufiger als herkömmliche Programmiersprachen von Autoren eingesetzt. Des Weiteren werden die kritischen Programmierfehler von Autoren reduziert. Der Nachteil besteht jedoch darin, dass Programmierkenntnisse vom Autor vorausgesetzt werden.

### Komponentenbasiertes Design

Das MR-Autorensystem sollte möglichst auf Komponentenbasis funktionieren, um die Effizienz und die Zuverlässigkeit des Systems zu steigern. Nach Balzert [Bal00] ist eine Komponente ein abgeschlossener binärer Software-Baustein, der eine anwendungsorientierte, semantisch zusammengehörende Funktionalität besitzt, die nach außen über Schnittstellen zur Verfügung gestellt wird. Komponenten sind beispielsweise Oberflächenelemente wie Druckknöpfe, Textfelder und Tabellen, die in gängige Standardsoftware zum Einsatz kommen.

Das Beachtenswerte an einer Komponente ist die Eigenschaft der Wiederverwendbarkeit, die jede Komponente besitzen sollte. Die Wiederverwendbarkeit der Komponenten ist dahingehend hilfreich, dass beim Erstellen von Anwendungen die einzelnen Komponenten seltener neu erstellt werden müssen. Dadurch gewinnt man wertvolle Zeit in der Entwicklungsphase jeder Anwendung. Weitere Vorteile der Wiederverwendung von Komponenten in der Softwareentwicklung liegen in geringeren Kosten, die durch Zeitersparnis während der Entwicklung und der Qualitätssicherung begründet werden, einer schnelleren Entwicklung, weil die Komponenten nicht immer wieder von Neuem

erstellt werden müssen und einer schnelleren Qualitätssicherung, da die Risiken nur einmalig berücksichtigt werden müssen.

#### Übersichtliche Komponentenbibliothek

Komponenten sind jedoch nutzlos, wenn sie weder dokumentiert, noch aufgelistet sind. Aus diesem Grund sollte ein gut entworfenes MR-Autorensystem, das auf Komponenten basiert, über eine indizierte Bibliothek verfügen, in der alle vorhandenen Komponenten gesammelt sind. Dies ermöglicht Autoren eine Übernahme der Komponenten aus der Bibliothek.

Zur Vereinfachung sollte diese Bibliothek in einer Datenbank abgebildet sein. Das hat den Vorteil einer flexiblen Suchmöglichkeit und einer einfachen Abbildung der Beziehungen zwischen den einzelnen Komponenten. Weiterhin darf nicht außer Acht gelassen werden, dass jede Komponente bestimmte Parameter beziehungsweise Eigenschaften besitzt, die bei dessen Einsatz im MR-Autorensystem zur Verfügung gestellt werden. Ein Druckknopf besitzt beispielsweise eine Bezeichnung oder eine bestimmte Farbe, die jeweils als Eigenschaft beziehungsweise als Parameter des Druckknopfes bezeichnet wird. Diese Merkmale sollten vom Autor frei wählbar sein und dem Autor sollte eine Anpassungsmöglichkeit zur Verfügung gestellt werden. Für die Vereinfachung dieser Bereitstellung sollten so genannte Eigenschaftseditoren, am besten GUI-basiert, erstellt werden, die die Anpassung der Parameter jeder Komponente sicherstellen.

#### Flexibilität

Eine wichtige Anforderung an MR-Autorensysteme ist die Flexibilität des Systems. Damit ist gemeint, dass das MR-Autorensystem sich an die individuellen Bedürfnisse des Anwenders anpassen lassen soll. Es muss zum Beispiel den Benutzern die Möglichkeit bieten, ihre Anwendungen sowohl mit Hilfe von Skriptsprachen als auch ohne Programmierung zu erstellen.

#### Skalierbarkeit

Der Einsatz einer Software kann nur dann über längere Zeit gewährleistet werden, wenn sie die ständig wachsenden Datenmengen verarbeiten kann. Ein Beispiel für diese wachsenden Datenmengen ist die rasant voranschreitende Weiterentwicklung der multimedialen Technologien. Das hat zur Folge, dass die Datenmengen, die durch diese Technologien entstehen, an Qualität und Größe gewinnen. Da MR-Autorensysteme Schnittstellen zu multimedialen Technologien wie zum Beispiel zu einer Kamera besit-

zen, sollten diese Systeme mit der oben genannten Entwicklung Schritt halten können.

Einem gut skalierten MR-Autorensystem würde es einfacher fallen, der oben genannten Entwicklung zu entsprechen und mit diesem stetigen und ständigen Wachstum leichter umzugehen. Skalierbarkeit und Flexibilität machen ein MR-Autorensystem zu einer zukunftssicheren Investition, deren Kosten und Nutzen sich auch für die Zukunft abschätzen lassen. Sie gestalten ferner ein MR-Autorensystem wirtschaftlicher, wodurch es diesem System leichter sein wird, sich auf dem Markt zu behaupten.

#### Portierbarkeit

Das MR-Autorensystem sollte möglichst eine hohe Portierbarkeit besitzen. Aus Sicht der Autoren ist es sicherlich wichtig, dass das MR-Autorensystem auf unterschiedlichen Plattformen eingesetzt werden kann beziehungsweise auf verschiedene Plattformen portiert werden kann. Eine Plattform ist eine Kombination von Betriebssystemen wie UNIX [Ope04], Windows [Mic04] oder Mac OS [App04] und Computerhardware.

#### Installation/Deinstallation

Das MR-Autorensystem sollte auf jeder Plattform, auf der es zum Einsatz kommt, eine vollständige Installation beziehungsweise Deinstallation der Software für den Autor ermöglichen. Ferner sollte die De- beziehungsweise Installation unbeaufsichtigt durchgeführt werden können. Für die Realisierung dieser Anforderung können existierende Technologien eingesetzt werden wie für Microsoft Windows Systeme, die MSI-Technologie (Microsoft Windows Installer) [Mic04] oder RPM (RedHat Package Manager) [Red04] für RedHat Linux Systeme. Bei diesen Technologien handelt es sich um paketbasierende Installationsroutinen, die eine vollständige Deinstallation der Software gewährleisten. Ferner sollten ausreichende Dokumentationen für die De- beziehungsweise Installation vorhanden sein, damit Benutzer selbstständig das MR-Autorensystem de- beziehungsweise installieren können. Hierbei sollte kein Spezialwissen eines Systemadministrators notwendig sein.

#### Wirtschaftlichkeit

Bei einem MR-Autorensystem sollte der Punkt Wirtschaftlichkeit nicht außer Acht gelassen und möglichst stark gewichtet werden. Das heißt, eine MR-Anwendung sollte mit Hilfe des MR-Autorensystems kosteneffektiv produziert werden können. Durch den Einsatz eines MR-Autorensystems sollte der Benutzer schneller und effizienter arbeiten können.

## 3.3 Mixed Reality-Erstellungsmethoden und -werkzeuge

In diesem Abschnitt werden einige Erstellungsmethoden für die MR-Anwendungen vorgestellt und diskutiert. Die Erstellung von MR-Anwendungen ist eine sehr schwierige und komplexe Aufgabe [ADG04]. Solche Anwendungen werden meistens durch direkte Codierung in einer Programmiersprache wie zum Beispiel C++ erstellt. Es gibt inzwischen jedoch einige Autorensysteme, wie zum Beispiel AMIRE [AMI04d] oder PowerSpace [HR02], die die Erstellung solcher Anwendungen viel einfacher ermöglichen. Dadurch können die MR-Anwendungen visuell interaktiv ohne Programmierkenntnisse erstellt werden.

Wie von Abawi ([ADG04], [AD04]) erläutert, existieren für die Erstellung von MR-Anwendungen Schwierigkeiten, deren Ursachen in verschiedene Ebenen unterteilt sind. Dabei handelt es sich entweder um technologische Ebenen oder Task-, Prozess- und Validierungsebenen.

Die technologische Ebene muss sich der Herausforderung stellen, reale und virtuelle Bilder in Echtzeit zu verarbeiten und zusätzlich dazu Spezialhardware, deren Einsatz essentiell für diese Verarbeitung ist, zu unterstützen.

Auf der Taskebene besteht die Aufgabe darin, klare Autorenrollen zu definieren und die Grenzen dieser Rollen eindeutig festzulegen. Es gibt ein breites Spektrum an Aufgaben, die von Autoren zu bewältigen sind. Diese sind zum Beispiel Erstellung, Modellierung und Animation von virtuellen Objekten sowie deren Integration in das System. Kaum ein Autor ist in der Lage, alle diese Aufgaben alleine zu erfüllen. Um dieses Problem zu lösen, müssen verschiedene Autoren mit unterschiedlichen Kenntnissen eingesetzt werden. Da Autoren nicht alle denselben Kenntnisstand besitzen, müssen ihre Aufgabenbereiche klar, unmissverständlich und eindeutig definiert und ihnen zugeordnet werden.

Auf der Prozessebene besteht die Hauptschwierigkeit darin, dass bisher kein Prozess existiert, der die effiziente Erstellung einer komplexen MR-Anwendung beschreibt [AD04]. Es existieren Ansätze dieser Prozesse, beispielsweise bei ARToolKit [KBP00], Tiles [PTB+01] oder beim Authoring Toolkit von Fischer [Fis02], die zum Teil realisiert wurden und immer mehr Anerkennung finden. Diese Anerkennung ist ein Indiz für den Bedarf für einen solchen Prozess.

Auf der Testebene liegt die Hauptschwierigkeit im so genannten Previewproblem, das die Problematik der Integration der Sichtweise des Endnutzers in das MR-Autorensystem behandelt. Dadurch, dass ein bestimmter realer Ort ein Teil der MR-Umgebung ist, ist es besonders problematisch für einen Autor, die produzierte MR-Anwendung zu testen, weil er in diesem Fall vor Ort sein müsste [ADG04].

Um die oben genannten Schwierigkeiten zu beseitigen, wurden Autorensysteme speziell für die MR-Anwendungsproduktion entwickelt. Im Folgenden werden die Erstellungswerkzeuge PowerSpace, alVRed, Authoring Wizard für Mixed Reality Assembler Instructor, AR-PDA und AMIRE vorgestellt.

### 3.3.1 PowerSpace

Das AR-Autorensystem PowerSpace [HR02] erlaubt die Erstellung von AR-Anwendungen auf eine schnelle und einfache Weise. Dabei verwendet dieses System die Funktionalität von einem 2D Präsentationsprogramm wie Microsoft PowerPoint [Pow04] als Basis für die Zusammensetzung von 3D-Inhalten. Mit Hilfe dieses Systems können verschiedene AR-Anwendungen für ein breites Spektrum von Aufgabenbereichen erstellt werden, darunter auch für Trainings-, Reparaturanwendungen, Montageanleitungen, Benutzerhandbücher und Stadtführungen.

Das Konzept von PowerSpace lässt sich wie im Folgenden erklären: Zunächst findet eine Erstellung und Anordnung der Elemente im 2D-Format in Microsoft PowerPoint statt. Als nächstes erfolgt eine räumliche Anordnung der Elemente durch einen speziellen Editor namens PowerSpace-Editor auf der Grundlage des Folienkonzepts. Im nächsten Schritt werden die Reihenfolge und die Beziehungen zwischen den Folien innerhalb des PowerSpace-Editors festgelegt. Als letztes wird eine Auswertung der Ergebnisse durch den „PowerSpace-Betrachter“ erstellt.

Die Schnittstelle zwischen diesen Tools basiert auf der XML-Technologie [W3C04]. XML (Extensible Markup Language) ist eine Metasprache zur Beschreibung und Instanziierung von Auszeichnungssprachen für allgemeine Dokumente [CS01]. Dabei exportiert der Autor die Microsoft PowerPoint Präsentation mit Hilfe eines „Visual Basic for Applications“ (VBA) Makros in eine XML-Datei, die wiederum vom PowerSpace-Editor importiert und weiterverarbeitet wird. Das gleiche Prinzip gilt auch für den PowerSpace-Editor, der den Export in eine XML-Datei vornimmt und diese für den „PowerSpace-Betrachter“ zur Verfügung stellt. Der PowerSpace-Betrachter bietet die Möglichkeit, zwischen den einzelnen Folien der Präsentation und durch die 3D-Folien selbst zu navigieren [Har02].

Der PowerSpace-Editor besitzt viele strukturelle Ähnlichkeiten mit Microsoft PowerPoint, wie beispielsweise die Folienübersicht, die auf der linken Seite zu sehen ist und die Menüleiste, die oben in der Abbildung 3.6 angebracht ist. Dieser Editor besteht aus einer Befehlsleiste, mehreren Toolbars, einem Folienfenster, dem Betrachter und einer Statusleiste. Die Steuerelemente in der Menüleiste sind Microsoft Office Anwendungen sehr ähnlich, damit sich der Anwender in einer vertrauten Umgebung wieder findet. Die Menüleiste von PowerSpace und Microsoft PowerPoint ähneln sich sehr. PowerSpace ist jedoch um 3D-Optionen erweitert. Zusätzlich gibt es die

so genannte Funktion „sub slides view“, welche die Animationsschritte der einzelnen Folien anzeigt.

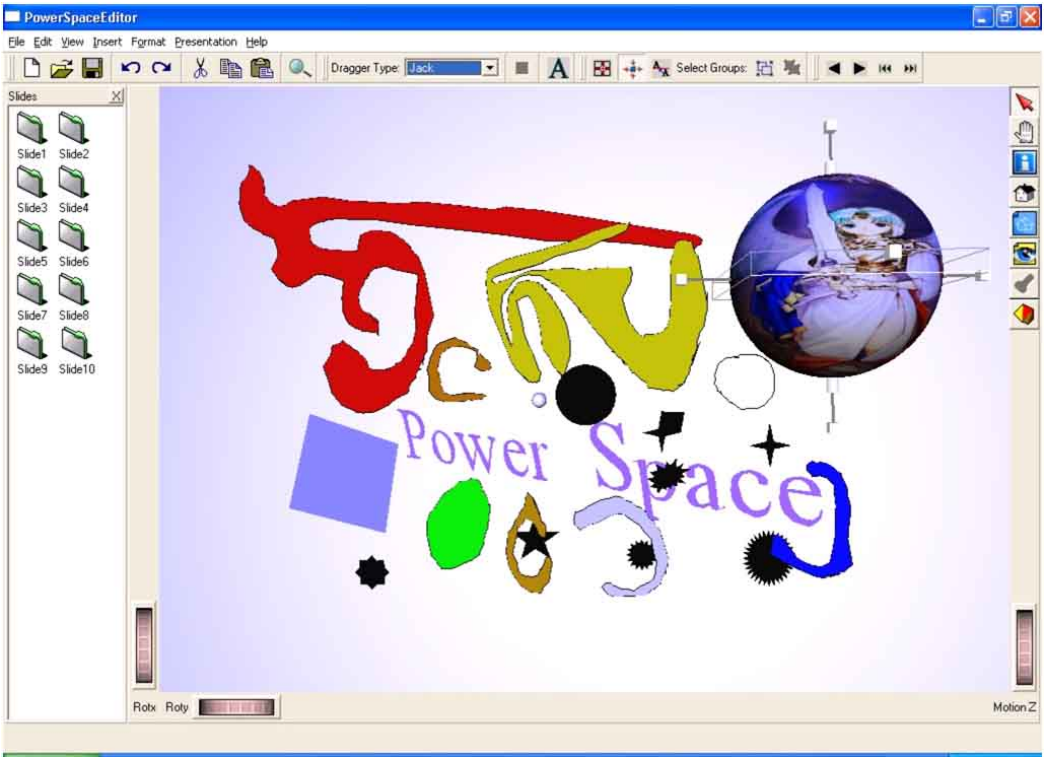


Abb. 3.6: PowerSpace-Editor [Har02]

Der PowerSpace-Editor bietet verschiedene Mechanismen an, um die aus PowerPoint importierten Daten zu bearbeiten. Im Folgenden werden einige von ihnen kurz vorgestellt.

### Auswahlmechanismus

Sobald ein Objekt vom Autor angeklickt wird, wird dieses im „PowerSpace-Betrachter“ markiert und es werden so genannte Manipulatoren, wie in Abbildung 3.7, um das Objekt angezeigt. Mit Hilfe dieser Manipulatoren kann das ausgewählte Objekt positioniert, skaliert und gedreht werden. Ferner ist eine Mehrfachmarkierung von Objekten möglich, was bei Objekten, deren Ausrichtung zueinander unverändert bleiben soll, vom großen Vorteil ist. Wenn mehrere Objekte ausgewählt werden, werden sie hervorgehoben und der Manipulator erscheint zentriert auf diesen Objekten, wie in Abbildung 3.8 zu sehen ist.

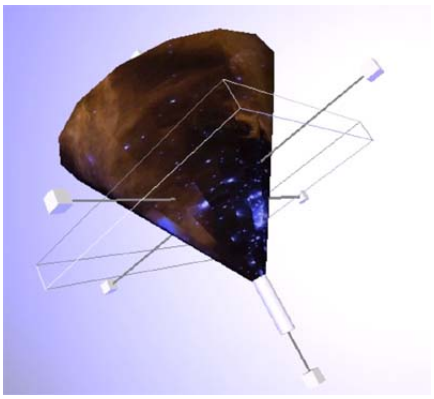

Abb. 3.7: Der Manipulator im PowerSpace-Editor [Har02]

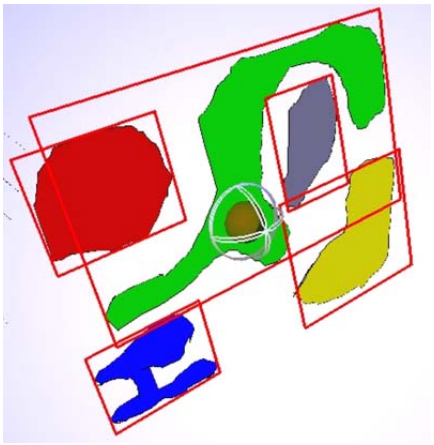

Abb. 3.8: Mehrfachauswahl von Objekten [Har02]

### Materialeditor

Dieser Editor, wie in der Abbildung 3.9 zu sehen ist, besteht aus einem Vorschaufenster und sechs Schiebereglern, die die Eigenschaften, wie Schärfe, Transparenz oder Spiegelung des ausgewählten Objektes verändern lassen [Har02].

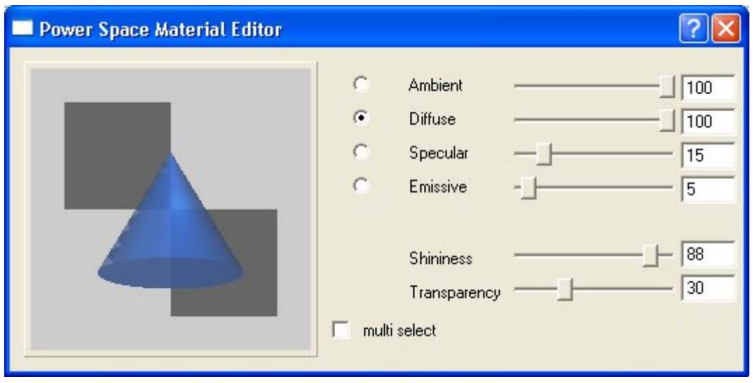


Abb. 3.9: Materialeditor [Har02]

### 3.3.2 Authoring Wizard

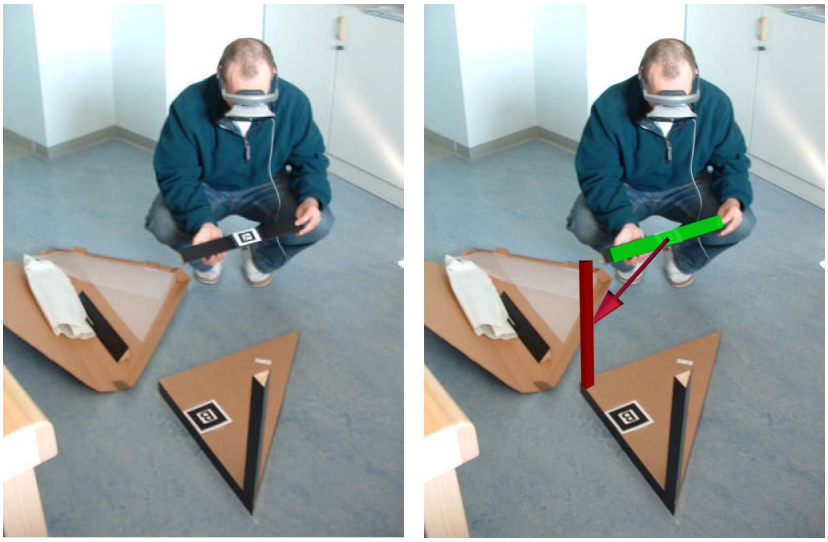

Abb. 3.10: MR Assembly Instructor (vgl. [ZHBH03])

Ein Problem, mit dem Anwender beim Zusammenbau von Möbeln häufig konfrontiert werden, ist das Lesen und Verstehen von komplizierten und meistens unübersichtlichen Handbüchern. Durch die MR-Technologie hofft man, eine Verbesserung in diesem Bereich der Industrie einführen zu können. Zauner und Haller [ZHBH03] stellen in ihrem Dokument ein Konzept für diese Verbesserung vor. Mit Hilfe vom „MR Assembly Instructor“ [Fai04] können Montageanleitungen einfach visualisiert werden. Dies soll als eine Alternative zu den herkömmlichen Bauanleitungen gelten. Dabei wird dem Benutzer Schritt für Schritt zum Beispiel das Zusammenbauen von Möbelstücken in einer sehr überschaubaren Art und Weise ermöglicht. Diese interaktiven Bauanleitungen werden

im Vergleich zu herkömmlichen Bedienungsanleitungen einfacher anwendbar sein, weil sie kontextbezogene und passende Informationen über den Zusammenbau von Möbelstücken anzeigen können.

Diese Informationen können in Form von Graphiken, Texte, 3D-Animationen und akustischen Signalen präsentiert werden. Das funktioniert so, dass das System die Schritte des Benutzers erkennt und dadurch den Überblick über das gesamte Geschehen behält. Hierbei muss der Benutzer ein Anzeigegerät tragen, wie im Unterkapitel 2.2 beschrieben. In Abbildung 3.10 wird der Fall dargestellt, wie ein Benutzer einen Tisch zusammenbaut. Auf der linken Seite der Abbildung wird die entsprechende Realität dargestellt, während auf der rechten Seite der Abbildung die Realität durch virtuelle Objekte erweitert wird. In dieser Abbildung soll der Benutzer auf der rechten Seite ein grün markiertes Bauteil in den rot markierten Bereich einsetzen, was durch einen roten Pfeil demonstriert wird.

Das System führt den Benutzer durch die Montageanleitung, wie in Abbildung 3.11 zu sehen ist. Beim Erkennen von realen Objekten wird anhand von den bereits erwähnten virtuellen Objekten angezeigt, wie der Benutzer vorgehen muss. Zusätzlich hat der Benutzer die Wahl, mehr Informationen über die einzelnen Bauteile aufzurufen. Zu dem grün markierten Objekt auf der linken Seite können weitere Informationen abgerufen werden, was in der rechten Abbildung angezeigt wird.


Abb. 3.11: GUI der Montageanleitung [ZHBH03]

„MR Assembly Instructor“ wird mit Hilfe von einem mächtigen und einfach anwendbaren Autorensystem namens „Authoring Wizard“ erstellt. „Authoring Wizard“ erlaubt den Autoren eine schnelle Implementierung des „MR Assembly Instructors“. In diesem Autorensystem werden Anwendungen auf Komponentenbasis erstellt. Dabei kann der Autor unter vielen Komponenten wählen und Beziehungen zwischen ihnen definieren. Dieser Ansatz ermöglicht es Autoren, die keine Programmierkenntnisse haben mit Hilfe dieses Werkzeugs eine auf MR-Technologie basierende Bauanleitung zu

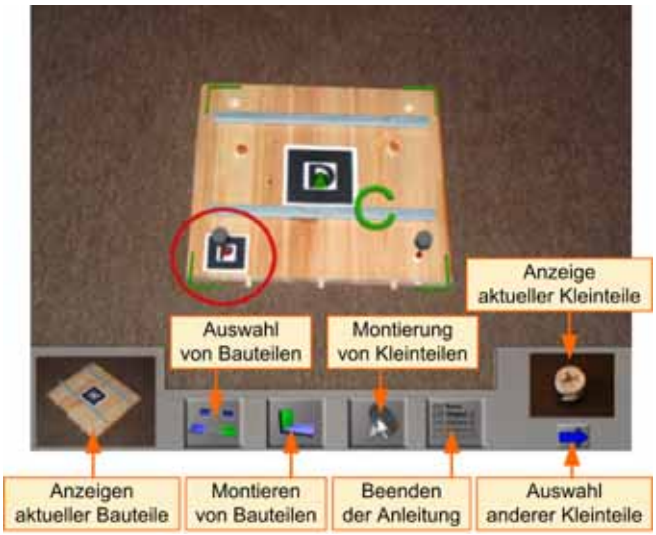


Abb. 3.12: Oberfläche von MR Assembly Instructor (vgl. [ZHBH03])

erstellen. Ferner ist noch zu erwähnen, dass dieses Werkzeug XML-Technologien einsetzt. Der Autor verwendet Authoring Wizard, um eine XML-basierte MR-Anwendung für „MR Assembly Instructor“ zu erzeugen, die dann in ein Framework geladen werden kann. Die MR-Anwendung besteht also aus einer Beschreibung in XML, die von einem Framework gelesen und interpretiert wird.

Dieses Autorensystem ist auf das markerbasierte Trackingsystem „ARToolKit“ aufgebaut, siehe Unterkapitel 2.2. Die Marker werden auf die unterschiedlichen Bauteile angebracht, welche die eindeutige Erkennung dieser Teile ermöglicht. Für die Erstellung kann der Autor am Anfang die Beziehung zwischen „ARToolKit-Marker“ und den realen Objekten definieren. Nun kann der Autor mit dem Erstellungsprozess beginnen.

Der Zusammenbau eines Möbelstücks besteht im Prinzip aus zwei immer wiederkehrenden Vorgänge:

- Befestigung der Kleinteile wie zum Beispiel die Schrauben an ein Bauteil
- Zusammensetzung verschiedener Bauteile

Für den Zusammenbau wird eine bestimmte Vorgehensweise definiert, welche der Autor bei der Erstellung der Anleitung einhalten soll. Zunächst muss ein so genanntes „Basisobjekt“ gewählt werden, das als Grundlage für die Installation der anderen Objekte gebraucht wird. Danach wird das nächste Objekt ausgewählt, das zu diesen „Basisobjekt“ passt. Dieses muss dann richtig positioniert und an dem zuvor gewählten „Basisobjekt“ montiert werden. Das neu zusammengesetzte Bauteil gilt nun als „Basisobjekt“ für weitere Schritte. Dieser Prozess wird iterativ solange durchgeführt, bis

das Möbelstück aufgebaut ist. In Abbildung 3.12 wird ein Beispiel von der Oberfläche dieses Autorensystems vorgestellt.

Zur Unterstützung der Autoren werden spezielle Werkzeuge, wie zum Beispiel das „Placement tool“ oder die „Placement Animation“, zur Verfügung gestellt. Diese helfen dem Autor bei der Positionierung und dem Einsatz von 3D-Objekten. „Placement tool“ stellt Funktionen wie Skalierung, Drehung und Verschiebung von 3D-Objekten zur Verfügung. Mit Hilfe von „Placement Animation“ kann sich der Autor die richtige Positionierung von 3D-Objekten anzeigen lassen. Dieses Werkzeug eignet sich insbesondere zur Bearbeitung von symmetrischen Bauteilen, die unter Umständen nicht eindeutig in ihrer Positionierung zu erkennen sind. In Abbildung 3.13 wird mit Hilfe einer kleinen Animation die korrekte Platzierung von einem Holzbrett dargestellt.

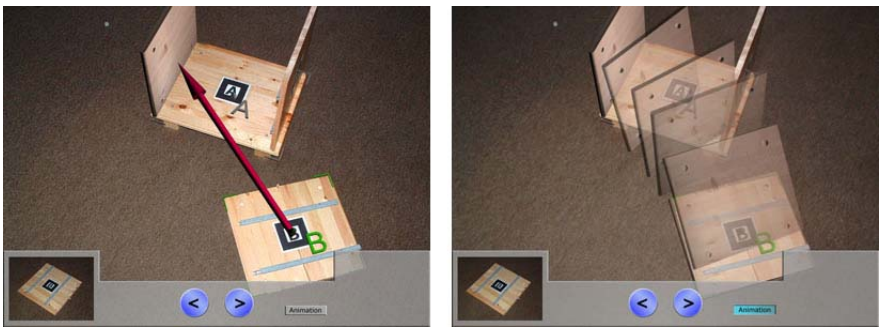

Abb. 3.13: Animation in MR Assembly Instructor [Fai04]

### 3.3.3 alVRed

Das Projekt alVRed [alV04] hat das Ziel, Methoden und Werkzeuge zur Erzeugung interaktiver, nichtlinearer und komplizierter Inhalte in VR-Umgebungen bereitzustellen. Dabei setzt dieses Werkzeug auf das VR-Framework „Avango“ auf, welches ein Software-Framework zur Erstellung virtueller Szenarien ist. Zur Verbesserung der Kommunikation zwischen Autoren, Programmierern und Modellierern wurden folgende Werkzeuge zur Verfügung gestellt:

- „VR-Authoring Tool“: Ein Autorensystem zur Modellierung der nichtlinearen Szenarien.
- „VR-Previewer“: Ein Vorschau-Werkzeug zur Visualisierung der modellierten dreidimensionalen Welt.
- „VR-Tuner“: Zur Feinabstimmung fertiger VR-Produktionen durch Autoren.

Die mit Hilfe des VR-Authoring Tools definierten Inhalte werden durch ein Interpreter automatisch mit Hilfe von XML-Dateien in das Avango-Framework übertragen. Im Folgenden werden die Werkzeuge näher beschrieben.

### VR-Authoring Tool

Die Kernanforderung an das VR-Authoring Tool [WGT+02] ist die Bereitstellung eines einfach zu bedienenden Werkzeuges für Autoren mit wenigen oder keinen Programmierkenntnissen, um ihre Vorstellung von einem nichtlinearen Szenario in einen Computer zu transformieren. Dazu bietet dieses Werkzeug dem Autor die Möglichkeit an, durch einfaches Drag & Drop ein nichtlineares Szenario in Form eines verzweigt gerichteten Graphen namens „Storygraph“ zu kreieren. In jedem Knoten des Graphen werden die für diesen Knoten wichtigen Beschreibungen als Text und in Form von angehängten Beispieldateien wie Bild, Audio und Video in den gängigen Formaten abgelegt. Dies geschieht im so genannten „Storyboarding“, welches als eine Kernfunktion des Autorensystems genannt werden kann. Modellierer können später in einem integrierten Media-Player diese Informationen abrufen. Somit bekommen sie ein klares Bild von den Vorstellungen des Autors. Alle für das Szenario bedeutsamen Objekte werden in Knoten des gerichteten Graphen angelegt.

Die eigentliche Szenariologik wird durch die Verbindung von Objekten mit Hilfe einer einfachen Scriptsprache aufgebaut. Durch den integrierten Player kann der Autor das entworfene Szenario zu Testzwecken abspielen lassen, bevor die Bilder der virtuellen Welt modelliert werden. Diese Möglichkeit zum Testen ist wichtig, da das Szenario rechtzeitig auf Konsistenz überprüft werden soll. Der linke Teil der Abbildung 3.14 zeigt ein Beispiel von einem solchen Storygraph. Auf der rechten Seite der Abbildung wird ein Beispiel für „Storyboarding“ angezeigt.

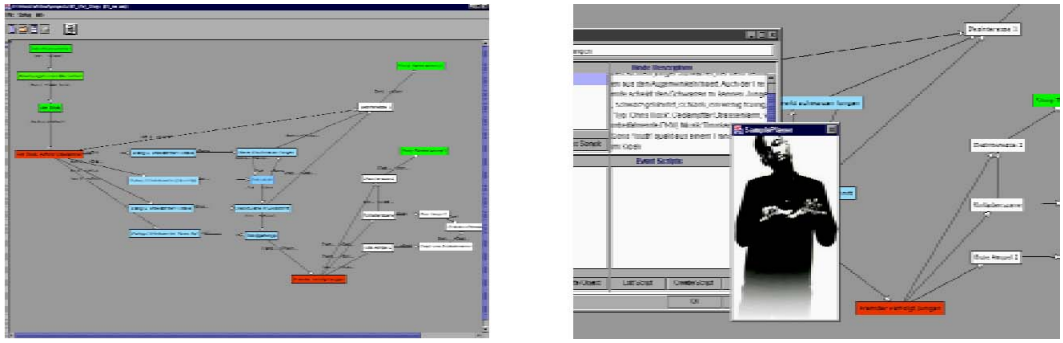

Abb. 3.14: Storygraph und Storyboarding von alVRed [WGT+02]

### VR-Previewer

Ein großes Problem in der Entstehung von VR-Inhalten liegt darin, dass es bisher wenige Möglichkeiten gab, schnell einen visuellen Eindruck von den Inhalten zu bekommen. Bisher war es so, dass in der Anfangsphase der Produktion nur Zeichnungen, Erklärungen oder Beispiele aus früheren Produktionen einen ersten Eindruck von dem zu erwartenden Endprodukt geben konnten. Der Grund dafür liegt in der Komplexität der VR-Szenen. Für die Erstellung einer VR-Szene werden bestimmte 3D-Modelle benötigt, die nicht als Skizzen entstehen können. Bis jedoch alle benötigten 3D-Modelle aufgebaut werden, vergeht sehr viel Zeit.

Der innerhalb des Projektes alVRed entwickelte VR-Previewer verfolgt das Ziel der Verkürzung der Zeitspanne zwischen den Ideen und deren ersten Visualisierung. Durch den VR-Previewer soll die parallele Entwicklung von 3D-Modellen und die Programmierung der MR-Anwendung ermöglicht werden. Mit einem GUI „Mini-Modeler“ können die bereits bestehenden Objekte als Vorlage von Geometrien oder Audio-Dateien in eine dreidimensionale Welt eingefügt werden. Sie können dann im Nachhinein in einem iterativen Prozess durch andere Objekte ersetzt werden. Damit wird es dem Kunden ermöglicht, in einer kurzen Zeit einen ersten Überblick von den Inhalten zu erhalten. Abbildung 3.15 zeigt ein Beispiel von VR-Previewer.

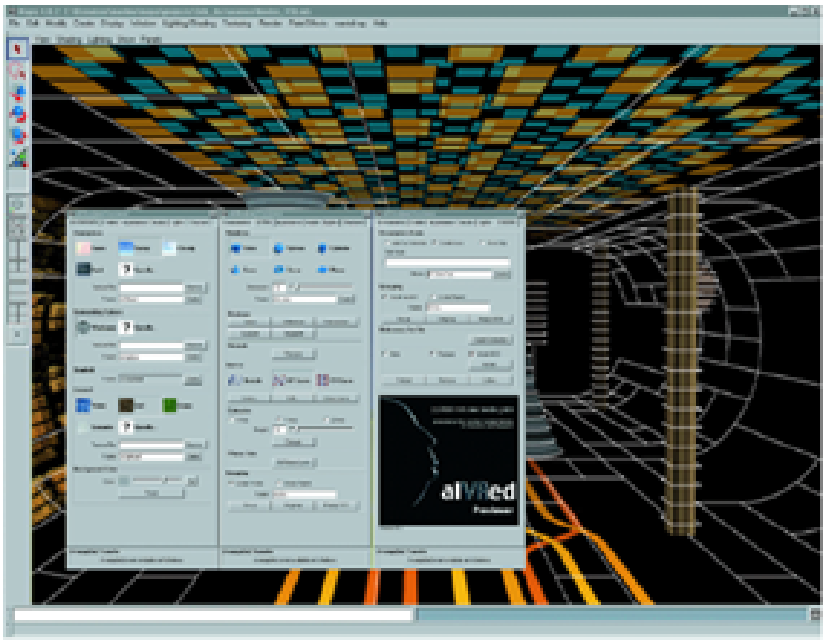


Abb. 3.15: VR-Previewer mit GUI-Elementen [BLM+04]

#### VR-Tuner

Der VR-Tuner ist ein Werkzeug, mit dem kreative Anwender ohne Programmier- oder Expertenkenntnisse aus dem Bereich VR eine programmierte Szene an ihre exakten Vorstellungen anpassen können. Dadurch kann der gesamte Arbeitsprozess schneller werden. Mit Hilfe einer GUI können zum Beispiel Änderungen, wie die Farbe eines Objektes, die Transformation von Objekten und Attribute aus der Bildbearbeitung für einzelne Objekte durchgeführt werden. Diese können dann sofort in der dreidimensionalen Umgebung sichtbar gemacht werden.

### 3.3.4 AR-PDA

Bei AR-PDA (Augmented Reality - Personal Digital Assistant) [ARP04] handelt es sich um ein Projekt, das vom Bundesministerium für Bildung und Forschung von 2001 bis 2004 gefördert wird. Das Ziel dieses Projektes ist die Realisierung eines Systems, das Verbrauchern bei ihren alltäglichen Aufgaben durch AR unterstützen soll. Dabei kommen derzeit als Endgeräte PDAs zum Einsatz, die mit einer Kamera ausgestattet sind. Auf diesen werden die realen Aufnahmen mit Erweiterungen wiedergegeben, die auf einem zentralen Server hinzugefügt werden. Diese Erweiterungen können 3D-Modelle, Grafiken und Audiodateien sein.

Das System funktioniert so, dass der Benutzer die integrierte Kamera des PDAs auf das gewünschte Objekt in seiner Umgebung richtet. Das aufgenommene Videosignal wird per Funk, zum Beispiel Mobilfunk oder drahtlose Netzwerke, an einen zentralen Server gesendet, der das Video analysiert. Dabei wird das Objekt vom Server erkannt und mit den entsprechenden kontextsensitiven Informationen, wie Videos, Audios, Texte und Graphiken erweitert. Demnach wird das Video an PDA zurückgeschickt und für den Benutzer visualisiert [GKR+01]. Abbildung 3.16 stellt genau diesen Vorgang dar.

Das Gesamtkonzept von AR-PDA besteht aus den folgenden Komponenten: AR-Server, AR-Client, AR-Anwendung, AR-Autorensystem.

Abb. 3.16: Funktionsprinzip des AR-PDAs [GKR+01]

#### AR-Server

Auf dem AR-Server werden alle rechenintensiven Vorgänge, wie Tracking und Bildverarbeitung durchgeführt, um an die Clients so wenige Anforderungen wie möglich bezüglich der Rechenleistung stellen zu müssen [EKFM02]. Dadurch wird der AR-Server

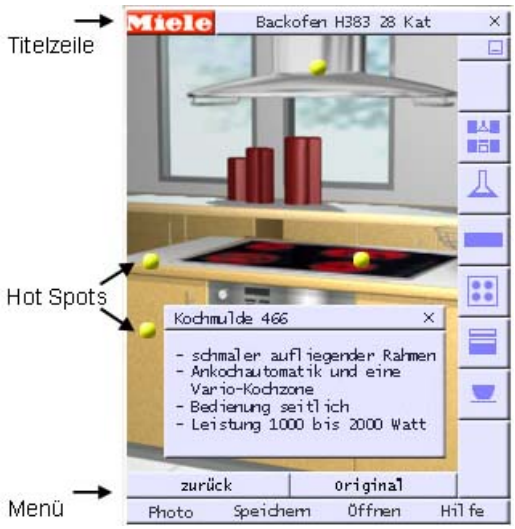


Abb. 3.17: Anwendungsoberfläche des AR-PDAs [PKFM04]

zum Herzstück des kompletten Systems.

### AR-Client

Als Client dient ein herkömmlicher PDA, der über eine integrierte Kamera verfügt und hauptsächlich als Anzeige der Daten fungiert, die vom AR-Server zur Verfügung gestellt werden. Zusätzlich kann er die Interaktionen des Benutzers verarbeiten [EKFM02].

### AR-Anwendung

Die AR-Anwendung ist der Teil, der für die Erweiterung der Realität mit Zusatzinformationen verantwortlich ist. Für den Verkauf von Backöfen wurde im Rahmen des Projektes AR-PDA eine Beispielanwendung erstellt, die über zwei Interaktionsmechanismen gesteuert wird. Ein Mechanismus besteht aus kontextbezogenen Interaktionselementen, auch „Hot Spots“ genannt, die grafisch wie kleine gelbe Punkte augmentiert werden. Bei einem Klick auf ein „Hot Spot“ werden dem Benutzer kontextbezogene Informationen angezeigt. Der andere Mechanismus ist die Steuerung der Anwendung über die Tasten des PDAs. Abbildung 3.17 zeigt die Oberfläche dieser Anwendung.

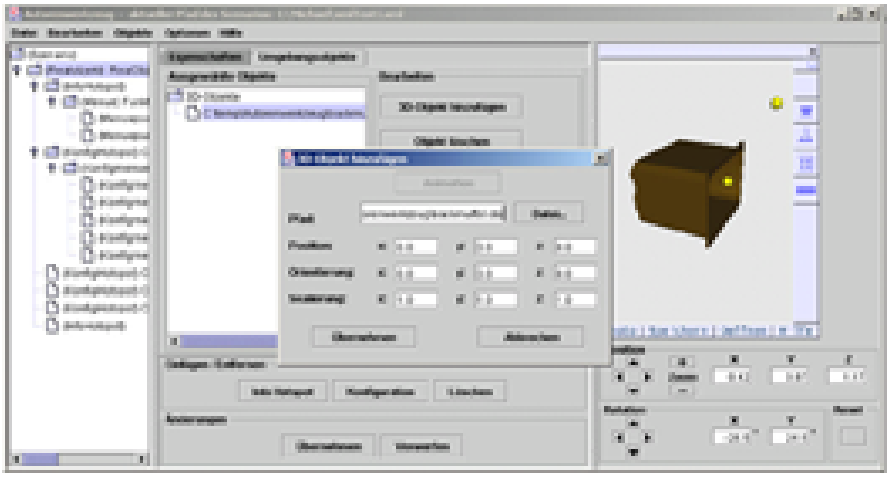

Abb. 3.18: AR-Autorensystem des AR-PDAs [PKFM04]

### AR-Autorensystem

Das AR-Autorensystem ist der Teil, der die Erstellung von AR-Anwendungen bewirkt. Dabei kommt ein benutzerfreundliches komponentenbasiertes System zum Einsatz, das als ein Kernelement des Projektes gilt. Die Vorgehensweise zur Erstellung der

Anwendung wird hier in drei Schritten durchgeführt.

- Schritt eins: Akquisition benötigter Daten für die Augmentierung in der AR-Anwendung.
- Schritt zwei: Vorbereitung zur Integration dieser Daten in die Anwendung.
- Schritt drei: Erstellung der Anwendung durch den Einsatz der vorbereiteten Daten aus Schritt zwei.

Für den Erstellungsprozess steht den Autoren ein dediziertes Entwicklungswerkzeug zur Verfügung, in dem notwendige Elemente für die Anwendung ohne detaillierte Programmierkenntnisse erstellt werden können. Dies geschieht über ein so genanntes Baukastenprinzip, bei dem eine bestimmte Menge von Bausteinen zur Verfügung stehen und vom Autor in die Anwendung eingebettet und konfiguriert werden können [GKR+01]. Das AR-Autorensystem unterstützt bei der Erstellung der Anwendung viele mögliche Quellen, wie Texte, 2D-Graphiken, 3D-Modelle, Audios und Videos [PKFM04]. Abbildung 3.18 zeigt die Oberfläche des AR-Autorensystems.

### 3.3.5 AMIRE

AMIRE (Authoring Mixed Reality) [AMI04d] bezeichnet ein von der EU gefördertes Projekt*, dessen Ziel die Entwicklung von Vorgehensweisen ist, mit deren Hilfe eine effiziente Erstellung und Anpassung von MR-Anwendungen ermöglicht wird [ADHZ04]. Das Projektkonsortium hat ein gleichnamiges MR-Autorensystem entwickelt, das hier vorgestellt und näher beschrieben wird.

Das Konzept von AMIRE sieht vor, den verschiedenen Autoren einer MR-Anwendung bestimmte Aufgaben zuzuteilen. Dabei wird jedem Autor eine eigene Rolle zugeteilt. Dadurch können Autoren auf der Anwendungsebene ohne Programmierung und Entwickler des Systems mit Programmierung an der Gesamtproduktion beteiligt werden.

Das Hauptmerkmal von AMIRE ist auf die komponentenbasierte Entwicklung von MR-Anwendungen gerichtet [DGHP02]. Dabei hat der Begriff Komponente je nach Autorenebene unterschiedliche Bedeutungen [ADG04]. Auf der technischen Ebene kann eine Komponente ein Algorithmus oder eine Komponente im softwaretechnischen Sinne sein, wie im Unterkapitel 3.2.2 erläutert. Auf der Anwendungsebene wird unter Komponente ein Objekt verstanden, das durch den Autor in die MR-Anwendung implementiert werden kann. Komponenten haben in AMIRE mindestens zwei Schnittstellen, durch die sie miteinander verbunden werden können. Diese sind die so genannten „In-Slots“ für eingehende Verbindungen und „Out-Slots“ für ausgehende Verbindungen.

*Projektnummer: IST-2001-34024

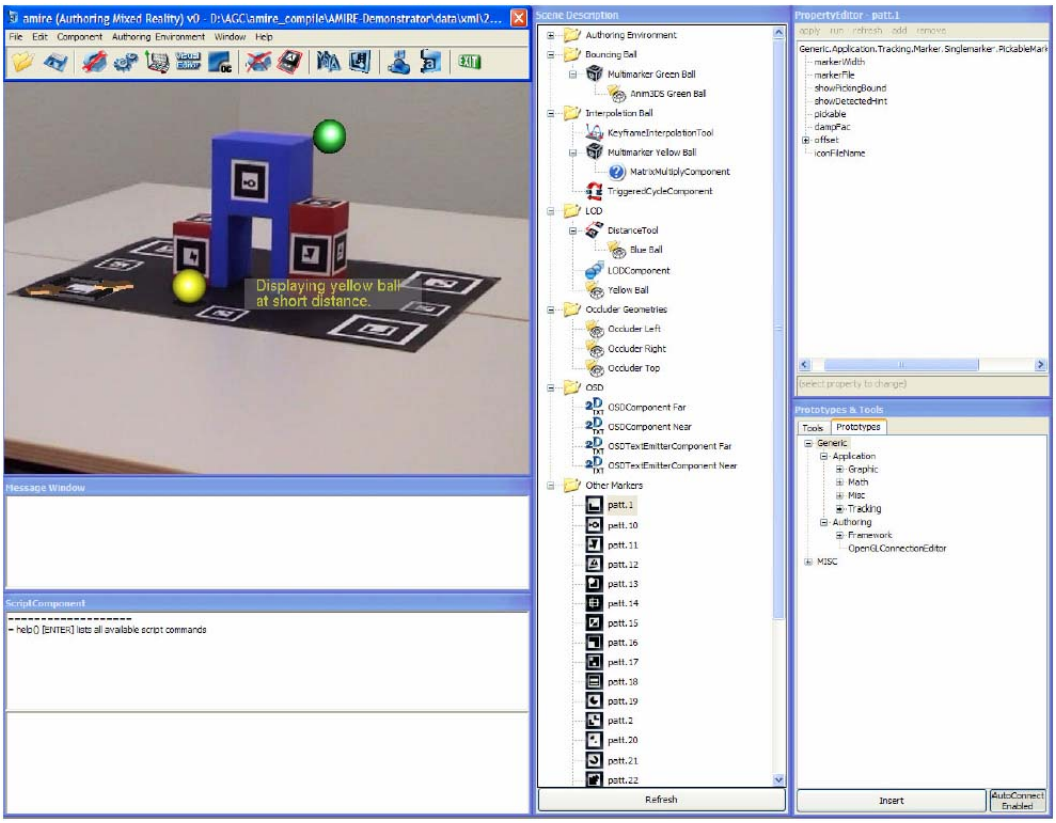


Abb. 3.19: Die Autorenumgebung von AMIRE [ARD04]

AMIRE setzt als Trackingsystem das markerbasierte „ARToolKit“ ein, siehe Unterkapitel 2.2. Damit dieses Trackingsystem richtig funktioniert, müssen in der realen Umgebung Marker vorhanden sein, was bereits im Unterkapitel 2.2 erklärt wurde. In AMIRE werden Marker einzeln hinzugefügt und als Komponenten in die produzierte MR-Anwendung eingebunden.

Der AMIRE-Autorenprozess führt den Autor Schritt für Schritt durch den Arbeitsprozess. Es ist wichtig, den Autor von den technischen Aspekten fernzuhalten. Es werden verschiedene Autorenrollen definiert, die die Aufteilung des gesamten Arbeitsprozesses ermöglichen [ADG04].

Die Entwicklung einer MR-Anwendung wird bei AMIRE in vier Phasen aufgeteilt:

- „Qualifikation“, in der die Komponenten ausgesucht werden.
- „Adaption“, in der die Eigenschaften der Komponenten angepasst werden.
- „Kombination“, in der die Komponenten miteinander verbunden werden.
- „Kalibrierung“, in der die Beziehungen zwischen den realen und virtuellen Objekten beschrieben werden.

Der Ablauf dieser Phasen muss nicht unbedingt in dieser Reihenfolge eingehalten werden. Dadurch kann jede erstellte MR-Anwendung durch Iteration dieser Phasen erweitert und gewartet werden.

AMIRE besitzt eine GUI, die ähnlich aufgebaut ist wie gängige Standardsoftware, zum Beispiel Windows-Programme. Die AMIRE-Autorenumgebung bietet dem Autor die Möglichkeit, die MR-Anwendung aus Sicht des Endbenutzers zu sehen und in dieser Sicht die MR-Anwendung zu erstellen. Dabei kann die Anwendersicht durch eine zuvor aufgenommene Video-Datei wiedergegeben werden. AMIRE bietet auch Werkzeuge zur Steuerung dieser Video-Dateien an, die bei der Erstellung und Validierung von Anwendungen vorteilhaft zum Einsatz kommen. Des Weiteren werden mehrere Dialogfenster mit genügend Hilfsmitteln für die Erstellung der MR-Anwendung dem Autor zur Verfügung gestellt. In Abbildung 3.19 wird die AMIRE-Autorenumgebung mit integrierter Sichtweise des Endbenutzers dargestellt.

In AMIRE ist es Autoren möglich eine MR-Anwendung aufzubauen, ohne Programmierkenntnisse einzusetzen. Bei der Erstellung der MR-Anwendung kann der Autor auf Standardwerkzeuge wie Microsoft Visio und Alias Maya zurückgreifen, weil diese von AMIRE unterstützt werden. Das hat zur Folge, dass der Kreis der in Frage kommenden Autoren vergrößert wird. Viele Autoren sind bereits mit diesen Werkzeugen vertraut und können ihre Erfahrungen für die Erstellung von MR-Anwendungen einbringen. Abbildung 3.20 stellt eine Beispielanwendung dar, die in Microsoft Visio für AMIRE angefertigt wurde. Dabei kommuniziert Microsoft Visio mit AMIRE über eine spezielle Schnittstelle, die extra dafür erstellt wurde.

Zusätzlich stellt AMIRE eine Skriptsprache zur Verfügung, mit deren Hilfe Anwendungen schnell und mit wenig Aufwand realisiert werden können. Die Skripte können in einem gängigen Texteditor erstellt und in AMIRE importiert werden.

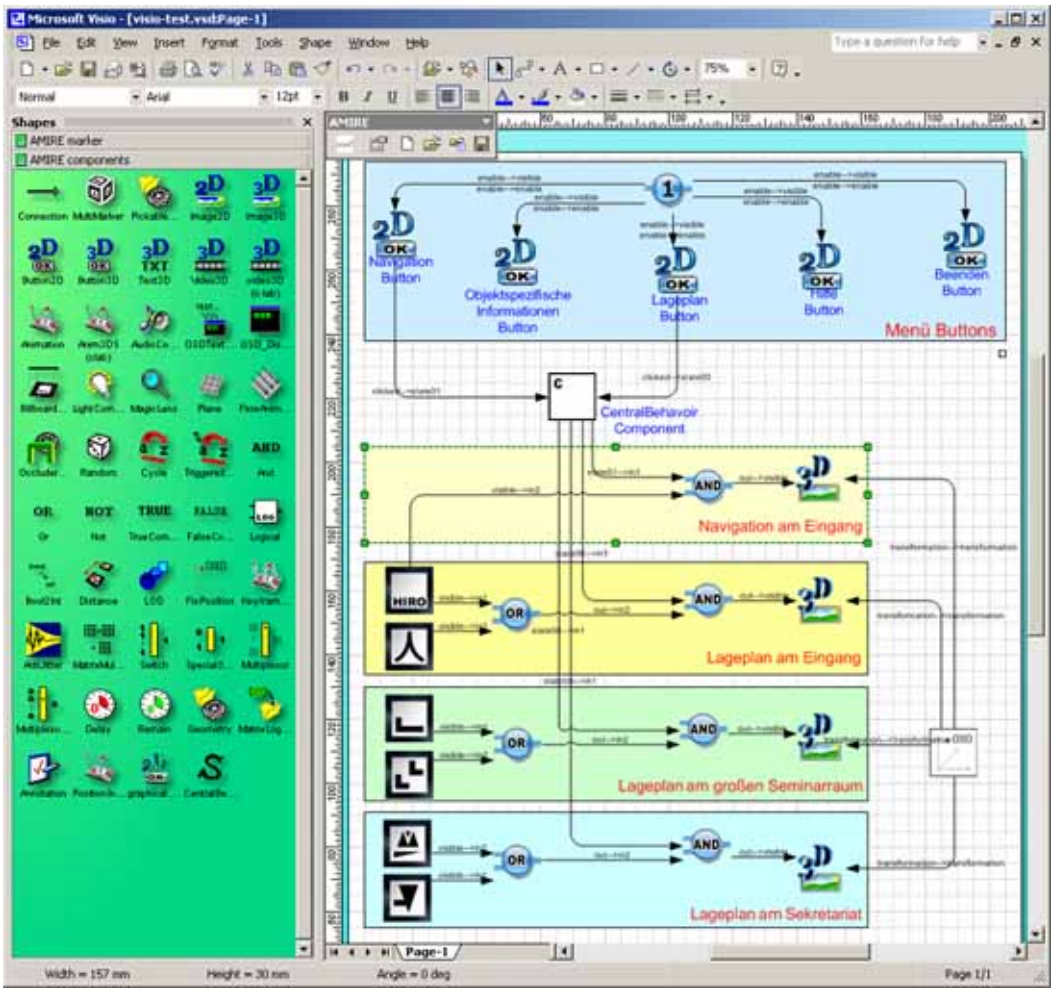


Abb. 3.20: MR-Anwendung von AMIRE, dargestellt in Microsoft Visio

## 3.4 Toolauswahl

Im Folgenden wird das Kapitel Analyse zusammengefasst und die Ergebnisse ausgewertet. Da die Entwicklung von MR-Anwendungen eine schwere und komplizierte Aufgabe ist, wurden spezielle MR-Autorensysteme entwickelt, die eine einfache Erstellung von MR-Anwendungen ermöglichen sollen. Um dieses Ziel zu erreichen, müssen MR-Autorensysteme spezielle Anforderungen im Bereich MR erfüllen. Diese Anforderungen wurden hier definiert und in obligatorische und fakultative Anforderungen aufgeteilt. Basierend auf diesen Anforderungen wurden einige bekannte Autorensysteme für MR-Anwendungen untersucht und analysiert. In diesem Unterkapitel wird eine Bewertung dieser Autorensysteme durchgeführt. Unter Berücksichtigung des Ergebnisses dieser Analyse soll ein geeignetes Autorensystem für die Erstellung einer MR-Anwendung aus-

gesucht werden. Dabei werden die wichtigsten Anforderungen an MR-Autorensysteme, die im Unterkapitel 3.2 erläutert wurden, als Bedingungen definiert. Bei den Recherchen wurden nicht alle Informationen bezogen auf die gestellten Anforderungen in Tabelle 3.1 gefunden und werden als unbekannt bezeichnet, weil in den Dokumentationen nur erfüllte Anforderungen Erwähnung finden. Bei der Entscheidung werden nur Anforderungen berücksichtigt, die erfüllt werden können.

Das Ergebnis dieser Entscheidungstabelle kann dem Autor bei der Suche nach einem geeigneten Werkzeug für die Erstellung einer MR-Anwendung behilflich sein. Für die Umsetzung der in dieser Arbeit zu realisierende MR-Anwendung wird AMIRE ausgewählt, da AMIRE die meisten der oben definierten Anforderungen erfüllt.

AMIRE bietet eine graphische Benutzeroberfläche, eine Möglichkeit zur Vorschau und verschiedene dedizierte Werkzeuge für die einfache Erstellung der MR-Anwendung an. Außerdem wird in AMIRE eine Skriptsprache verwendet, die den Erstellungsprozess beschleunigen kann. Die für die MR-Anwendung benötigten Audio-, Video-, Graphik- und Text-Dateien können extern erstellt und in AMIRE eingesetzt werden. Ein weiterer großer Vorteil von AMIRE ist die Aufteilung der Autorenrollen. Dadurch können verschiedene Autoren mit unterschiedlichen Kenntnissen an der Erstellung der MR-Anwendung zusammenarbeiten. Mit Hilfe von AMIRE kann die MR-Anwendung einfach erweitert und verbessert werden, weil AMIRE die Eigenschaft der Abwärtskompatibilität besitzt.

Im vorangegangenen Kapitel wurden Autorensysteme analysiert und bewertet. Als Ergebnis wurde AMIRE als das Autorensystem ausgesucht, das die meisten Anforderungen erfüllt. Im nächsten Kapitel wird eine MR-Anwendung spezifiziert. Diese soll unabhängig von dem hier ausgesuchten Autorensystem stattfinden, weil die Umsetzbarkeit der MR-Anwendung mit dem ausgesuchten MR-Autorensystem keine relevante Rolle spielt.

| | PowerSpace | Authoring Wizard | alVRed | AR-PDA | AMIRE |
|---|---|---|---|---|---|
| GUI | erfüllt | erfüllt | erfüllt | erfüllt | erfüllt |
| Vorschau | erfüllt | erfüllt | erfüllt | unbekannt | erfüllt |
| Unterstützung externer Dateien | erfüllt | unbekannt | erfüllt | erfüllt | erfüllt |
| Dedizierte Werkzeuge | erfüllt | unbekannt | erfüllt | unbekannt | erfüllt |
| Klar definierte Autorenrollen | unbekannt | unbekannt | erfüllt | unbekannt | erfüllt |
| Skriptunterstützung | unbekannt | unbekannt | erfüllt | unbekannt | erfüllt |
| Wartbarkeit | erfüllt | unbekannt | unbekannt | erfüllt | erfüllt |
| Abwärtskompatibilität | unbekannt | unbekannt | unbekannt | unbekannt | erfüllt |

Tabelle 3.1: Vergleich der Eigenschaften von MR-Autorensystemen

# Kapitel 4

# Spezifikation einer Mixed Reality-Anwendung

In diesem Kapitel wird eine MR-Anwendung beschrieben, deren Funktion die Realisierung eines Rundganges im Gebäude der Professur für Graphische Datenverarbeitung (GDV) der Goethe-Universität Frankfurt am Main ist, was im Folgenden mit „Professur für GDV“ abgekürzt wird. Im Laufe dieser Arbeit wird diese Anwendung als „MR-Referenzanwendung“ bezeichnet. Bevor im Folgenden genauer auf die Spezifikation dieser MR-Anwendung eingegangen wird, wird zunächst der Begriff Softwarespezifikation näher erläutert.

Wie bereits im Unterkapitel 2.1 definiert, kann eine Spezifikation als ein Abkommen zwischen Hersteller und Verbraucher einer Dienstleistung angesehen werden [GJM91]. Die Softwarespezifikation legt fest, welche Funktionen von dem System verlangt werden ([Som01], [SG84]). Eine sorgfältige Durchführung einer Spezifikation ist von großer Bedeutung, weil sie als Basis einer erfolgreichen Softwareentwicklung betrachtet werden kann. Sie kann sogar eine rasche und wirtschaftliche Herstellung eines Softwareprodukts positiv beeinflussen. Die Softwarespezifikation sollte möglichst für Kunden und Programmierer verständlich sein.

Nach der Definition von Zeller [Zel03] sollte jede Spezifikation vollständig sein, um jeden Aspekt des Systemverhaltens abzudecken. Weiterhin sollte sie folgende Anforderungen erfüllen: Sie sollte einerseits widerspruchsfrei sein, um klar zu stellen, was implementiert werden soll, und andererseits vorausschauend sein, um unvorhergesehene Umstände abzufangen, damit die Robustheit gesteigert wird.

Die „MR-Referenzanwendung“ wird hier fachlich spezifiziert ohne auf ihre technische Realisierung zu achten, das heißt es wird nur erläutert, was das System leisten soll und nicht etwa wie es realisiert werden soll. Dabei werden die Ziele und die Anforderungen an das System neben den Funktionen und den Einschränkungen für die „MR-Referenzanwendung“ beschrieben. Es werden Anwendungsfälle erläutert, die einen Teil der Anforderungen an das System beschreiben. Anwendungsfälle sind ein wichtiges Hilfsmittel bei der Spezifikation und werden im folgenden Kapitel ausführlich erläutert.

Im Unterkapitel 4.1 werden Anwendungsziele und Anforderungen an die „MR-Referenzanwendung“ definiert und später spezifiziert. Im Unterkapitel 4.2 werden alle Funktionen der „MR-Referenzanwendung“ ohne Bezug zur Benutzeroberfläche vorgestellt. Dies wird mit Hilfe von Funktionsdiagrammen verdeutlicht. Im Unterkapitel 4.3 werden Anwendungsfälle beschrieben, die die Interaktion zwischen Benutzern und dem System abbilden. Die Anwendungsfälle werden mittels Abbildungen veranschaulicht. Im Unterkapitel 4.4 werden die Benutzeroberflächen der „MR-Referenzanwendung“ für alle Funktionen aus Unterkapitel 4.2 anhand von Entwurfsbildern erläutert.

## 4.1 Ziele und Anforderungen

Mit Hilfe dieser „MR-Referenzanwendung“ sollen die Besucher die Möglichkeit bekommen, durch das Gebäude geführt zu werden und dabei aktuelle und nützliche Informationen zu erhalten. Dadurch soll der Besuch des Gebäudes einfacher und informativer gestaltet werden. Der Anwender erhält beim Betreten des Gebäudes einen tragbaren Computer mit einer Kamera. Durch die „MR-Referenzanwendung“ kann der Benutzer durch das Gebäude geführt werden. Dabei kann das System sich im Gebäude selbstständig orientieren und gespeicherte Informationen über den Standort des Anwenders zur Verfügung stellen. Zu jedem Zeitpunkt kann der Anwender sich diese Informationen optional anzeigen lassen.

Ähnlich wie im Kapitel 3 können die Anforderungen an die „MR-Referenzanwendung“ formuliert werden. Das heißt, dass die Anforderungen, die jedes Softwaresystem erfüllen sollte auch für die „MR-Referenzanwendung“ gelten. Die MR spezifischen Anforderungen sollen zusätzlich aufgeführt werden.

#### Benutzbarkeit

Die Besucher der „Professur für GDV“ können in zwei Gruppen aufgeteilt werden:

- Besucher, die zum ersten Mal das Gebäude betreten.
- Besucher, die mit dem Gebäude vertraut sind.

Die „MR-Referenzanwendung“ sollte den individuellen Wünschen dieser unterschiedlichen Gruppen entsprechen. Der Benutzer sollte mühelos die Idee der „MR-Referenzanwendung“, sowie deren Nutzung verstehen können, was eine einfache Bedienung voraussetzt. Ferner sollte sie dem Benutzer die Möglichkeit geben, den Rundgang mittels der „MR-Referenzanwendung“ schnell und unkompliziert durchzuführen.

### Wartbarkeit

Andere wichtige Aspekte beim Erstellen der „MR-Referenzanwendung“ sind die Produktion und die Wartung, die im Unterkapitel 3.2.1 erläutert werden. Die „MR-Referenzanwendung“ sollte einerseits schnell und einfach produziert werden können und andererseits einfach zu pflegen und zu erweitern sein. Die folgende Punkte helfen dabei, diesen Anforderungen gerecht zu werden:

- Die „MR-Referenzanwendung“ sollte spezifische Auskünfte zur vorhandenen Räumen oder Kunstwerken im Gebäude zur Verfügung stellen, dies wird im Unterkapitel 4.2.1 näher beschrieben. Die speziellen Informationen für Räume (raumspezifische Informationen) und Mitarbeiter der Gebäude (personenspezifische Informationen) können sich relativ schnell ändern. Daher sollten die Änderungen und Erweiterungen der „MR-Referenzanwendung“ ohne Programmierung und ohne spezielle Kenntnisse eines MR-Experten möglich sein.
- Die Pflege und Erstellung der „MR-Referenzanwendung“ sollte unabhängig vom Ort durchgeführt werden können. Das bedeutet, dass die Pflege und die Verbesserungen des Systems nicht gezwungenermaßen vor Ort stattfinden müssen.
- Wie bereits erwähnt, müssen die regelmäßigen Änderungen der raumspezifischen und personenspezifischen Informationen beim Gebrauch der „MR-Referenzanwendung“ berücksichtigt werden. Daher sollte das System diese und ähnliche Informationen selbständig aus Datenbanken, die im Unternehmen etabliert sind und dort eingesetzt werden, beziehen können.

### Korrektheit

Die „MR-Referenzanwendung“ soll, ähnlich wie im Unterkapitel 3.2.1 beschrieben, die für sie spezifizierten Funktionalitäten erfüllen. Weiterhin muss bei der Entwicklung der „MR-Referenzanwendung“ beachtet werden, dass sich die reale Umgebung, in der sie eingesetzt wird, ständig verändern kann. Das hat zur Folge, dass sich die „MR-Referenzanwendung“ ihrer Umgebung anpassen muss, um einsetzbar zu bleiben. Wenn beispielsweise bestimmte Informationen geändert werden, wie der Umzug des Sekretariats, dann müssen diese Informationen in der „MR-Referenzanwendung“ aktualisiert werden. Damit wird Korrektheit nach Veränderungen in der Anwendung gesichert.

### Effizienz

Die „MR-Referenzanwendung“ sollte die Systemressourcen, die ihr zur Verfügung stehen, nicht verschwenden, siehe Unterkapitel 3.2.1. Zum Beispiel sollten Graphiken,

die für die „MR-Referenzanwendung“ zum Einsatz kommen, zwar von guter Qualität sein, jedoch komprimiert eingebettet werden.

#### Zuverlässigkeit

Die „MR-Referenzanwendung“ soll, so ähnlich wie im Unterkapitel 3.2.1 beschrieben, Stabilität und Konsistenz vorweisen können und fehlerfrei funktionieren. Jedoch sollte die Gewährleistung dieser Zuverlässigkeit nicht zu viel Arbeitszeit in Anspruch nehmen. Wie im Kapitel 3 in Abbildung 3.2 visualisiert, steigen die Kosten umso mehr, je zuverlässiger das System ist. Dieses Verhältnis wird durch eine exponentiale Steigung dargestellt.

#### Realisierung mit Videoaufnahmen

Das MR-Autorensystem, das für die Realisierung der „MR-Referenzanwendung“ verwendet wird, sollte die Produktion auf Basis von Videoaufnahmen durchführen können. Dadurch kann von dem Gang im Gebäude eine Videoaufnahme erstellt werden, mit der dann die „MR-Referenzanwendung“ konstruiert werden kann. Dies ermöglicht, dass die Produktion nicht gezwungenermaßen vor Ort durchgeführt werden muss.

#### Beleuchtung

Als nächstes soll auf die Wichtigkeit einer kontinuierlichen Beleuchtung des Einsatzbereiches eingegangen werden. Die natürliche Beleuchtung des Gebäudes durch das Tageslicht verändert sich im Laufe des Tages. Dies beeinflusst die Funktion markerbasierter Trackingsysteme, siehe Unterkapitel 2.2. Daher sollte der Einsatzbereich der „MR-Referenzanwendung“ ausreichend künstlich beleuchtet werden.

#### Positionierung der Marker

Eine andere wichtige Vorraussetzung für die Funktion der „MR-Referenzanwendung“ ist die Positionierung der Marker, siehe Unterkapitel 2.2. Diese sollten dabei möglichst unauffällig an den Wänden angebracht werden, um die Aufmerksamkeit der Besucher nicht zu sehr in Anspruch zu nehmen. Für jeden Raum können unterschiedlich ausgerichtete Markierungen angebracht werden, damit sie vom System aus verschiedenen Perspektiven im Gang erkannt werden können. Bei der Positionierung von Markern muss man die unterschiedliche Anatomie der Anwender berücksichtigen. Man muss davon ausgehen, dass die Anwender verschiedene Körpergrößen oder Behinderungen

haben können, zum Beispiel kleinwüchsige Menschen oder Rollstuhlfahrer. Dabei muss eine Position ausgesucht werden, die für unterschiedlich große Menschen geeignet ist. Der Benutzer sollte nicht seiner Bewegungsfreiheit beraubt werden. Er sollte sich frei bewegen und trotzdem alle Informationen sehen können.

#### Benutzerschnittstelle

Die Anwendung sollte eine graphische Benutzeroberfläche anbieten, die einfach zu bedienen sein sollte. Dabei sollte sie möglichst eine Menüsteuerung anbieten und konsistent in der Gestaltung und Bedienung sein. Weiterhin sollte sie eine Hilfsfunktion anbieten, die dem Benutzer während der Entwicklung zur Seite steht. Die Menge der dargestellten Informationen sollte angemessen und deren Format sollte verständlich sein. Der Anwender sollte das System intuitiv bedienen können.

#### Hardware

Das Visualisierungsgerät, das zum Einsatz kommt, sollte leicht tragbar sein. Der Bildschirm, die Kamera und die Eingabegeräte müssen ein angenehmes Arbeiten ermöglichen und sollten ergonomisch geformt sein. Das Visualisierungsgerät sollte mindestens die folgenden Anforderungen erfüllen:

- Ein Bildschirm für die Anzeige der „MR-Referenzanwendung“
- Eine Maus, ein mausähnliches Eingabegerät oder eine angemessene Schnittstelle für den Anschluss eines externen Eingabegerätes
- Eine Tastatur, tastaturähnliches Eingabegerät oder eine angemessene Schnittstelle für den Anschluss eines externen Eingabegerätes
- Eine Kamera, die für Aufnahme von Videos auf dem PC geeignet ist, oder eine passende Schnittstelle für den Anschluss einer externen Kamera
- Eine eigene Stromversorgung für den kabellosen Einsatz

Diese Anforderungen erfüllt ein Tablet PC [Mic04] oder ein PDA (Personal Digital Assistent). Ein Tablet PC ist ein von Microsoft definierter Standard für stiftbediente Personalcomputer. Tablet PCs können durch Stiftbewegungen auf dem Bildschirm bedient werden. Als Betriebssystem kommt dabei eine modifizierte Version von Windows XP zum Einsatz. Als PDA bezeichnet man tragbare Computer in Notizbuchformat zur Verwaltung und Abfrage von Termin- und Adressdaten. PDAs bestehen meist aus einer Anzeige, die den Großteil der Oberfläche des Gerätes einnimmt. Bei den meisten PDAs

werden die Eingaben mittels eines speziellen Stiftes direkt auf der druckempfindlichen Anzeige gemacht.

## 4.2 Anwendungsfunktionen

Im Folgenden werden die Funktionen der „MR-Referenzanwendung“ veranschaulicht. Dabei stellt Abbildung 4.1 diese in einer hierarchischen Struktur dar, während Abbildung 4.2 die grobe Darstellung der Funktionen mit dazugehörigen Übergängen demonstriert. Darüber hinaus soll in Abbildung 4.3 der zeitliche Ablauf dieser Funktionen angezeigt werden.

### 4.2.1 Funktionale Übersicht

Nach Definition von Balzert [Bal00] kann eine Funktion aus Eingabedaten Ausgabedaten ermitteln oder eine Veränderung des Inhalts oder der Struktur von Informationen bewirken. Im Folgenden werden die Funktionen für die „MR-Referenzanwendung“ genannt und als hierarchische Struktur in einem Diagramm dargestellt. Diese hierarchische Art von Darstellung wird nach Balzert [Bal00] als Funktionsbaum bezeichnet. Die Funktionen werden in diesen Funktionsbaum als beschriftete Rechtecke und ihre Beziehungen durch unbeschriftete Verbindungslinien dargestellt. Das folgende Diagramm besteht aus sieben verschiedenen Hauptfunktionen, die im nächsten Unterapitel genauer erläutert werden. Die erste Ebene in diesem Diagramm stellt die Funktion „Start“ dar. Beim Starten des Systems wird der Benutzer durch einen eingeblendeten Begrüßungstext und einer Sprachausgabe willkommen geheißen.

In der zweiten Ebene des Diagramms werden die Funktionen „Hauptmenü“, „Hilfe“ und „Beenden“ dargestellt. Das „Hauptmenü“ verursacht den Aufruf von drei Untermenüs, „Navigation“, „Objektspezifische Informationen“ („Objekt Info“) und „Lageplan“, die in der nächsten Ebene dargestellt werden. Die „Navigation“ soll den Benutzern dabei helfen, ein bestimmtes Ziel im Gebäude zu erreichen. Diese Funktion kann von jedem beliebigen Ort im Gebäude aufgerufen werden. Die Besucher sollen ihr Ziel im Gebäude aus den drei Kategorien „Personen“, „Räumen“ und „Kunstwerken“ bestimmen können. In der Kategorie „Personen“ kann der Anwender in einer Liste aller Mitarbeiter nach einer gewünschten Person suchen. Ferner besteht die Möglichkeit, sich eine Liste der vorhandenen Räume oder Kunstwerke im Gebäude anzeigen zu lassen. Der Prozess der Auswahl des Anwenders wird in der Abbildung 4.1 als „Entscheidung“ bezeichnet. Danach wird dem Benutzer der Weg mit Hilfe von richtungsweisenden virtuellen Schildern angezeigt. Wenn der Anwender sich vor seinem Zielort befindet, wird er visuell durch ein virtuelles Schild informiert.

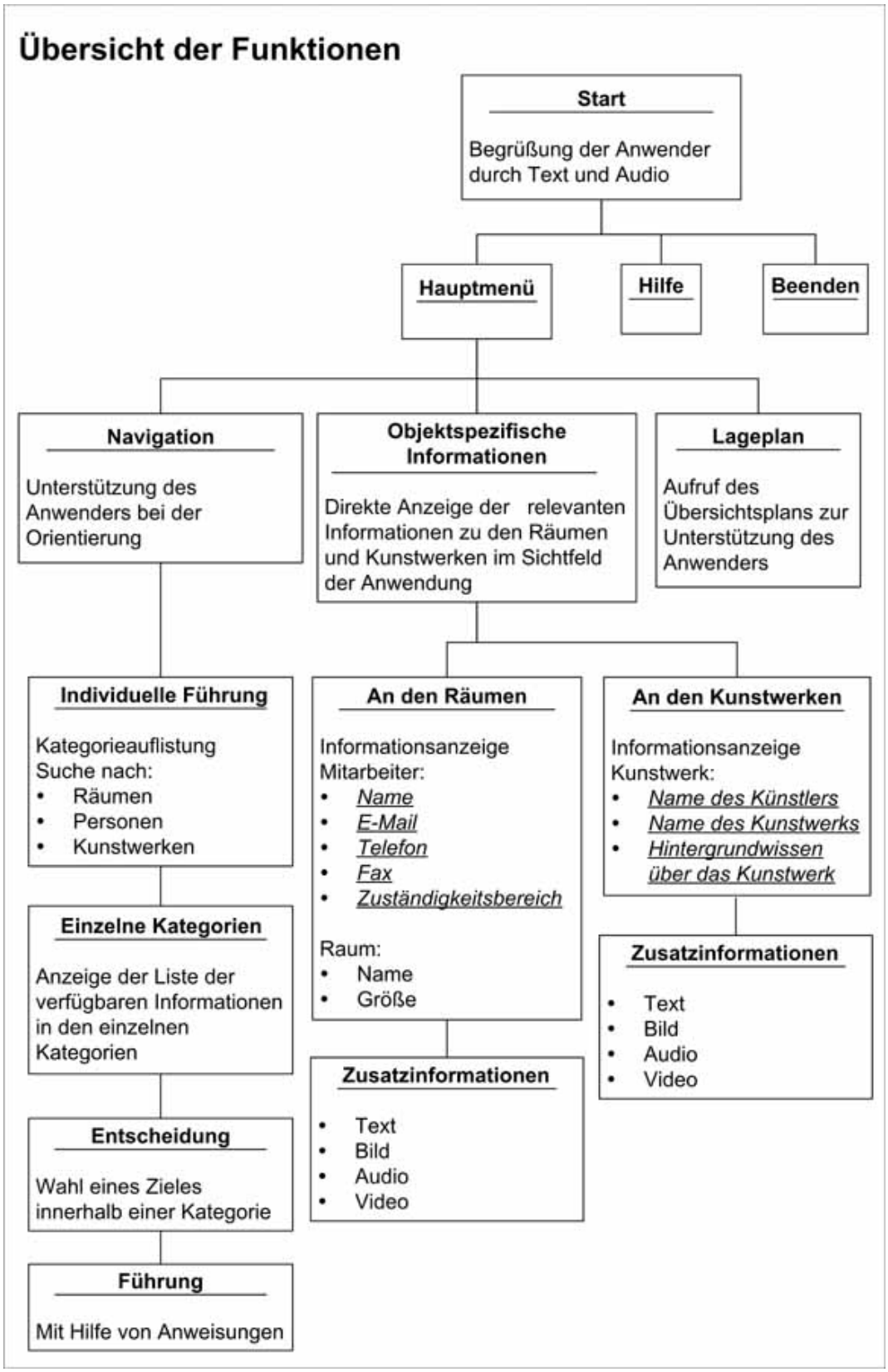


Abb. 4.1: Funktionale Übersicht der MR-Referenzanwendung

Als nächstes wird die Funktion „Objekt Info“ erläutert. Zunächst wird der Begriff „Objekt“ definiert. Mit „Objekten“ sind Räume und Kunstwerke im Gebäude gemeint, zu denen jeweils spezifische Informationen für den Benutzer verfügbar sind. Wenn der Benutzer sich für die Funktion „Objekt Info“ entscheidet, erhält er spezifische Auskünfte zu vorhandenen Räumen oder Kunstwerken im Gebäude. Diese Informationen können in Form von Texte, Audios, Videos und/oder Graphiken zur Verfügung gestellt werden. Dabei ist es wichtig, dass der Benutzer sich in der Nähe dieser Objekte befindet, damit diese vom System erkannt werden. Steht der Benutzer beispielsweise am Eingang, kann er keine Informationen über das Sekretariat erhalten. Diese Informationen erhält der Anwender nur, wenn er sich in der Nähe vom Sekretariat befindet, so dass die „MR-Referenzanwendung“ es erkennen kann.

Beim Aufruf der Funktion „Lageplan“ erhält der Benutzer einen Übersichtsplan vom Gebäude. Auf diesem Plan wird der Standort des Benutzers markiert. Diese Funktion soll dem Anwender dabei helfen, sich im Gebäude zu orientieren.

Als nächstes wird die „Hilfe-Funktion“ beschrieben. Diese Funktion unterstützt den Anwender bei der Verwendung der „MR-Referenzanwendung“. Der Anwender kann jeder Zeit diese Funktion aufrufen, um spezielle Informationen über alle Funktionen der zweiten Ebene des Diagramms zu erhalten. Diese Funktion soll die Benutzung der „MR-Referenzanwendung“ vereinfachen und kann von jedem Ort des Gebäudes aufgerufen werden. Ferner bietet diese Funktion eine schnelle Einführung in die „MR-Referenzanwendung“ an. Bei der Funktion „Beenden“ kann sie durch den Benutzer dementsprechend beendet werden. Diese Funktion kann auch wie die „Hilfe“ jederzeit und an jedem Ort ausgeführt werden.

### 4.2.2 Aktivitätsdiagramm

Um die Komplexität der Darstellung der Aktivitäten und Funktionen für die „MR-Referenzanwendung“ zu reduzieren, werden UML CASE Werkzeuge eingesetzt, die im Folgenden genauer erläutert werden. UML (Unified Modeling Language) ist eine von der „Object Management Group“ entwickelte und standardisierte Entwurfssprache in Form einer graphischen Notation, um Strukturen und Abläufe in objektorientierten Programmsystemen darzustellen [OMG04].

Damit das Diagramm und der Übergang zwischen den Funktionen für den Leser so einfach und verständlich wie möglich dargestellt werden, werden zunächst die Hauptfunktionen und die möglichen Übergänge zwischen ihnen in Abbildung 4.2 visualisiert. Als nächster Schritt werden in einem zweiten Diagramm namens Aktivitätsdiagramm in Abbildung 4.3 alle Funktionen mit deren möglichen Übergängen dargestellt. Dabei werden Systemzustände und Ereignisse, die Übergänge von einem Zustand zum anderen auslösen, definiert und erläutert.

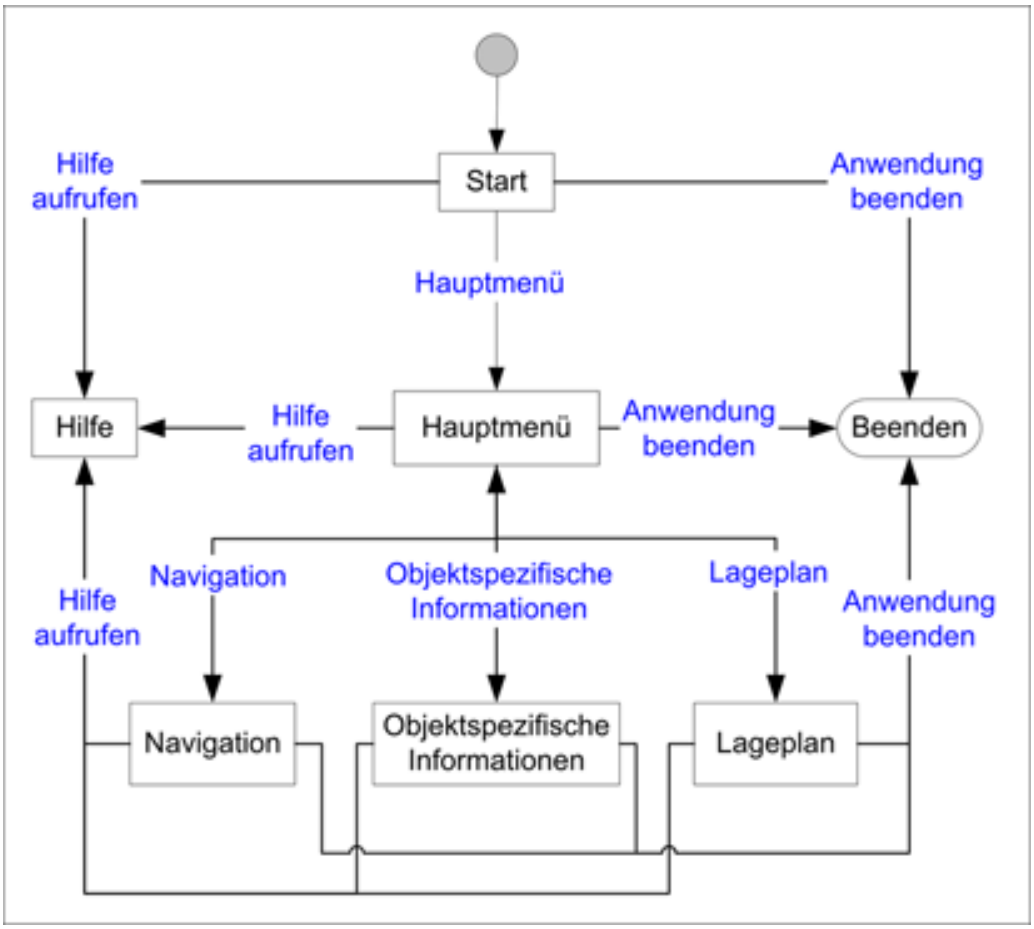


Abb. 4.2: Darstellung der Funktionen und Übergänge

Diese Diagramme werden mit Hilfe der Applikation Microsoft Visio erstellt. Es dient einem besseren Verständnis der Funktionsabläufe in der „MR-Referenzanwendung“. Durch diese Art von Darstellung soll das Verhalten der „MR-Referenzanwendung“ als Reaktion auf externe Ereignisse dargestellt werden. Laut Balzert [Bal00] besteht ein Aktivitätsdiagramm aus Zuständen und Zustandsübergängen. Jeder Zustand modelliert einen Schritt innerhalb der Gesamtverarbeitung, während ein Zustandsübergang zwei Zustände verbindet. Er kann nicht unterbrochen werden und wird stets durch ein Ereignis ausgelöst. In der UML Notation wird der Übergang von einem Zustand in den anderen anhand von beschrifteten Pfeilen dargestellt. Für die „MR-Referenzanwendung“ soll die Handlungsabfolge zunächst wie folgt aussehen:

Im Zustand „Start“ wird der Anwender begrüßt. Zusätzlich können die „Hauptfunktionen“ aktiviert werden. Als „Hauptfunktionen“ werden die Funktionen „Hilfe“,

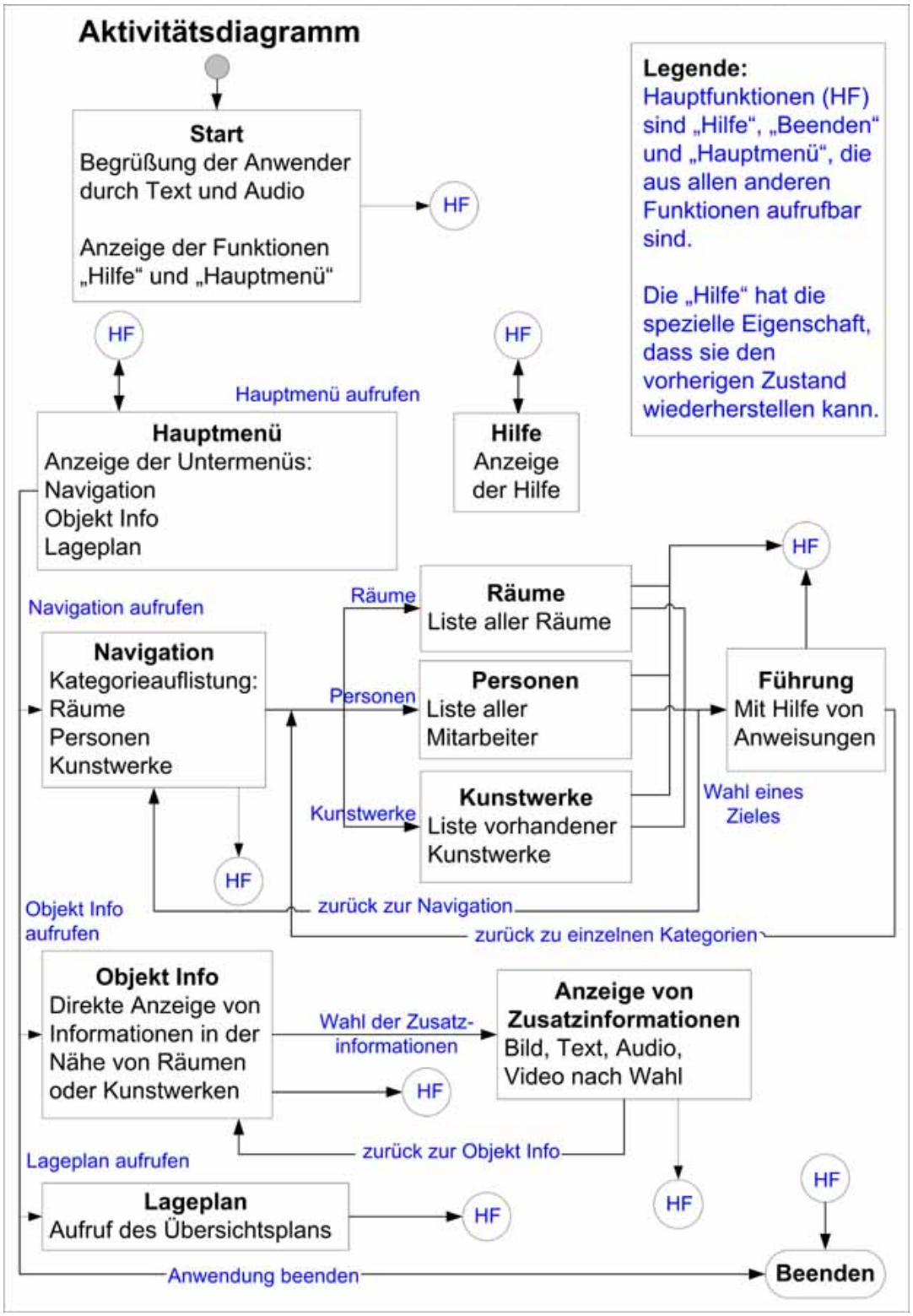


Abb. 4.3: Aktivitätsdiagramm der MR-Referenzanwendung

„Hauptmenü“ und „Beenden“ definiert. Diese drei Funktionen werden deshalb so bezeichnet, weil der Benutzer aus allen anderen Zuständen jeder Zeit zu diesen wechseln kann. Durch Wählen von „Hauptmenü“ oder „Hilfe“ wird der Zustand „Start“ verlassen und der Begrüßungstext verschwindet. Im Zustand „Hauptmenü“ werden die Untermenüs „Navigation“, „Objekt Info“ und „Lageplan“ zur Auswahl gestellt. Beim Wählen von einem dieser Untermenüs wird zum jeweiligen Zustand gewechselt. Wenn der Zustand „Navigation“ aktiviert wird, werden die Kategorien „Räume“, „Personen“ und „Kunstwerke“ aufgelistet. Durch Wahl einer Kategorie wird der Zustand „Navigation“ verlassen und in den Zustand der gewählten Kategorie gewechselt. Es besteht auch die Möglichkeit, die „Hauptfunktionen“ zu wählen. Diese Möglichkeit wird im Diagramm durch „On-page Referenzen“ mit der Bezeichnung HF visualisiert. In den Zuständen „Räume“, „Personen“ und „Kunstwerke“ wird jeweils eine Liste der vorhandenen Informationen zur Verfügung gestellt. Durch Wählen des gewünschten Zieles wird in den Zustand „Führung“ gewechselt, der den Benutzer durch Anweisungen zu seinem Ziel führt. Auch hier besteht die Möglichkeit in den vorherigen Zustand „Navigation“ oder in die Zustände der „Hauptfunktionen“ zu wechseln. Nachdem der Benutzer sein Ziel erreicht hat.

Im Zustand „Objekt Info“ werden Informationen zu den Objekten im Sichtfeld des Systems direkt angezeigt. Ferner werden zusätzliche Informationen zur Verfügung gestellt. Beim Wählen einer dieser Zusatzinformationen wird dieser Zustand verlassen und der ausgewählte Zustand aktiviert. Auch hier besteht die Möglichkeit in den vorherigen Zustand oder zu den „Hauptfunktionen“ zurückzukehren.

Als letztes wird der Zustand „Lageplan“ erläutert, in dem ein Übersichtsplan angezeigt wird. Auch hier kann zu den Zuständen der „Hauptfunktionen“ gewechselt werden.

## 4.3 Anwendungsfälle

Die erste Stufe jedes Softwareentwurfsprozesses besteht darin, sich die Beziehungen zwischen der entworfenen Software und der Umwelt klar zu machen [Som01]. Für den Entwurf der „MR-Referenzanwendung“ ist es zunächst wichtig, die Beziehungen zwischen dieser Anwendung und ihrer Umwelt zu verstehen, Dies kann die Struktur der Anwendung stark beeinträchtigen. Um das Verständnis dieser Beziehungen zu erleichtern, wird in diesem Abschnitt zuerst ein Anwendungsfallmodell vorgestellt, welches die Interaktionen zwischen den systemexternen Akteuren und der „MR-Referenzanwendung“ modelliert [JCJÖ92]. Danach werden die Funktionalitäten genauer beschrieben. Nach der Ansicht von Balzert [Bal00] besteht ein Anwendungsfall (engl. Use case) aus mehreren zusammenhängenden Aufgaben. Sie werden von einem Akteur durchgeführt, um ein Ziel zu erreichen oder ein gewünschtes Ergebnis zu erhalten. Dabei ist ein Akteur eine Rolle, die ein Benutzer des Systems spielt.

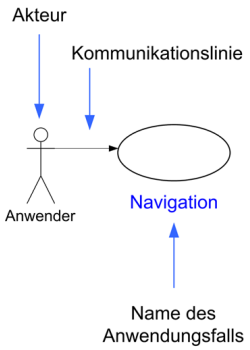


Abb. 4.4: Allgemeines Beispiel für ein Anwendungsfalldiagramm (vgl. [GBZK01])

Die Anwendungsfälle beschreiben das Verhalten des Systems im Einsatz. Dabei zeigt ein Anwendungsfall nur das an, was möglich sein muss und nicht wie es funktioniert. Jeder Anwendungsfall beschreibt die erforderliche Folge von Interaktionsschritten bei der Durchführung einer Systemfunktion. Dabei steht die Kommunikation der Akteure mit dem System im Mittelpunkt, und nicht etwa die interne Struktur des Systems. Das Zusammenspiel mehrerer Anwendungsfälle untereinander und mit den Akteuren wird in einem Anwendungsfalldiagramm dargestellt [GBZK01]. In UML werden Akteure durch Strichmännchen und Interaktionen des Benutzers mit dem System als gekennzeichnete Ellipse dargestellt, siehe Abbildung 4.4.

Abbildung 4.5 veranschaulicht ein Anwendungsfalldiagramm für die „MR-Referenzanwendung“. Nachfolgend werden alle Anwendungsfälle im Einzelnen aufgeführt und beschrieben. Zur Veranschaulichung der Anwendungsfälle wird bei der Erklärung von jedem Fall eine Abbildung präsentiert. Dabei geht diese Abbildung auf die Funktionalität jedes Anwendungsfalls ein und dient als Leitfaden für die Realisierung der „MR-Referenzanwendung“.

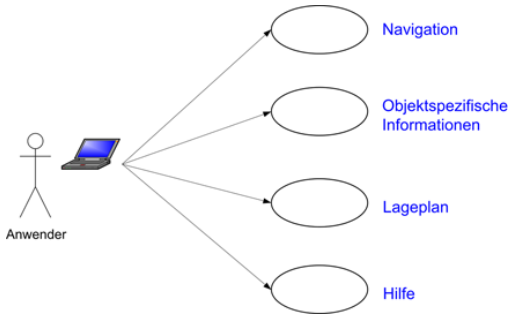


Abb. 4.5: Anwendungsfalldiagramm für die MR-Referenzanwendung

### 4.3.1 Navigation

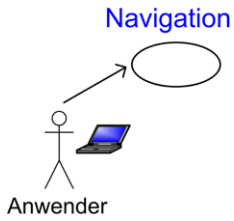


Abb. 4.6: Anwendungsfall Navigation


Abb. 4.7: Start der Navigation

Dieser Anwendungsfall beginnt, wenn der Anwender im Menü die Funktion „Navigation“ anfordert, durch die das System dem Benutzer eine individuelle Führung anbieten kann. Der Anwender kann anhand einer Navigationshilfe ein von ihm bestimmtes Ziel im Gebäude erreichen. Dabei muss er sein Ziel im Gebäude festlegen und bekommt dementsprechend einen Wegweiser, mit dessen Hilfe er dies erreichen kann. Der Benutzer muss zunächst den Navigationsbutton wählen; danach werden die vorhandenen Kategorien und deren Untergruppierungen ihm zur Auswahl gestellt. Das heißt, nach der Wahl der „Navigation“ werden die drei Kategorien „Räume“,

„Personen“ und „Kunstwerke“ zur Auswahl gestellt. Der Anwender soll nun die Entscheidung treffen, ob er nach vorhandenen Räumen oder Kunstwerken im Gebäude suchen will. Ferner kann er sich über alle Mitarbeiter des Gebäudes informieren. Wenn der Anwender sich für eine Kategorie entscheidet, sollen alle Informationsgruppen zu dieser Kategorie angezeigt werden, die wiederum durch den Benutzer gewählt werden können. Des Weiteren soll die Möglichkeit bestehen, die gewählte Suchfunktion rückgängig zu machen und somit zum alten Zustand zurückzukehren.

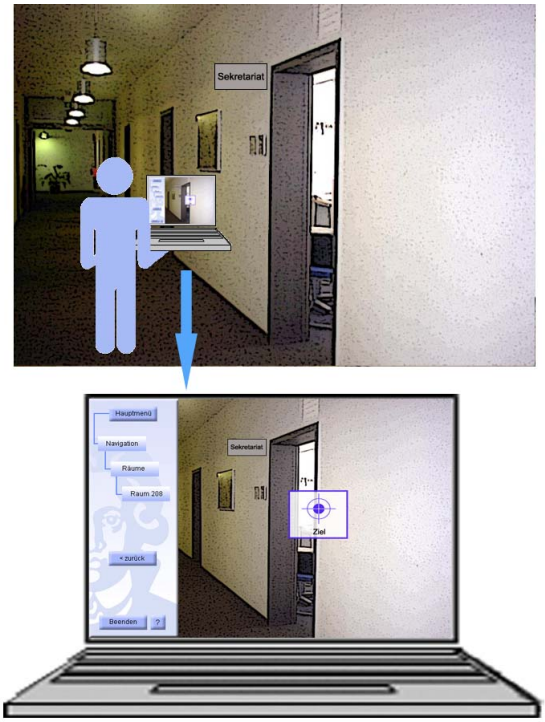


Abb. 4.8: Navigation am Ziel

Der Weg, den der Anwender begehen muss, wird ihm in Form von Anweisungen auf beschrifteten Schildern mitgeteilt. Abbildung 4.7 und Abbildung 4.8 stellen diese Funktion graphisch dar. Im oberen Teil der Abbildung 4.7 wird der Anwender, der sich am Eingang des Gebäudes befindet, mit einem tragbaren Anzeigegerät angezeigt. Dabei benutzt er die „MR-Referenzanwendung“, um von seinem Standort zu dem gewünschten Zielort, in diesem Fall das Sekretariat des Gebäudes, zu gelangen. Im unteren Teil der Abbildung 4.7 wird die Anzeige des Eingabegerätes des Benutzers deutlicher dargestellt. Der Anwender sieht zusätzlich zu dem angezeigten Realitätsbild einerseits Wegweiseranweisungen zu seinem Zielobjekt, andererseits diverse Buttons für die Wahl der Interaktion des Benutzers mit dem System. Das Ergebnis der Nachfrage wird anhand eines Schildes dargestellt, das den Benutzer zu seinem Ziel führt.

In Abbildung 4.8 wird der Fall visualisiert, wie das System die Funktion „Navigation“ erfolgreich beendet hat und am Ende seines Durchlaufs den Benutzer darüber informiert. In diesem Beispielsfall steht nun der Anwender vor dem Sekretariat und wird auf seiner Anzeige darauf hingewiesen, dass er sein Ziel erreicht hat, wie auf dem unteren Teil der Abbildung 4.8 zu sehen ist.

### 4.3.2 Objektspezifische Informationen


Abb. 4.9: Anwendungsfall Objekt Info


Abb. 4.10: Objekt Info zum Sekretariat

Dieser Anwendungsfall stellt die Option dar, wie der Benutzer im Menü „Objekt Info“ anfordert. Wie oben im Unterkapitel 4.2.1 geschildert, werden Objekte in diesem Zusammenhang als Räume und Kunstwerke bezeichnet, und es werden Informationen darüber je nach Aufforderung des Anwenders mitgeteilt. Diese Zusatzinformationen können als Text, Graphik, Audio oder Video zur Verfügung gestellt werden. Dabei muss sich das System in der Nähe des Objektes befinden. Sobald der Anwender sich in der Nähe eines Objektes befindet, sollen ihm optional alle Informationen über dieses Objekt mitgeteilt werden. Abbildung 4.10 und Abbildung 4.11 zeigen diesen Fall für den Raum Sekretariat des Gebäudes graphisch an.

Im oberen Teil der Abbildung 4.10 wird ein Benutzer dargestellt, der sich mit seinem Anzeigegerät direkt vor dem Sekretariat im Gebäude befindet. Das System erkennt die Position des Anwenders und kann Informationen über das Sekretariat anzeigen. Der untere Teil der Abbildung zeigt, wie der Anwender vorhandene Informationen zusätzlich zu dem angezeigten Realitätsbild auf seiner Anzeige sieht. Dabei können alle verfügbaren Informationen über das Sekretariat, wie zum Beispiel raumspezifische Informationen, sowie Informationen über die Mitarbeiter des Sekretariats in den bereits erwähnten Formen optional angezeigt werden.

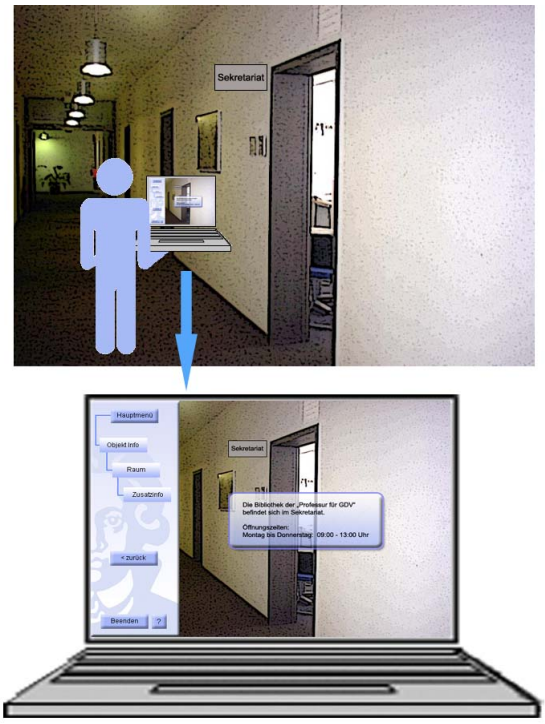


Abb. 4.11: Zusatzinfo beim Objekt Info

In Abbildung 4.11 wird die Funktion „Zusatzinfo“ dargestellt. Darin werden zusätzliche Informationen über das Sekretariat auf der Anzeige des Anwenders dargestellt, was im unteren Teil der Abbildung 4.11 zu sehen ist. Wenn der Benutzer den Funktionsbutton „Foto“ drückt, wird das Bild der ausgesuchten Person im Sekretariat dargestellt. Entscheidet sich der Anwender für das Abspielen eines Videos, kann zum Beispiel eine Aufnahme des Sekretariats von innen gezeigt werden. „Audio-Funktion“ kann eine Audio Datei abspielen lassen. Dies kann zum Beispiel ein kurzes Musikstück sein, das für die Entspannung des Benutzers sorgt oder als Information dient.

Für die Kunstwerke, die sich im Gebäude befinden, können die objektspezifische Informationen ähnlich wie für die Räume aussehen. Dabei werden Informationen zu den Kunstwerken angezeigt, die für die Benutzer interessant sind. Diese enthalten Auskünfte über den Künstler, sein Kunstwerk, Hintergrundwissen darüber oder Informationen in Form von kurze Werbefilme und Musikstücke.

### 4.3.3 Lageplan

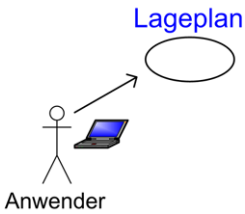


Abb. 4.12: Anwendungsfall Lageplan

Der Anwendungsfall „Lageplan“ beginnt, wenn der Benutzer im Menü den „Lageplan“ anfordert. Der Anwender soll die Möglichkeit haben, am Eingang, sowie vor jedem Raum des Gebäudes diesen Plan aufzurufen, um seine Position mit Hilfe des Lageplanes zu erkennen. Diese Funktion kann, wie in Abbildung 4.13 zu sehen ist, den Benutzern helfen, sich im Gebäude zu orientieren.

Im oberen Teil des Bildes sieht man einen Benutzer mit einem tragbaren Computer, der sich in der Nähe des Sekretariats im Gebäude befindet. Im unteren Teil des Bildes wird die Anzeige des Eingabegerätes deutlich dargestellt. Nachdem der Benutzer sich für die Funktion „Lageplan“ entschieden hat, erscheint ein Umriß des Gebäudes, in dem das System die Position des Anwenders erkennt und sie markiert. In diesem Beispiel wird das Sekretariat auf dem Lageplan durch blaue Markierung hervorgehoben.

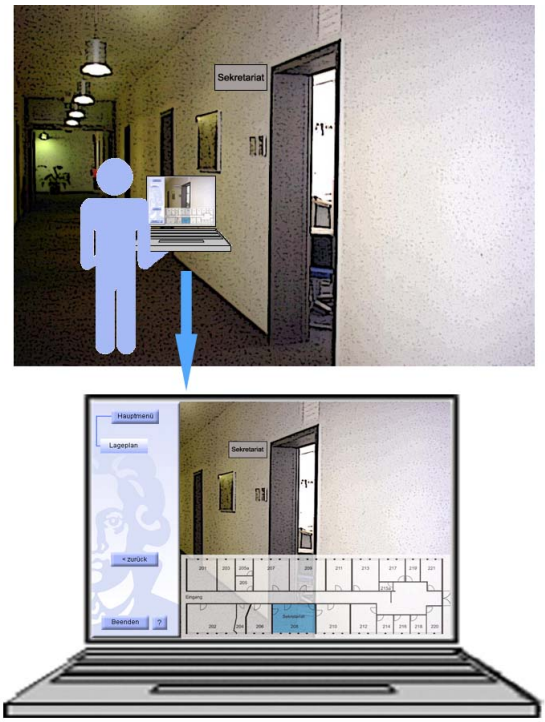


Abb. 4.13: Lageplan am Sekretariat

### 4.3.4 Hilfe

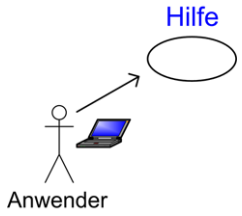


Abb. 4.14: Anwendungsfall Hilfe

Dieser Anwendungsfall tritt ein, wenn der Benutzer im Menü die Funktion „Hilfe“ anfordert. Diese Funktion unterstützt den Anwender, damit er die Idee der „MR-Referenzanwendung“, sowie die einzelnen Funktionen schnell und einfach erlernen kann. Daher sollte zu jeder Funktion ausreichend Hilfe zur Verfügung gestellt werden. Ferner sollte der Benutzer in der Lage sein, diese Hilfsfunktion ortsunabhängig und zu jeder Zeit während der Nutzung der „MR-Referenzanwendung“ aufzurufen. Die „Hilfe“ sollte dem Benutzer eine simple Beschreibung einzelner Funktionen im Detail ermöglichen. Der Inhalt der Hilfsfunktion soll kontextbezogen angezeigt werden. In Abbildung 4.15

wird dieser Fall veranschaulicht. Im oberen Teil der Graphik wird ein Benutzer angezeigt, der sich im Flur des Gebäudes befindet. Dabei hält er ein Eingabegerät in der Hand, dessen Anzeige im unteren Bereich der Abbildung verdeutlich wird. Nachdem der Benutzer den Befehl „Hilfe“ eingegeben hat, erscheint ein Hilfstext auf dem Bildschirm.

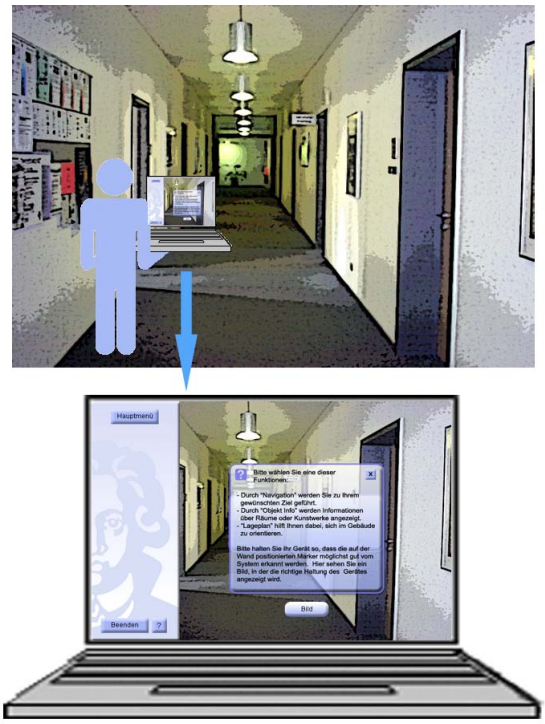


Abb. 4.15: Aufruf der Hilfefunktion

## 4.4 Design der grafischen Benutzeroberfläche

In diesem Abschnitt wird eine graphische Benutzeroberfläche für die „MR-Referenzanwendung“ entworfen. Bisher wurden im Unterkapitel 4.2 nur Funktionen der „MR-Referenzanwendung“ ohne Bezüge zur Benutzeroberfläche beschrieben. Die Spezifikation der GUI findet hier statt. Dabei wird schrittweise anhand von Entwurfsbildern erläutert und veranschaulicht, wie die Oberfläche der „MR-Referenzanwendung“ bei allen Funktionen aus Unterkapitel 4.2 aussehen soll.

Bevor im Folgenden die Oberfläche im Einzelnen näher beschrieben wird, werden einige grundlegende Elemente als allgemein gültig für alle Funktionen definiert. Diese sind zum Beispiel die Hintergrundfarbe, die Schriftfarbe und Schriftgröße der Buttons, die aus den folgenden Abbildungen zu entnehmen sind. Für die Beschriftung der Buttons soll die Farbe schwarz gewählt werden, damit die Beschriftungen durch kontrastreiche

Hervorhebung vom Anwender am einfachsten erkannt werden. Auf der linken Seite der Oberfläche wird das Logo der Johann Wolfgang Goethe-Universität in blau-weißer Farbe angezeigt, wie in folgenden Abbildungen zu sehen ist. Auf dem Bereich, wo das Logo platziert wird, werden die Buttons für die Hauptfunktionen „Hauptmenü“, „Hilfe“ und „Beenden“ angebracht. Die Buttons sollten möglichst klein erstellt werden, damit sie nicht viel Platz in der Anzeige in Anspruch nehmen. Aus ästhetischen Gründen sollten sie alle möglichst die gleiche Breite und Höhe haben.

Die Größe der einzelnen Buttons, die für die „MR-Referenzanwendung“ gewählt werden soll, entspricht der Größe in den folgenden Abbildungen. Die Buttons für die „Hauptfunktionen“ sollten dabei in quadratischer Form und die Buttons für die Untermenüs „Navigation“, „Objekt Info“ und „Lageplan“ in ovaler Form und in einer anderen Farbe dargestellt werden. Die unterschiedlichen Farben sollen eine Änderung des Systemzustandes verdeutlichen [Som01]. Die Buttons, die immer sichtbar sind und die „zurück“-Buttons sollten blau sein. Die restlichen Buttons sollen zweifarbig in blau und weiß angezeigt werden, wobei diese zwei Farben ineinander verlaufen sollen.

Die graphische Benutzerschnittstelle der „MR-Referenzanwendung“ verfügt über ein hierarchisches Menüsystem, welches dem Anwender die Möglichkeit bietet, zwischen verschiedenen Funktionsmöglichkeiten auszusuchen [Som01]. Dieses Menüsystem wird als eine hierarchische Baumstruktur dargestellt. Die hierarchische Baumstruktur zwischen Obermenüs und Untermenüs wird durch blaue Linien und durch die Reihenfolge der Buttons verdeutlicht.

### 4.4.1 Start

Zunächst wird erklärt, wie das System beim Starten der „MR-Referenzanwendung“ aussieht. Wenn das System gestartet wird, wird der Benutzer visuell und auditiv durch den Text „Willkommen in der Professur für Graphische Datenverarbeitung der Johann Wolfgang Goethe-Universität Frankfurt am Main“ begrüßt. Abbildung 4.16 stellt ein Entwurf der Oberfläche des Systems beim Starten der „MR-Referenzanwendung“ dar. Diese Abbildung zeigt, wie der Benutzer beim Starten des Systems zusätzlich zu der dargestellten Realität den Begrüßungstext und je ein Button für „Hauptmenü“, „Hilfe“ und „Beenden“ sehen kann, wie in Abbildung 4.16, Markierung 1 und 2 dargestellt. Der Begrüßungstext sollte wie in Abbildung 4.16, Markierung 3 am besten zentriert, mit schwarzen Buchstaben auf weißem Hintergrund und in einem blauen Rahmen dargestellt werden.

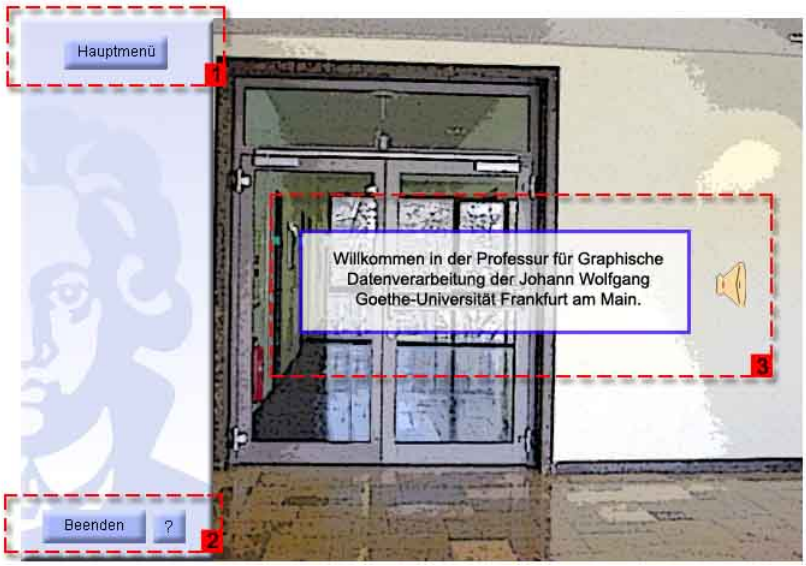


Abb. 4.16: Entwurf der Startoberfläche

### 4.4.2 Navigation

Durch diese Funktion wird der Benutzer zu seinem gewünschten Ziel innerhalb des Gebäudes geführt. Der Ablauf dieser Funktion ist im Großen und Ganzen wie folgt: Sobald der Navigationsknopf ausgewählt wird, erscheint eine Liste der möglichen Navigationskategorien. Nachdem der Benutzer sich für eine Kategorie entschieden hat, erscheinen die Auswahlmöglichkeiten in dieser Kategorie. Nach Auswahl des bestimmten Ziels, beginnt die Führung des Benutzers mit Hilfe von Anweisungen, die zum Ziel führen. Die folgenden Abbildungen veranschaulichen den Ablauf der Funktion „Navigation“ in einzelnen Schritten. In Abbildung 4.17 wird der erste Schritt dieses Vorgangs visualisiert. Nach Aktivierung des Hauptmenüs sind die Untermenüs in der Reihenfolge „Navigation“, „Objekt Info“ und „Lageplan“ direkt unter dem „Hauptmenü“ auf dem linksseitigen Logo zu sehen, siehe Abbildung 4.17, Markierung 1 und 2.

Nun wird der Navigationsbutton ausgewählt. Nach der Wahl der „Navigation“ wird dieser Button deaktiviert. Ein aktiver Zustand wird durch einen schattierten Button verdeutlicht. Nun sind die Untermenüs „Objekt Info“ und „Lageplan“ nicht mehr sichtbar. Es erscheinen, wie in Abbildung 4.18, Markierung 3 zu sehen ist, die bei der Funktion „Navigation“ die möglichen Optionen „Räume“, „Personen“, „Kunstwerke“ und „zurück“ in dieser festgelegten Reihenfolge. Durch „zurück“ gelangt der Benutzer in „Hauptmenü“ zurück.

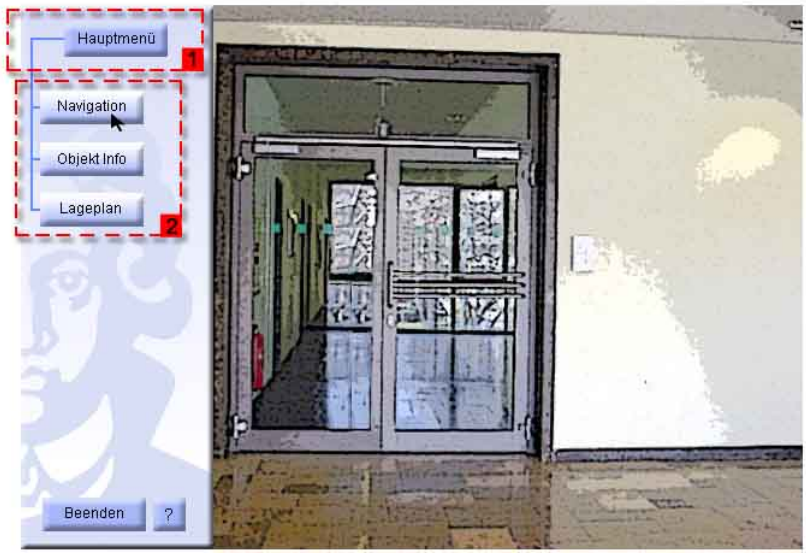


Abb. 4.17: Entwurf der Oberfläche im Hauptmenü

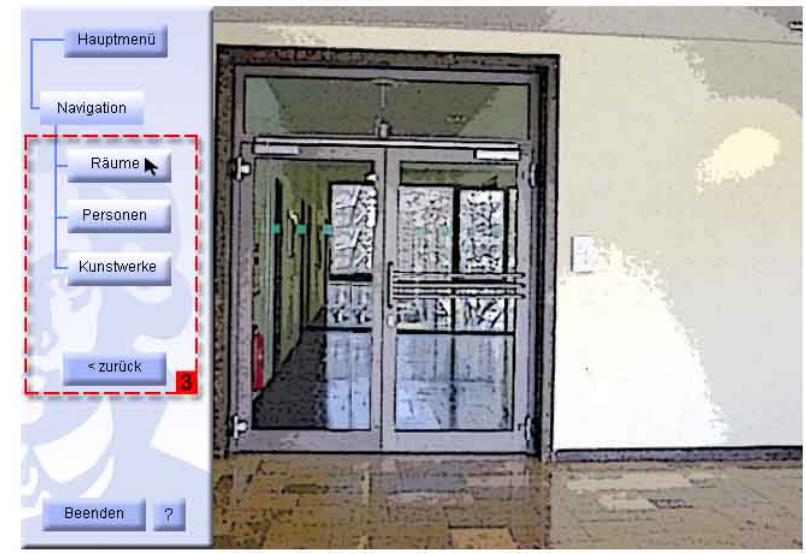


Abb. 4.18: Entwurf der Oberfläche nach der Wahl von Navigation

Im unten dargestellten Beispielfall wird als nächstes die Option „Räume“ ausgewählt. Nun wird der Button „Räume“ deaktiviert und es werden die Untermenüs „Eingang“ und „Raum 202“ bis „Raum 221“ in einer aufsteigenden Reihenfolge von links nach rechts sichtbar, siehe Abbildung 4.19, Markierung 4. Auch hier soll nach dem bereits beschriebenen Prinzip der hierarchischen Baumstruktur vorgegangen werden. Eine Rückführung in den vorherigen Zustand wird hier durch ein „zurück“-Button ermöglicht. Für den Fall, dass der Benutzer sich für die Optionen „Personen“ oder „Kunstwerke“ entscheiden sollte, sieht die Struktur der Oberfläche ähnlich wie bei „Räume“ aus. Für die Option „Personen“ sollen die Untermenüs mit den Namen der Mitarbeiter des Gebäudes lexikographisch aufwärts sortiert von links nach rechts angezeigt werden. Falls der Anwender sich für eine Person entscheidet, wird er zu seinem Arbeitsraum geführt. Ferner hat der Benutzer die Möglichkeit, nach den Kunstwerken des Gebäudes zu suchen. Auch hier werden die Untermenüs mit dem Namen der verfügbaren Kunstwerke im Gebäude nach oben genannter Reihenfolge sortiert und dargestellt. Nachdem der Anwender sich für ein Kunstwerk entschieden hat, wird er zu seinem Zielobjekt geführt.

Im nächsten Schritt dieses Beispiels drückt der Benutzer den Button „Raum 208“. Nun verschwinden alle Buttons, die zur Funktion „Räume“ gehören außer „Raum 208“. Dieser Button erscheint unter dem Button „Räume“. Insgesamt wird die hierarchische Reihenfolge der bisher gewählten Buttons auf dem linksseitigen Logo sichtbar gemacht, um den Benutzer über den aktuellen Zustand zu informieren. Auch hier wird eine Rückführung in den vorherigen Zustand durch einen „zurück“-Button ermöglicht. Zusätzlich wird das Ziel der ausgesuchten Funktion in Form eines Hinweises: „Den Gang Richtung Ende folgen!“ beziehungsweise „Den Gang Richtung Haupteingang folgen!“ abhängig von der Position des Anwenders im Gang, zentriert auf dem Bildschirm angezeigt. Diese Meldung soll auf einem weißen Quadrat blau umrandet mit einem zusätzlichen links platzierten „i“ als Abkürzung für Information dargestellt werden.

Als nächstes wird der Fall visualisiert, wie der Benutzer sich in der Nähe seines Ziels befindet. Das System erkennt die Position des Benutzers und informiert ihn durch eine Meldung. Diese Meldung soll zentriert in einem weißen Quadrat blau umrandet und mit einem in Abbildung 4.21 dargestellten blauen Symbol sichtbar gemacht werden. Diese Meldung soll mit der Bezeichnung „Ziel“ in schwarz zusätzlich markiert sein, siehe Abbildung 4.21. Für das oben genannte Beispiel erscheint diese Meldung, sobald sich der Benutzer dem Sekretariat nähert.

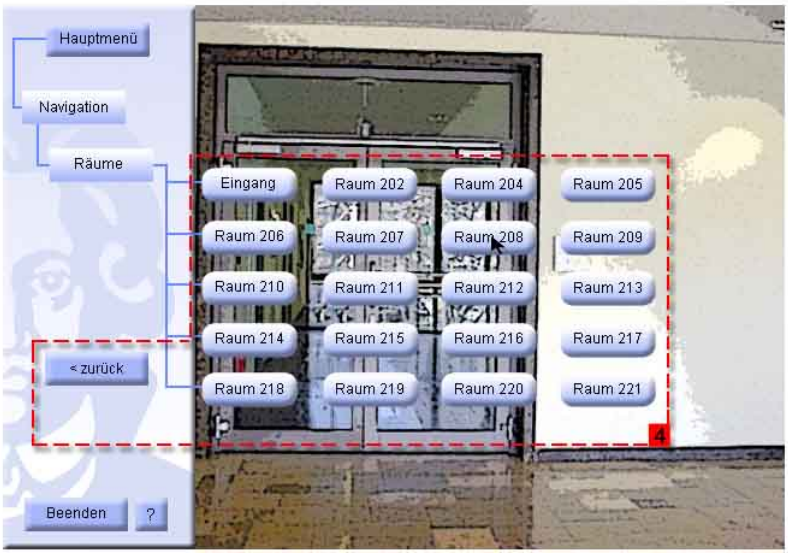


Abb. 4.19: Anzeige der Auswahl aller Räume


Abb. 4.20: Entwurf der Oberfläche während der Benutzerführung

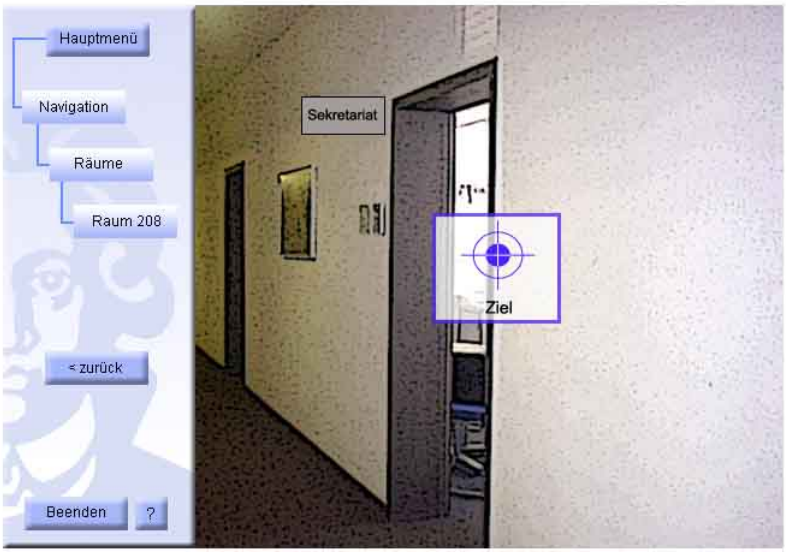


Abb. 4.21: Entwurf der Oberfläche am Ziel

### 4.4.3 Objektspezifische Informationen

Um das Verständnis dieser Funktion für den Leser so einfach wie möglich zu machen, wird im Folgenden anhand von Bildern erläutert, wie die Oberfläche des Systems bei der Wahl von „Objekt Info“ aussieht. Dabei wird Schritt für Schritt anhand von Beispielen erklärt, wie die Funktion „Objekt Info“ dem Benutzer Informationen über Räume oder Kunstwerke im Gebäude zur Verfügung stellt. Zunächst muss der Anwender diese Funktion vom „Hauptmenü“ aus aufrufen, wie in Abbildung 4.22, Markierung 2 zu sehen ist.

Wie bereits aus dem Unterkapitel 4.2.1 zu entnehmen ist, werden Objekte in diesem Zusammenhang als Räume und Kunstwerke bezeichnet. Abhängig davon, ob der Anwender sich in der Nähe eines Raumes oder in der Nähe eines Kunstwerkes befindet, werden hier die verfügbaren Informationen zu dem jeweiligen Objekten dargestellt. In der Nähe von Räumen werden Informationen über diese und über deren Mitarbeiter zur Verfügung gestellt. Die raumspezifischen Informationen beinhalten Namen und Größe des Raumes, während bei den personenspezifischen Informationen Name, E-Mail Adresse, Telefon- und Faxnummer, sowie der Zuständigkeitsbereich von allen Mitarbeitern in dieser Reihenfolge dargestellt werden. Dabei sollten diese Informationen mit schwarzen Buchstaben auf weißem Hintergrund und in einem blauen Rahmen dargestellt werden. Um die Übersicht der dargestellten Informationen für die Benutzer zu verbessern, werden raumspezifische und personenspezifische Informationen getrennt angezeigt.

Im hier vorgestellten Beispielfall sieht man einen Anwender, der sich in der Nähe des Sekretariats befindet. Diese Situation ist in den folgenden Abbildungen dargestellt. Falls


Abb. 4.22: Objekt Info vor dem Sekretariat

zusätzliche Informationen zu einem Raum vorhanden sind, werden sie als Buttons dargestellt. Wie in vorherigen Abschnitten erläutert, können diese Informationen in Form von Video, Audio, Text oder Bild präsentiert werden. Der Benutzer kann durch das Drücken eines entsprechend bezeichneten Buttons, welche sich rechts neben den personspezifischen Informationen befindet, von jedem Mitarbeiter ein Foto abrufen. Für die Räume werden jeweils drei Knöpfe mit den Beschriftungen „Video“, „Audio“ und „Zusatzinfo“ neben den raumspezifischen Informationen dargestellt. Die Reihenfolge und Anordnung der Buttons kann der Abbildung 4.23, Markierung 3 entnommen werden. Ferner wird ein „zurück“-Button angezeigt, der den aktuellen Zustand in „Hauptmenü“ zurückführt.

In Abbildung 4.24 wird der Fall dargestellt, wie ein Benutzer die Option „Foto“ auswählt. Dieses Foto erscheint zentriert in einem hellblauen Bilderrahmen. Zudem soll die Reihenfolge der bisher gewählten Buttons abgebildet werden, um den gegenwärtigen Zustand zu veranschaulichen. Auch hier gibt es die Option „zurück“ mit einer entsprechenden Rückführung in den vorherigen Zustand.

Sollte sich der Anwender für die Funktion „Zusatzinfo“ entscheiden, erscheinen zusätzliche Informationen über das Objekt in einem zentrierten Feld, wie in Abbildung 4.25 zu sehen ist. Diese Zusatzinformationen sind spezielle Informationen, die mit dem Raum in Zusammenhang stehen. In unserem Beispiel wird angezeigt, dass die Bibliothek der „Professur für GDV“ sich im Sekretariat befindet und wann die Öffnungszeiten der Bibliothek sind.

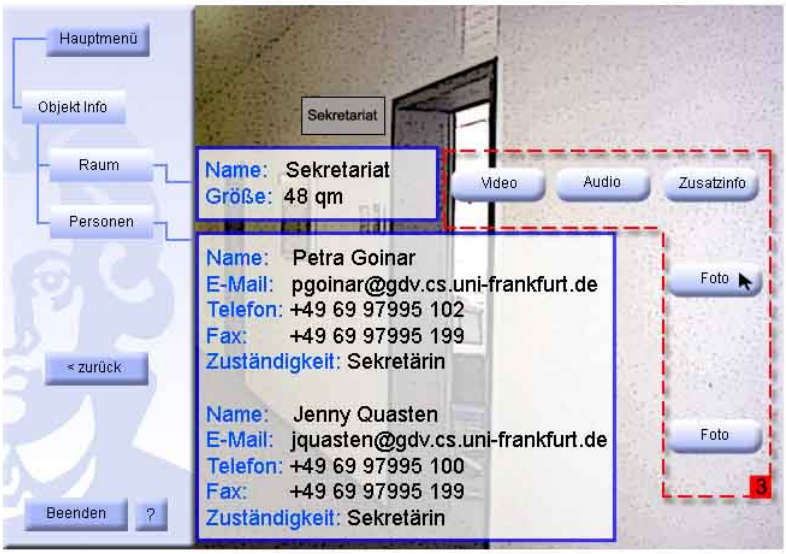


Abb. 4.23: Entwurf der Oberfläche bei der Wahl von Objekt Info


Abb. 4.24: Oberflächenentwurf bei der Anzeige von Mitarbeiterfotos

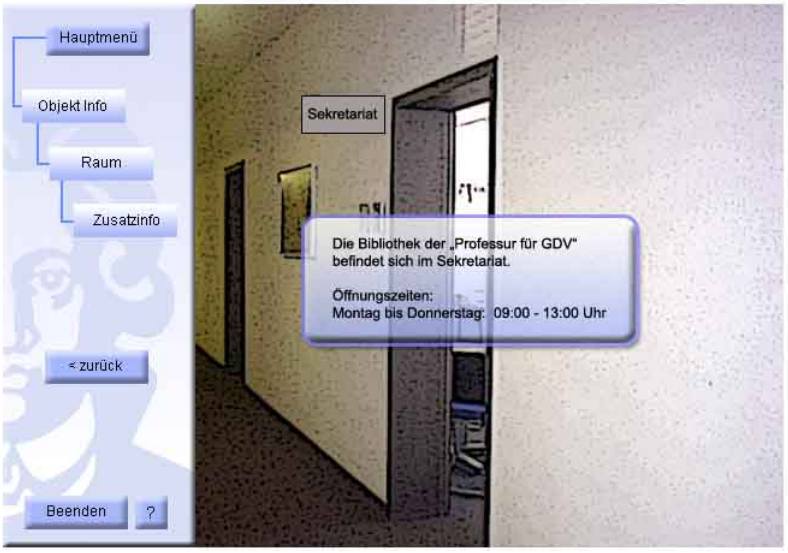


Abb. 4.25: Entwurf der Oberfläche nach Drücken auf Zusatzinfo

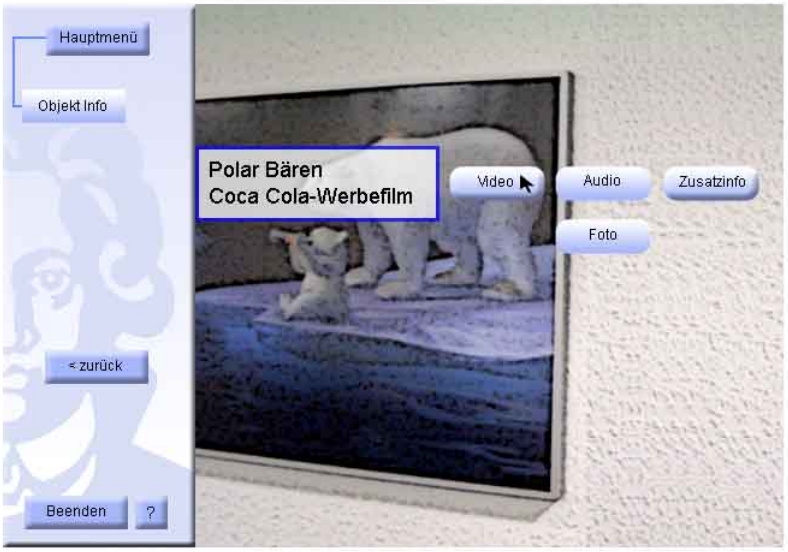


Abb. 4.26: Objekt Info vor einem Kunstwerk

Die Funktion „Objekt Info“, wie oben schon erwähnt, steht mit einem ähnlichen Oberflächendesign, auch für die Kunstwerke im Gebäude zur Verfügung. Allerdings ist für die Kunstwerke eine andere Art von Information relevant. Hierbei werden der Name des Künstlers und des Kunstwerkes sowie Hintergrundwissen über das Kunstwerk vorgestellt. Auch hier können zusätzliche Informationen auf gleiche Art wie oben dargestellt optional angezeigt werden. In Abbildung 4.26 wird der Fall demonstriert, wie der Benutzer sich in der Nähe von dem Bild „Polar Bären“ im Flur des Gebäudes befindet. Diese Buttons sind nur dann aktiv, wenn für sie Informationen hinterlegt sind. In Abbildung 4.26 ist der Button „Zusatzinfo“ aktiv und der Button „Audio“ nicht.

### 4.4.4 Lageplan

Damit sich der Anwender im Gebäude gut orientieren kann, kann er vor jedem Raum den Übersichtsplan des Gebäudes aufrufen. Dies geschieht wie folgt: Zunächst muss der Button „Hauptmenü“ ausgewählt werden. Danach wird das Untermenü „Lageplan“ ausgewählt, wie in Abbildung 4.27, Kasten 1 und 2 zu sehen ist. Durch das Klicken dieses Buttons erscheint der Lageplan zentriert unten auf dem Bildschirm, wie in Abbildung 4.28 demonstriert wird. Der Standort des Benutzers soll auf dem Lageplan blau markiert sein. Damit der Benutzer über den zuletzt eingegebenen Befehl informiert bleibt, wird dieser Befehl in Form eines inaktiven Buttons auf dem Logo angezeigt. Für die Funktion „Lageplan“ wird dies in Abbildung 4.28 veranschaulicht.


Abb. 4.27: Oberflächenentwurf für Lageplan 1

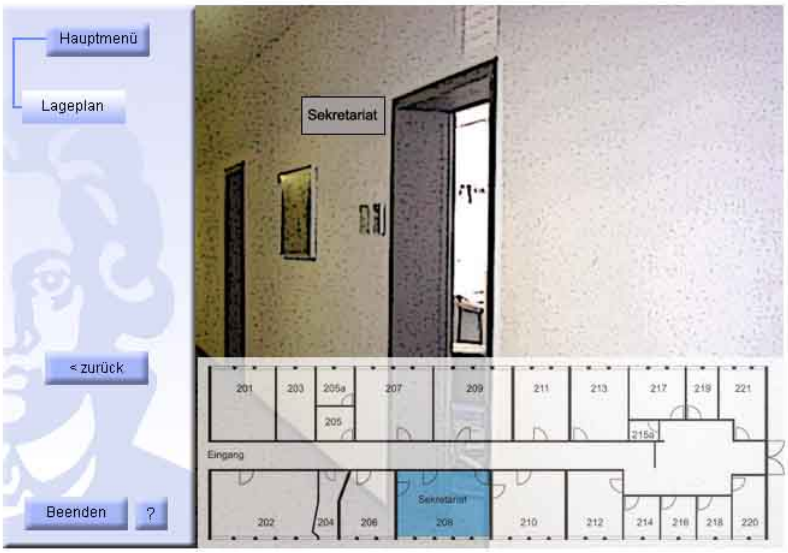


Abb. 4.28: Oberflächenentwurf für Lageplan 2

### 4.4.5 Hilfe

Die Hilfefunktion soll auf einer halbtransparenten hellblauen Fläche in einem rechteckigen Rahmen zentriert auf dem Bildschirm angezeigt werden. Der Text soll in schwarzer Schrift verfasst werden. In der oberen linken Ecke des Quadrates soll ein Fragezeichen eingeblendet werden. Auf der rechten Seite soll ein Button zum Schließen des Hilfefensters angebracht werden. Optional soll ein Button zum Anzeigen von Zusatzinformationen oder Graphiken unterhalb des Hilfefensters erstellt werden. Der Textinhalt der Hilfe soll kontextbezogen und leicht verständlich sein. Beim Schließen des Hilfefensters soll der Benutzer in den vorherigen Zustand zurück versetzt werden, von dem aus die Hilfe aufgerufen worden ist.

Abbildung 4.29 stellt ein Entwurf für die Hilfefunktion bei der Wahl der Option „Hauptmenü“ dar. Um eine einfache Benutzung des Systems für die Benutzer zu ermöglichen, soll im Zustand „Hauptmenü“ kurz und einfach der Umgang mit dem Eingabegerät erläutert werden. Damit ist zum Beispiel das richtige Halten des Gerätes und die Erläuterung der Hauptfunktionen „Navigation“, „Objekt Info“ und „Lageplan“ gemeint. Die unten dargestellten Abbildungen zeigen ein Beispiel für diesen Ablauf. Drückt der Benutzer den Button „Bild“, erscheint ein Foto von einem Benutzer, der zeigt, wie ein Anwender idealerweise das Gerät halten sollte. Das Foto selbst erscheint in einem abgegrenzten blauen Quadrat, wie in Abbildung 4.30 zu sehen ist.


Abb. 4.29: Oberflächenentwurf für Hilfe 1


Abb. 4.30: Oberflächenentwurf für Hilfe 2

Hiermit ist die Spezifikation der „MR-Referenzanwendung“ vollständig. In diesem Kapitel wurden zusätzlich zu den Anforderungen und Zielen der „MR-Referenzanwendung“, ihre Funktionen, Anwendungsfälle und Design ihrer graphischen Benutzeroberfläche beschrieben. Diese Spezifikation ist unabhängig vom AMIRE erstellt worden. Im nächsten Kapitel wird diese Spezifikation auf AMIRE überführt.

# Kapitel 5

# Überführung der Spezifikation

In diesem Kapitel wird die Spezifikation der „MR-Referenzanwendung“ aus Kapitel 4 auf das MR-Autorensystem AMIRE, das im Unterkapitel 3.4 gewählt wurde, überführt. Aufgrund der Anforderungsanalyse im Kapitel 3 wurde das MR-Autorensystem AMIRE ausgewählt, da es die meisten gestellten Anforderungen erfüllt. Nun wird die Realisierung der „MR-Referenzanwendung“ speziell mit AMIRE näher betrachtet und im Detail beschrieben. Hierbei ist es wichtig zu erwähnen, welche Komponenten und Funktionen des MR-Autorensystems AMIRE zum Einsatz kommen.

## 5.1 AMIRE Arbeitsprozessphasen

Wie schon im Unterkapitel 3.3.5 erklärt, arbeitet AMIRE auf Basis von Komponenten. Dabei müssen die entsprechenden Komponenten ausgewählt, parametrisiert und miteinander verbunden werden. Für die Oberfläche der „MR-Referenzanwendung“ werden zum Beispiel GUI-Elemente wie Buttons oder Graphiken verwendet, die jeweils als Komponenten in die „MR-Referenzanwendung“ eingebunden werden. Das geschieht bei AMIRE in den bereits im Unterkapitel 3.3.5 beschriebenen Phasen „Qualifikation“, „Adaption“, „Kombination“ und „Kalibrierung“, die im Folgenden detailliert erläutert werden. Diese Phasen spielen bei der Umsetzung der „MR-Referenzanwendung“ eine essentielle Rolle. Des Weiteren stellt AMIRE zahlreiche Dokumentationen und Anleitungen zur Unterstützung der Autoren an ([AMI04c], [AMI04a], [AMI04b]).

#### Qualifikation

Die erste Phase wird „Qualifikation“ genannt. Darin wählt der Autor die Komponenten aus, die in der MR-Anwendung zum Einsatz kommen. Die Komponenten werden hinsichtlich ihrer Funktion in der MR-Anwendung ausgewählt und werden aus einer Komponentenbibliothek entnommen. Abbildung 5.1 visualisiert den oben beschriebenen Vorgang.

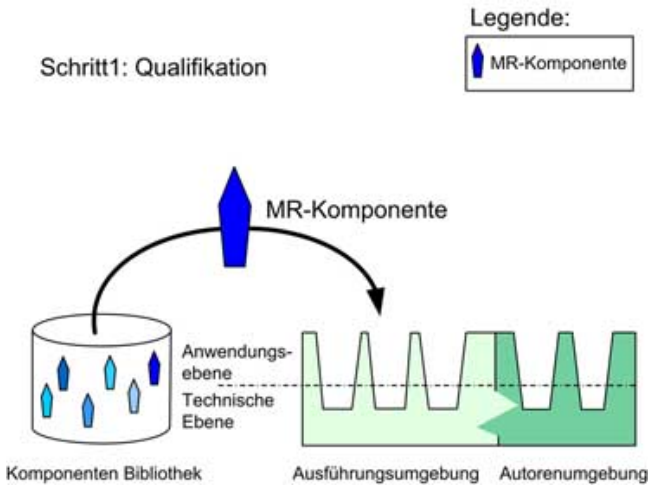


Abb. 5.1: Qualifikationsphase (vgl. [AD04])

Für die Erstellung der „MR-Referenzanwendung“ bedeutet diese Phase, dass der Autor die „MR-Referenzanwendung“ dahingehend analysieren muss, dass erkennbar wird, welche Komponenten er benötigt. Dabei können viele Beispiele aus AMIRE-Dokumentationen den Autor unterstützen, um die richtigen Auswahl zu treffen. Ferner gibt es das Werkzeug „Prototypes and Tools“, das dem Autor ermöglicht, die Komponenten aus einer Bibliothek einfach auszuwählen und in die „MR-Referenzanwendung“ einzubetten. Dieses Werkzeug wird im Unterkapitel 6.1 näher beschrieben. Wenn das Bild einer Mitarbeiterin aus dem Sekretariat beispielsweise angezeigt werden sollte, dann muss eine „Image3D-Komponente“ aus der Bibliothek ausgewählt und in die „MR-Referenzanwendung“ eingebunden werden. Dies geschieht in der Qualifikationsphase.

Bei der Umsetzung der „MR-Referenzanwendung“ kommen hauptsächlich folgende Komponenten zum Einsatz, die unter dem jeweiligen Pfad im Fenster „Prototypes and Tools“ zu finden sind.

- **Application-Komponenten:**
  - „PickableMarker-Komponente“: zur Positionierung der virtuellen Objekten abhängig von realen Objekten.
    *„Generic.Application.Tracking.Marker.Singlemarker.PickableMarker Component“*
  - „Zentrale Zustandskomponente“: zur Steuerung von Zuständen der „MR-Referenzanwendung“.
    *„Generic.Application.Behavior.CentralStateComponent“*

  - „Image3D-Komponente“: zum Laden und Anzeigen von 2D-Graphiken. *„Generic.Application.Graphic.3D.Image3D“*

  - „Button2D-Komponente“: zum Darstellen von Buttons. *„Generic.Application.Graphic.UI.graphical.Button2D“*

  - „Audio-Komponente“: zum Laden und Abspielen von Sound-Dateien. *„Generic.Application.Misc.AudioComponent“*

  - „Video3D-Komponente“ *: zum Laden und Abspielen von Video-Dateien. *„Generic.Application.Graphic.3D.Video3D“*

  - „FixPosition-Komponente“: zur Positionierung von Graphiken oder Buttons auf der Anzeige. *„Generic.Application.Tracking.FixPosition“*

- **MISC-Komponenten:**

  - „True-Komponente“: zum Sichtbarmachen von Graphiken oder Buttons. *„common.TrueComponent“*

- **Authoring-Komponenten:**

  - „VideoControl-Komponente“, „Connection-Editor“ und „Matrix-Editor“, die im Kapitel 6 beschrieben werden.

### Adaption

In der zweiten Phase „Adaption“ werden die Komponenten an die aktuellen Anforderungen der „MR-Referenzanwendung“ angepasst. Abbildung 5.2 stellt diese Phase graphisch dar. Bei der Erstellung der „MR-Referenzanwendung“ muss der Autor in dieser Phase die Eigenschaften der jeweiligen Komponenten verändern. Um das oben genannte Beispiel weiter auszuführen, muss nun die ausgewählte „Image3D-Komponente“ an die „MR-Referenzanwendung“ angepasst werden. Es müssen Eigenschaften wie die Größe der Graphik auf der Anzeige bestimmt werden, um den Anforderungen zu entsprechen. Dies geschieht in der Adaptionsphase. AMIRE bietet ein Editor „Properties“, in der die Eigenschaften leicht angepasst werden. Dieser Editor wird im Unterkapitel 6.1 näher beschrieben.

*Da die „Video3D-Komponente“ Probleme beim Laden verursacht, wurde stattdessen die „Video3D (clab)-Komponente“ verwendet. *„Generic.Application.Graphic.3D.Video3D (clab)“*

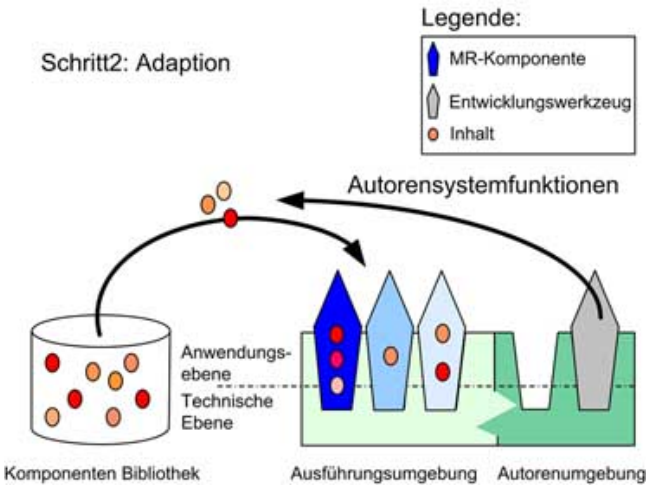


Abb. 5.2: Adaptionsphase (vgl. [AD04])

### Kombination

In der dritten Phase „Kombination“ werden nun die gewählten Komponenten miteinander verbunden. Für die „MR-Referenzanwendung“ bedeutet diese Phase, dass der Autor die Beziehungen zwischen den gewählten Komponenten und dem Framework festlegen muss. Abbildung 5.3 visualisiert diese Phase. Eine graphische Benutzerschnittstelle namens „Connection-Editor“ unterstützt den Autor bei der Durchführung dieser Phase. Dieser Editor wird im Unterkapitel 6.1 näher beschrieben. Wenn wir das obere Beispiel weiter führen und davon ausgehen, dass laut Anforderungen die oben ausgewählte „Image3D-Komponente“ nur dann erscheinen soll, wenn ein bestimmter Marker vom System erkannt wird, dann wird in dieser Phase die Beziehung zwischen der „Image3D-Komponente“ und dem bestimmten Marker beschrieben, siehe Unterkapitel 2.2.

### Kalibrierung

In Phase vier der „Kalibrierung“ werden die virtuellen Objekte, die durch die Komponenten repräsentiert werden, mit den realen Objekten verbunden. Für die „MR-Referenzanwendung“ bedeutet dies, dass der Autor die Positionierung von beispielsweise 3D-Modellen bestimmen soll. Wenn in der „MR-Referenzanwendung“ zum Beispiel eine Statue als 3D-Modell zum Einsatz kommt, die an einer realen Säule ausgerichtet werden soll, dann muss die Statue auf der Säule fest angebracht werden. Sie darf nicht frei schweben. Dies geschieht in der Kalibrierungsphase mit Hilfe eines einfach zu be-

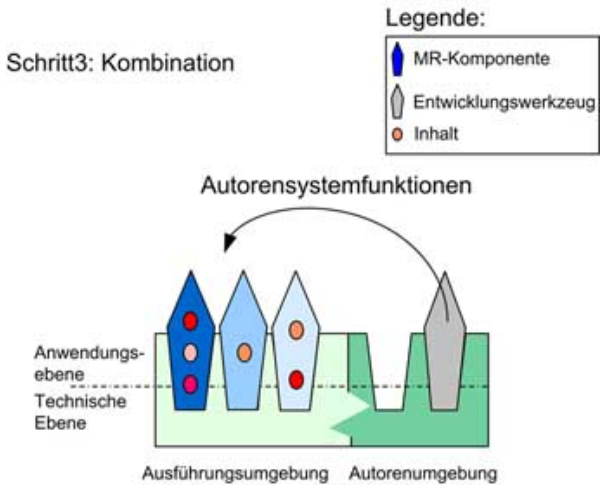


Abb. 5.3: Kombinationsphase (vgl. [AD04])

dienenden Werkzeugs namens „Matrix-Editor“. Die Werte für die Positionierung und Rotation der Statue können in diesem Editor eingegeben werden. Im Unterkapitel 6.1 wird dieses Werkzeug näher beschrieben.

Ferner ist noch zu erwähnen, dass der Ablauf dieser Arbeitsschritte nicht linear erfolgen muss. Das heißt der Autor kann die Reihenfolge dieser Phasen beliebig bestimmen und durch Iterationen dieser Phasen die „MR-Referenzanwendung“ vervollständigen.

## 5.2 Autorenrollen

Wie im Unterkapitel 3.3.5 erwähnt, wird bei AMIRE zwischen verschiedenen Autorenrollen unterschieden. Diese sind bei AMIRE in vier verschiedene Rollen aufgeteilt:

- Anwendungsautor, der Komponenten zur Produktion der MR-Anwendung verwendet.
- Komponentenautor, der Komponenten für den Anwendungsautor bereitstellt.
- Frameworkautor, der das Framework zur Einbettung von Komponenten bereitstellt.
- Werkzeugautor, der Werkzeuge für andere Autoren wie Anwendungsautoren, Komponentenautoren und Frameworkautoren bereitstellt.

In Abbildung 5.4 wird diese Aufteilung der Autorenrollen visualisiert. Autoren an der Spitze der Pyramide profitieren von den Produkten aller Autoren, die unter ihnen angesiedelt sind und vorab die technischen und schwierigeren Aufgaben bewältigen müssen

[AD04]. Ohne diese Rollenaufteilung ist es sehr schwierig, für einen Autor mit wenig MR-spezifischen Kenntnissen die MR-Anwendung zu ändern, auch wenn es sich nur um kleinere Änderungen handelt [AD04].

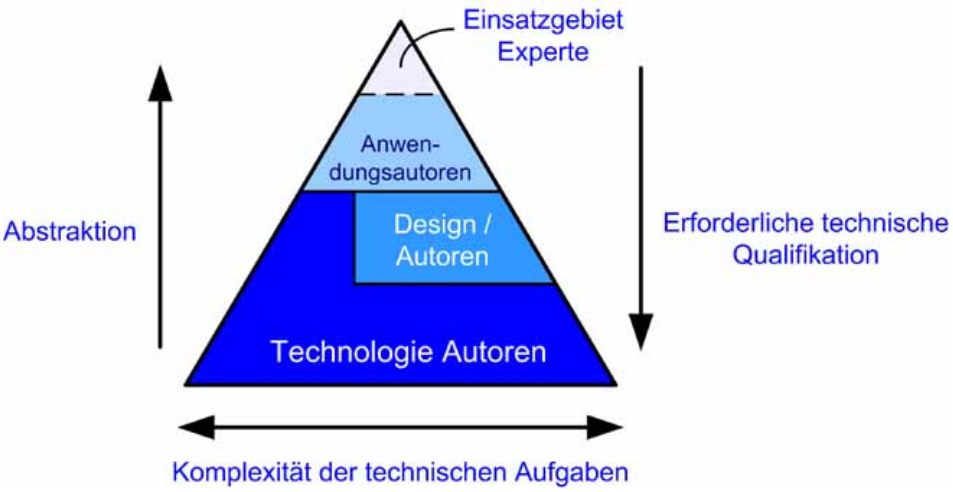


Abb. 5.4: Autoren Pyramide (vgl. [AD04])

Ein Autor nimmt während der Produktion der „MR-Referenzanwendung“ verschiedene der oben genannten Rollen ein. Bei der Erstellung der „MR-Referenzanwendung“ befindet sich der Autor in der Rolle des Anwendungsautors, während bei der Anfertigung der benötigten Objekte wie Graphik-, Audio- und Video-Dateien der Autor die Rolle des Design-Autors einnimmt. Andererseits übernimmt der Autor die Rolle des Einsatzgebietsexperten während er die „MR-Referenzanwendung“ testet.

## 5.3 Einzusetzende Komponenten

Eine MR-Anwendung wird in AMIRE als ein Netzwerk von verschiedenen Komponenten dargestellt. Wie im Unterkapitel 3.3.5 erläutert, besitzen Komponenten folgende Schnittstellen: „In-Slots“ und „Out-Slots“ zur Kommunikation, die jeweils für das Empfangen und Senden von Informationen verwendet werden. Sie sind jeweils mit Namen gekennzeichnet und können über ihren Namen erreicht werden [ZHHL03]. Abbildung 5.5 zeigt den Aufbau einer Komponente.

Als typische und oft verwendbare „Slots“ für die „MR-Referenzanwendung“ können folgende genannt werden, die für das Senden und Empfangen von Informationen zuständig sind:

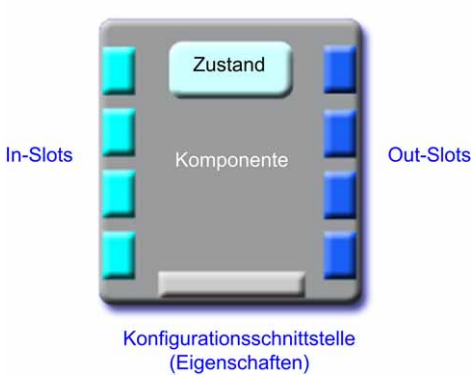


Abb. 5.5: Komponentenschnittstellen (vgl. [ZHHL03])

- **In-Slots:**
  - „visible“ hat einen booleschen Datentyp und ist für die Steuerung der Sichtbarkeit einer Komponente zuständig. Durch den „visible-In-Slot“ wird gesteuert, ob zum Beispiel ein GUI-Element oder eine Graphik angezeigt werden soll oder nicht.
  - „transformation“ ist ein Datentyp, der Vektoren abbildet und ist für die Bestimmung der Position und Orientierung einer Komponente zuständig (um die Werte der sechs Freiheitsgrade festzulegen [Krö04]).
  - „activate“ hat einen booleschen Datentyp und ist für die Steuerung einer Komponente zuständig, die eine Aktion ausführen soll. Die „Audio-Komponente“ braucht beispielsweise zum Abspielen einer Audio-Datei ein „True-Signal“, das über diesen „activate-Slot“ zu empfangen ist.
  - „stop“ hat ähnlich wie „activate“ einen booleschen Datentyp und ist für die Steuerung einer Komponente zuständig, die eine Aktion beenden soll. Durch diesen „Slot“ werden zum Beispiel die Information empfangen, ob eine „Audio-Komponente“ mit dem Abspielen einer Audio-Datei aufhören soll oder nicht.
- **Out-Slots:**
  - „visible“ hat einen booleschen Datentyp und gibt an, ob eine Komponente gerade angezeigt wird. Durch den „visible-Out-Slot“ werden zum Beispiel die Informationen über die Sichtbarkeit einer Graphik weitergegeben.
  - „transformation“ ist ein Datentyp, der Vektoren abbildet und ist für das Weitergeben der Position und Orientierung einer Komponente zuständig (um die Werte der sechs Freiheitsgrade weiterzugeben [Krö04]).

AMIRE stellt dem Autor die Möglichkeit zur Verfügung, mittels einer benutzerfreundlichen GUI Komponenten zu verknüpfen. Dadurch wird der Erstellungsprozess von MR-Anwendungen vereinfacht, was den Einsatzbereich von AMIRE deutlich vergrößert. AMIRE bietet eine Sammlung von Komponenten an, die bei Bedarf verändert oder erweitert werden kann. Abbildung 5.6 zeigt, wie verschiedene Komponenten miteinander verbunden werden. Diese Verbindung besteht zwischen dem Ausgang (Out-Slots) der einen Komponenten und dem Eingang (In-Slots) der anderen, wie in dieser Abbildung zu sehen ist. Im AMIRE-Framework können nur gleiche Datentypen miteinander verbunden werden.

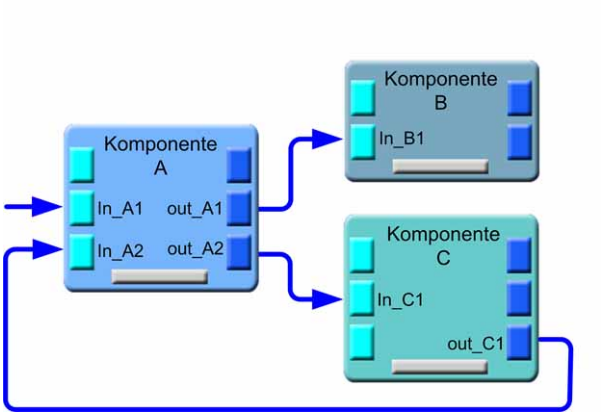


Abb. 5.6: Verbindungen zwischen den Komponenten (vgl. [ZHHL03])

Zusätzlich können Eigenschaften von Komponenten definiert werden. Durch diese Eigenschaften werden zum Beispiel Dateinamen von 2D-Graphiken oder ihre Größe festgelegt. Dabei kann eine MR-Anwendung durch Auswahl, Parametrisierung und Verbindung von entsprechenden Komponenten erzeugt werden. Für die Erstellung der „MR-Referenzanwendung“ gibt es folgende Hauptaufgaben, die ein Autor mit Hilfe des Autorensystems zu bewältigen hat:

- Bereitstellung von Marker für das Trackingsystem „ARToolKit“ zur Positionierung der virtuellen Objekten abhängig von realen Objekten, wie im Unterkapitel 2.2 erklärt.
- Aufteilung der „MR-Referenzanwendung“ in wiederkehrenden Zuständen wie zum Beispiel die Menüführung.
- Laden und Anzeigen von 3D-Modellen und 2D-Graphiken.
- Laden und Abspielen von Audio- und Videodateien.
- Darstellen von GUI-Elementen, wie zum Beispiel Buttons und Checkboxes.
- Herstellung von logischen Verknüpfungen, wie zum Beispiel AND-, OR-Bausteine.

Für die Erfüllung dieser Aufgaben werden geeignete Komponenten aus einer Bibliothek ausgewählt, die in einer hierarchischen und sehr überschaubaren Baumstruktur für Autoren zur Verfügung steht. Wie bereits im Unterkapitel 3.3.5 erwähnt, gibt es zwei Typen von Komponenten:

- Inhaltskomponenten, die in die „MR-Referenzanwendung“ eingebettet werden, wie zum Beispiel 3D-Geometrien oder Graphiken mit zusätzlichen externen Inhalten, wie 3D-Modelle oder Bilder.
- Werkzeugkomponenten, die den Autor bei der Erstellung der „MR-Referenzanwendung“ unterstützen, wie zum Beispiel „VideoControl-Komponente“, „Connection-“ und „Matrix-Editor“.

Für die Erstellung dieser Anwendung werden diese zwei Komponententypen eingesetzt. Einige von ihnen werden im Folgenden genauer beschrieben.

### PickableMarker-Komponente

Da in AMIRE das markerbasierte Trackingsystem „ARToolKit“ eingesetzt wird, müssen die Marker als Komponenten in die „MR-Referenzanwendung“ eingefügt werden, siehe Unterkapitel 2.2. Das bedeutet für die Qualifikationsphase, dass für jeden Marker eine „PickableMarker-Komponente“ gewählt werden muss. In der Adaptionsphase müssen die Eigenschaften dieser Komponenten angepasst werden. Für diese Komponente müssen beispielsweise entsprechende Markermuster gewählt werden. Das System kann diese Markermuster mit den realen Bildern vergleichen und dann erkennen, wenn diese Muster in der realen Umgebung vorhanden sind. Außerdem müssen ihre Maße oder andere Eigenschaften angepasst werden. Auch Marker besitzen wie alle anderen Komponenten in AMIRE „In-Slots“ und „Out-Slots“, durch die sie mit anderen Komponenten verbunden werden. Ein Lageplan soll beispielsweise in der „MR-Referenzanwendung“ erscheinen, sobald der Benutzer vor dem Sekretariat steht. Deshalb muss der an dem Sekretariat angebrachte Marker mit der „Image3D-Komponente“ „Sekretariatslageplan“ über einen „visible-Slot“ verbunden werden. Sobald das System diesen Marker in der Realität erkennt, sendet es ein TRUE über den entsprechenden „visible-Out-Slot“ zum „visible-In-Slot“ der „Image3D-Komponente“.

### Logische Komponenten

Logische Komponenten werden verwendet, um eine logische Verknüpfung zwischen verschiedenen Komponenten aufzubauen. Im Folgenden soll der Gebrauch dieser Komponente anhand von einem Beispiel gezeigt werden. Bei der „MR-Referenzanwendung“ werden an jedem Raum zwei Marker an einem Prisma angebracht, damit von beiden

Laufrichtungen ein Marker vom System erkannt werden kann, während der Benutzer den Gang durchläuft, siehe Unterkapitel 2.2. Deshalb sollten diese zwei Marker zunächst durch die logische „OR-Komponente“ verbunden werden, bevor sie mit weiteren Komponenten, wie zum Beispiel „Image3D- oder Audio-Komponente“, in Zusammenhang gebracht werden.

### Zentrale Zustandskomponente

Es gibt eine Komponente, die bei fast allen MR-Anwendungen, die mit AMIRE erstellt werden, zum Einsatz kommt. Diese ist die so genannte „Zentrale Zustandskomponente“ (ZZK), die den gesamten Ablauf der MR-Anwendung steuert. Abbildung 5.7 zeigt das grobe Prinzip der ZZK. Durch diese Komponente kann der Autor bestimmen, welche Aktion passiert, wenn ein oder mehrere Ereignisse auftreten. Durch diese Aufteilung der Komponenten in Aktion und Ereignis entstehen nun zwei neue Begriffe, die im Folgenden erläutert werden:

- „Zustandsaktivierer“: Besteht aus einer oder mehreren Komponenten, die ein Ereignis auslösen, beziehungsweise einen Zustandswechsel bewirken.
- „Zustandsabhängige“: Sind eine oder mehrere Komponenten, die durch einen Zustandswechsel aktiviert, beziehungsweise deaktiviert werden. Mit Hilfe der ZZK kann der Autor Zustandswechsel in allen Zuständen bestimmen.


Abb. 5.7: Zentrale Zustandskomponente

Wenn der Benutzer beispielsweise den Button „Hauptmenü“ auswählt (Ereignis), erscheinen die Untermenüs (Aktion). Beim Drücken des „zurück-Buttons“ wird die Rückführung in den vorherigen Zustand ermöglicht. Diese Komponente eignet sich hervorragend zur Unterstützung von wiederkehrendem und oft verwendetem Verhalten in der „MR-Referenzanwendung“, wie zum Beispiel die in der „MR-Referenzanwendung“

verwendeten Menüs. Sie ist so konzipiert, dass sie die vom Autor in verschiedenen Zuständen aufgeteilte Szene und den Übergang zwischen diesen Zuständen dirigiert.

In Abbildung 5.8 wird die Verwendung der ZZK im Bezug auf „MR-Referenzanwendung“ grob dargestellt. Darin werden die „Zustandsaktivierer“ und „Zustandsabhängige“ im „Start-Zustand“ angezeigt und in ihrer Abhängigkeit mit der ZZK verbunden.

- „Zustandsaktivierer“ sind „Button2D-Komponenten“ „Hauptmenü“ und „Beenden“. Sie können einen Ihnen zugeordneten Zustand aktivieren.
- „Zustandsabhängige“ sind: „Image3D-Komponente“, „Audio-Komponente“, welche für die Begrüßung des Anwenders zuständig sind, und die „Button2D-Komponente“, die der Hilfsfunktion zuzuordnen ist. Diese sind in der „MR-Referenzanwendung“ vom „Start-Zustand“ abhängig.

Die ZZK bei AMIRE bietet eine maximale Anzahl von 100 Zuständen an, was in der Abbildung 5.8 durch die Nummerierung N<=100 verdeutlicht wird. Die „True-Komponente“ bewirkt, dass die damit verbundenen Komponenten in der „MR-Referenzanwendung“ immer sichtbar sind. Die „FixPosition-Komponente“ liefert die vom Autor definierten Werte für die sechs Freiheitsgrade [Kröm04] für die „Image3D-Komponente“, die unverändert bleiben.

#### Image3D-Komponente

Für die Integration von Bildern in 3D in der „MR-Referenzanwendung“ wird eine „Image3D-Komponente“ verwendet. Es gibt gewisse Eigenschaften, wie zum Beispiel die Größe oder die Positionierung der Graphiken, die im Fenster „Properties“ bestimmt werden können. Die Bilder können mit Hilfe von einem Bildbearbeitungsprogramm, wie zum Beispiel Adobe Photoshop [Ado04] erstellt und in einem Unterordner im AMIRE-Programmverzeichnis untergebracht werden. Im Fenster „Properties“ muss für jedes Bild die richtige Datei gewählt werden. Der Vorteil der Rollenaufteilung in AMIRE kann hier gut eingesetzt werden. Somit können die Designer Graphiken erstellen, während die Anwendungsentwickler sich mit den Funktionalitäten der „MR-Referenzanwendung“ beschäftigen. Dadurch können mehrere Entwickler parallel an der Produktion arbeiten. Dies gilt jedoch auch für weitere Multimedia-Inhalte, wie zum Beispiel für Audio-, Video, oder Text-Dateien. Auch hier können die Inhalte mit Standardtools erzeugt und später in AMIRE verwendet werden.

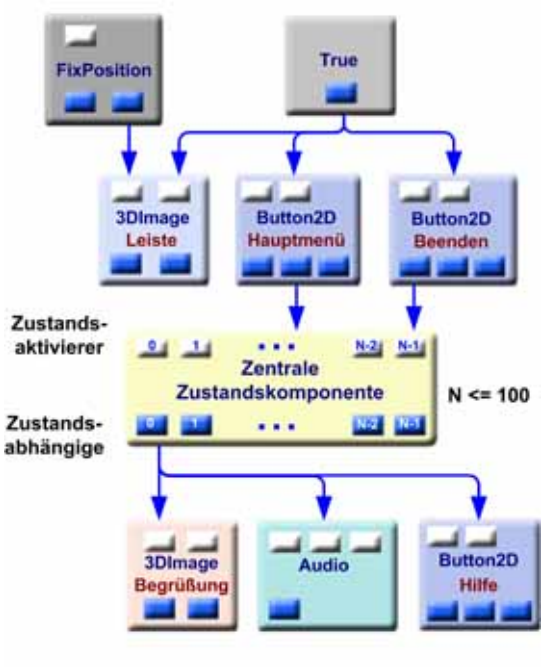


Abb. 5.8: Komponenten des Start-Zustandes

Für den Fall, dass mehrere Bilder aufeinander liegen, gibt es die Möglichkeit, die Reihenfolge dieser Graphiken zu bestimmen. AMIRE bietet als Lösung für dieses Problem eine „Translate-Funktion“, die die Anordnung der Objekte festlegt. Dabei muss man den Wert für die „Z-Achse“ verändern, um zu bestimmen, welche Objekte vorne und welche hinten erscheinen sollen. Für die „MR-Referenzanwendung“ gibt es zum Beispiel auf der linken Seite der Anzeige einen Bereich, der das Logo der Johann Wolfgang Goethe-Universität beinhaltet, worauf die „Hauptmenü-Buttons“ positioniert werden sollen. Mit Hilfe dieser Funktion kann die Reihenfolge und Anordnung dieser Objekte eingestellt werden, so dass dieser Bereich immer hinter den Buttons und nie davor erscheint.

### FixPosition- und True-Komponente

Damit die Graphiken an einer festen Position auf dem Bildschirm angezeigt werden, gibt es eine „FixPosition-Komponente“. Für die „MR-Referenzanwendung“ gibt es Graphiken oder Buttons, die immer sichtbar sein sollen, wie zum Beispiel der Bereich mit dem Logo auf der linken Seite des Anzeigefensters oder der „Hauptmenü-Button“. Diese müssen mit einer „True-Komponente“ verbunden werden, damit sie immer sichtbar sind.

### GUI-Elemente

Für Buttons wird eine „Button2D-Komponente“ ausgewählt. In der Phase „Adaption“ müssen die Eigenschaften dieser Komponente angepasst werden. Als Eigenschaften können solche Parameter, wie zum Beispiel Name, Größe, Hintergrundfarbe und Schriftfarbe bestimmt werden. Auch die Positionierung der Buttons im Anzeigefenster kann in dieser Phase durch den Autor festgelegt werden. Zusätzlich besteht die Möglichkeit, die Buttons in einem Bildbearbeitungsprogramm in verschiedenen Formen und Farben zu erstellen. Für die „MR-Referenzanwendung“ werden zum Beispiel Buttons in zwei verschiedenen Formen eingesetzt: eckige und ovale mit unterschiedlichen Farben. In der Adaptionsphase können solche Graphiken für den Hintergrund der Buttons gewählt werden.

### Audio-Komponente

Eine weitere Komponente, die in der „MR-Referenzanwendung“ eingesetzt wird, ist die „Audio-Komponente“, die über die „Slots“ „activate“ und „stop“ gesteuert werden kann. Durch diese Komponente können Audio-Dateien abgespielt werden. Die Audio-Dateien können als gängige Formate, wie zum Beispiel WAV oder MP3 erstellt und in AMIRE verwendet werden. In der Adaptionsphase muss beim Anpassen der Eigenschaften die richtige Datei vom Autor gewählt werden. Das System kann die Audio-Dateien abspielen, wenn ein Marker vom System erkannt oder ein bestimmter Zustand erreicht wird. Beim Starten der „MR-Referenzanwendung“ soll zum Beispiel der Anwender auditiv begrüßt werden. Dies kann durch die Verbindung des „activate-Slots“ der „Audio-Komponente“ mit dem „Start-Zustand“ der „Zentralen Zustandskomponente“ erreicht werden.

### Video3D-Komponente

Die für die „MR-Referenzanwendung“ benötigten Video-Dateien können mit Hilfe von einer „Video3D-Komponente“ * eingesetzt und abgespielt werden. Als Format, für die Speicherung der Video-Dateien kann AVI verwendet werden. In der Adaptionsphase soll der richtige Pfad der Video-Datei gewählt werden. Es gibt auch andere Eigenschaften, wie zum Beispiel die Auflösung der Video-Datei oder ihre Positionierung im Anzeigefenster, die in der Adaptionsphase bestimmt werden. Auch hier kann, wie bei der „Audio-Komponente“, die Datei abhängig vom Erreichen eines bestimmten Zustands oder vom Erkennen eines oder mehrerer Marker abgespielt werden. Dies wird vom Autor in der Kombinationsphase bestimmt.

*Da die „Video3D-Komponente“ Probleme beim Laden verursacht, wurde stattdessen die „Video3D (clab)-Komponente“ verwendet. *„Generic.Application.Graphic.3D.Video3D (clab)“*

**Text3D-Komponente**

Zur Anzeige von Meldungen, wie zum Beispiel des Begrüßungstextes gibt es zwei Möglichkeiten. Einerseits kann eine „Text3D-Komponente“ eingesetzt werden. Der Vorteil dabei liegt darin, dass der zur verwendete Textinhalt mit wenig Aufwand in einem herkömmlichen Texteditor geschrieben und dann in einen Unterordner des AMIRE-Programmverzeichnises eingefügt werden kann. Nachdem in der Qualifikationsphase die „Text3D-Komponente“ gewählt und in die „MR-Referenzanwendung“ integriert wird, kann in der Adaptionsphase beim Anpassen der Eigenschaften dieser Komponente der richtige Pfad gewählt werden. Dieser Text kann bei Bedarf einfach modifiziert werden. Aus dem Aspekt der Wartbarkeit der „MR-Referenzanwendung“ kann diese Komponente vorteilhaft sein. Der Nachteil dieser Komponente liegt darin, dass man die Text-Datei nicht an verschiedene gewünschte Eigenschaften, wie zum Beispiel Schriftfarbe des Textes anpassen kann. Dadurch wird die Qualität der Ergebnisse verschlechtert.

Andererseits wäre eine alternative Lösung zum Beispiel, die Erstellung der Texte als Graphik und ihre Integration in die MR-Anwendung mit Hilfe von „Image3D-Komponente“. Für die „MR-Referenzanwendung“ wurde diese Lösung aus ästhetischen Gründen bevorzugt.

In diesem Kapitel wurde die aus Kapitel 4 spezifizierte „MR-Referenzanwendung“ auf AMIRE überführt. Im nächsten Kapitel wird nun ihre Realisierung mit AMIRE beschrieben.

# Kapitel 6

# Realisierung des Demonstrators

In diesem Kapitel wird die AMIRE-Autorenumgebung kurz vorgestellt. Darin werden die Benutzeroberfläche von AMIRE und die Werkzeuge erläutert. Anschließend wird beispielhaft die Umsetzung der „MR-Referenzanwendung“ beschrieben. Es werden die Vorgehensweisen für die Erstellung dieser Anwendung und die gesammelten Erfahrungen bei der Erstellung aufgeführt. Danach werden die Testszenarien der „MR-Referenzanwendung“ beschrieben, wobei die Testphase anhand desselben Beispiels erläutert wird. Zuletzt wird die „MR-Referenzanwendung“ beispielhaft mit Hilfe von Diagrammen und Screenshots dokumentiert.

## 6.1 AMIRE-Autorenumgebung

Die Benutzeroberfläche von AMIRE wird in Abbildung 6.1 dargestellt. Diese wird in folgende Bereiche aufgeteilt: Links findet man die „Menubar & Quickbuttons“, „Camera View“, „Messages“ und „Script“, rechts befinden sich „Scene“, „Properties“ und „Prototypes and Tools“. Die Anordnung der Fenster kann variieren, wenn die Auflösung des Bildschirms sich ändert. Dieser Wert kann beim Starten des Systems festgelegt werden.

#### Camera View

Im Fenster „Camera View“ wird die Sicht des Endbenutzers wiedergegeben. Bei dieser Sicht kann es sich sowohl um eine angeschlossene Kamera, als auch um ein zuvor aufgenommenes Video handeln. Dies ist ein großer Vorteil von AMIRE, weil dadurch die Erstellung und die Validierung nicht „live“ vor Ort durchgeführt werden müssen. Zur Handhabung von zuvor aufgenommenen Videos bietet AMIRE ein Werkzeug, mit dem die Wiedergabe kontrolliert werden kann. Das heißt, dass die Videos vor- und zurückgespult oder angehalten werden können. Dieses Werkzeug wird als „VideoControl-Komponente“ bezeichnet und befindet sich im Fenster „Prototypes and Tools“ unter dem Pfad „Generic.Authoring.Framework.VideoControlComponent“.

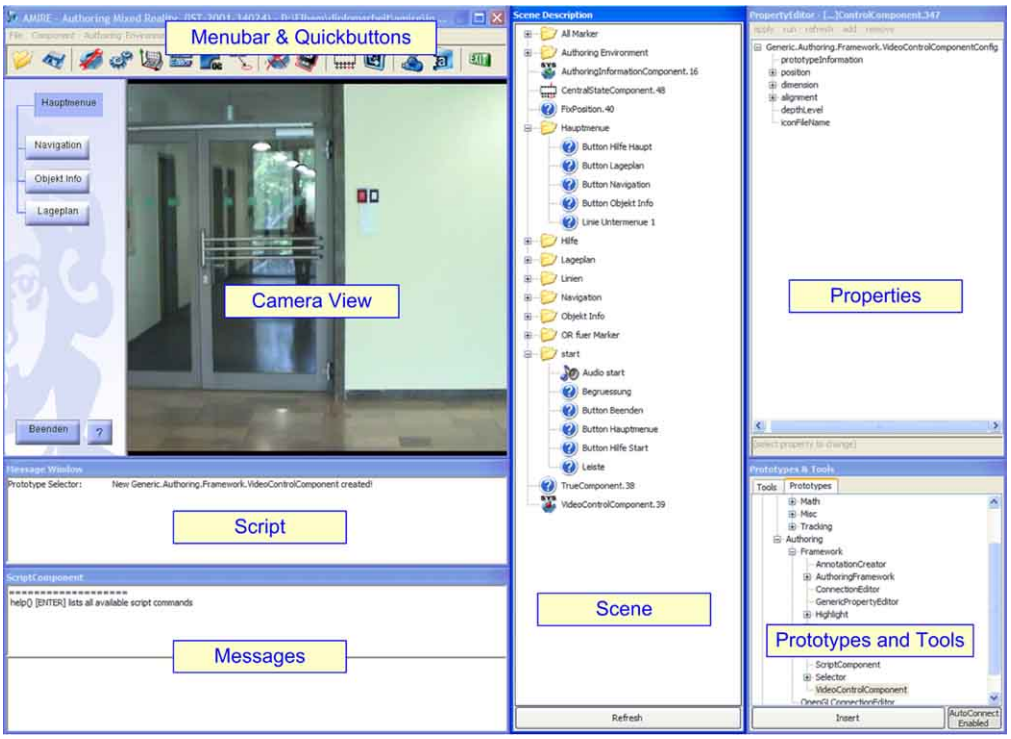


Abb. 6.1: Die AMIRE-Autorenumgebung (vgl. [AMI04c]

### Menubar & Quickbuttons

Oberhalb des „Camera Views“ liegen die „Menubar & Quickbuttons“, wie in Abbildung 6.2 zu sehen ist. Durch die „Menubar“ erreicht man das Systemmenü und durch die „Quickbuttons“ wird ein schneller Zugriff auf Systemwerkzeuge ermöglicht.


Abb. 6.2: Menubar & Quickbuttons

### Messages

Weiter unten auf der linken Seite ist das Fenster „Messages“ zu sehen, in dem Status-, Fehler- oder Warnmeldungen wiedergegeben werden.

### Script

Darunter wird das Fenster „Script“ dargestellt, in dem Skriptbefehle von AMIRE eingegeben und ausgeführt werden können. Wie in der Abbildung 6.3 zu sehen ist, wird dieses Fenster in zwei Bereiche aufgeteilt. Im unteren Bereich werden die Befehle eingegeben und im oberen Bereich werden die ausgeführten Befehle angezeigt.

Es besteht auch die Möglichkeit für einen Autor, Befehle in einer Text-Datei zu schreiben und sie später im „Script-Fenster“ über den Befehl: *run („../data/script/-beispiel.txt“)* auszuführen. Der Vorteil dabei liegt in einer Zeitersparnis. Für die „MR-Referenzanwendung“ kommen zum Beispiel 50 Marker zum Einsatz, siehe Unterkapitel 2.2. Mit Hilfe eines einfachen Skriptes können alle Marker stapelweise in diese Anwendung eingebunden werden. Die Liste der vorhandenen Befehle und der weiteren Details zur AMIRE „Script-Komponente“ können der Dokumentation von AMIRE [AMI04a] entnommen werden.


Abb. 6.3: AMIRE-Skriptfenster

### Prototypes and Tools

Die Phase „Qualifikation“, die im Kapitel 5 beschrieben wurde, wird durch die Auswahl der Komponenten im Fenster „Prototypes and Tools“ realisiert. Die auswählbaren Komponenten werden in einer hierarchischen Struktur zur Verfügung gestellt, welche die Suche nach geeigneten Komponenten für den Autor vereinfacht. In der hierarchischen Struktur wird eine Aufteilung der Komponenten vorgenommen. Diese Aufteilung bezieht sich zum Beispiel auf die „Application-Komponenten“, die in die Anwendung eingebunden werden, die „Authoring-Komponenten“, die während der Erstellung einer Anwendung als Werkzeuge eingesetzt werden und die „MISC-Komponenten“, die nicht unter „Application-“ oder „Authoring-Komponenten“ eingeordnet werden können. Durch Doppelklick kann der Autor die gewählten Komponenten in die Szene einfügen. Abbildung 6.4 zeigt das Fenster „Prototypes and Tools“.

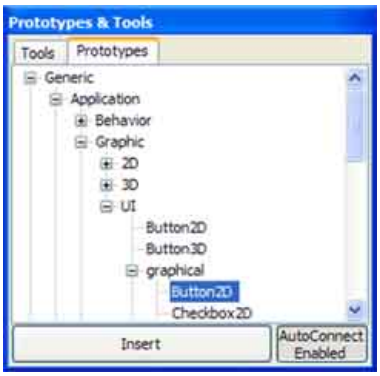


Abb. 6.4: Prototypes and Tools-Fenster

### Properties

Die im Kapitel 5 beschriebene Phase „Adaption“, bei der die Eigenschaften einer Komponente geändert werden können, wird im Fenster „Properties“ realisiert. Dieses Fenster wird in zwei Bereiche aufgeteilt, wie es in der Abbildung 6.5 zu sehen ist. Um die Eigenschaften an die „MR-Referenzanwendung“ anzupassen, kann im oberen Bereich der Name der zu ändernden Eigenschaft gewählt werden, siehe Abbildung 6.5, Markierung 1. Im unteren Bereich des Fensters können die gewünschten Werte eingetragen werden, siehe Abbildung 6.5 Markierung 2. In dieser Abbildung wird beispielsweise die Eigenschaft „ImageFilename“ einer „Image3D-Komponente“ angepasst.

Abb. 6.5: Properties-Fenster

### Connection-Editor

In der Phase „Kombination“, die im Kapitel 5 erläutert wurde, werden die Komponenten miteinander verbunden. Dies geschieht mit Hilfe des so genannten „Connection-Editor“, wie in Abbildung 6.6 zu sehen ist. Dabei müssen die „In-Slots“ und „Out-Slots“ der jeweiligen Komponenten miteinander verbunden werden. Wie aus dieser Abbildung zu entnehmen ist, wird auf der linken Seite des Editors eine Liste der verfügbaren „Out-Slots“ einer ausgewählten Komponente angezeigt. Auf der rechten Seite des Editors wird die Liste der vorhandenen „In-Slots“ einer anderen ausgesuchten Komponente bereitgestellt. Die Verbindung der passenden „Slots“ kommt durch eine entsprechende Markierung der „Slots“ und durch die Aktivierung des „Connect-Buttons“ zustande.

### Matrix-Editor

Die letzte Phase der Anwendungsproduktion in AMIRE ist die „Kalibrierung“, in

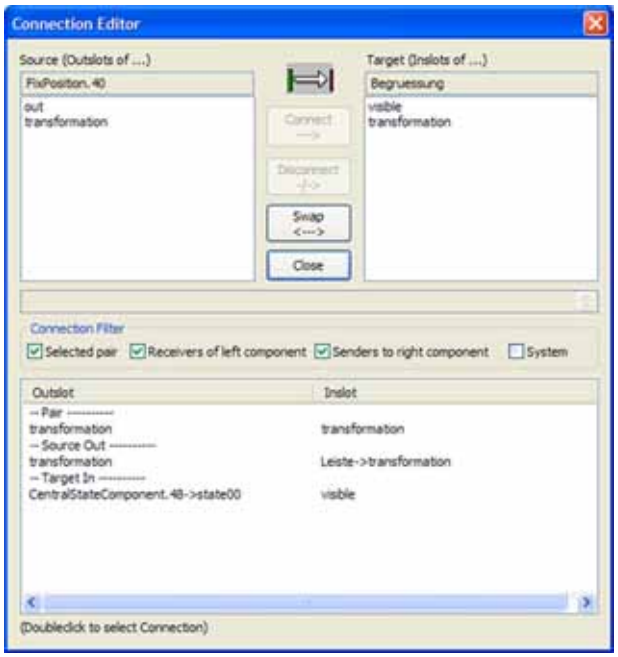


Abb. 6.6: Connection Editor

der die Verbindung zwischen der zu augmentierenden Umgebung und den virtuellen Komponenten realisiert wird. Die „Kalibrierung“ geschieht durch den „Matrix-Editor“, der in Abbildung 6.7 zu sehen ist. Der „Matrix-Editor“ hat die Aufgabe der Feinabstimmung eines virtuellen Objektes, wie zum Beispiel der Orientierung, Skalierung und Positionierung des Begrüßungstextfeldes für die Funktion „Start“. In AMIRE gibt es die Möglichkeit, noch während der Änderung dieser Daten das Ergebnis im „Camera View“ zu betrachten und dadurch einen guten Überblick über die Änderungen zu erhalten.

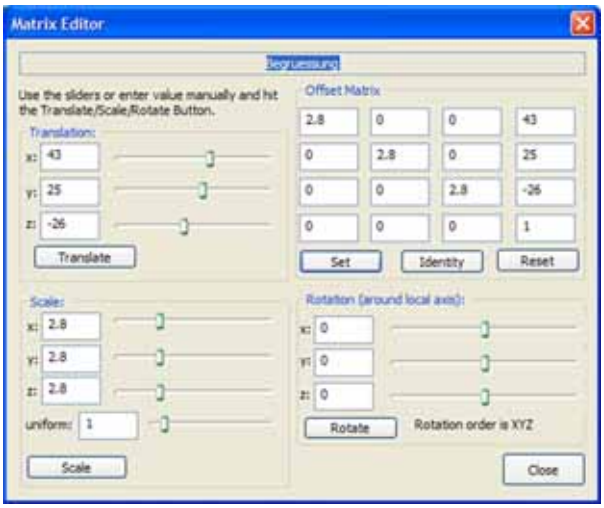


Abb. 6.7: Matrix Editor Dialog Fenster

### Scene

Alle Komponenten, die in der Anwendung eingesetzt werden, werden im Fenster „Scene“ abgelegt. Der Autor hat hier die Möglichkeit, die Komponenten in einer individuellen Reihenfolge mit Hilfe von Ordnern abzulegen. Dabei kann der Autor die Komponenten beliebig benennen und ihre hierarchische Struktur mit Hilfe von Drag & Drop bestimmen. Details zu der Bedienung des Autorensystems AMIRE können den Dokumentationen der Anwendung ([AMI04c], [AMI04a], [AMI04b]) entnommen werden.


Abb. 6.8: Scene-Fenster

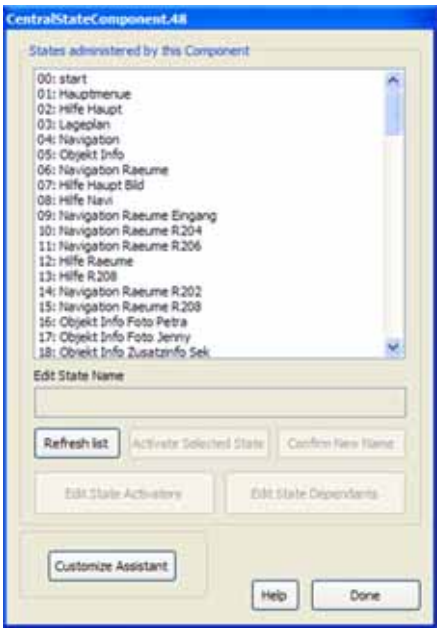


Abb. 6.9: Zentrale Zustandskomponente

### Zentrale Zustandskomponente

Es wird im Folgenden ein weiteres Werkzeug für die Bedienung und Steuerung der „Zentralen Zustandskomponente“ (ZZK) vorgestellt. Darin können für jeden Zustand die Komponenten „Zustandsaktivierer“ und „Zustandsabhängige“ bestimmt werden. Die ZZK bietet insgesamt 100 Zustände, die gesteuert werden können. Beim ersten Zustand ist der „Zustandsaktivierer“ vorgegeben und kann nicht geändert werden. Für diesen Zustand können nur die „Zustandsabhängige“ vom Autor bestimmt werden. Mit Hilfe der in Abbildung 6.9 gezeigten GUI können diese Zustände der ZZK definiert und benannt werden.

Durch das Drücken des Buttons „Edit State Activators“ wird ein weiteres Fenster geöffnet, in dem die „Zustandsaktivierer“ bestimmt werden, wie in der Abbildung 6.10 links zu sehen ist. Durch das Klicken des Buttons „Edit State Dependants“ öffnet sich das Fenster, in dem „Zustandsabhängige“ bestimmt werden können, wie es in Abbildung 6.10 rechts dargestellt wird. Der Autor hat hier die Möglichkeit, aus einer Liste die „Zustandsabhängige“ auszuwählen. Es gibt zwei Buttons „Add Selected DIRECTLY“ und „Add Selected INDIRECTLY“, die abhängig von der jeweiligen Situation vom Autor gewählt werden können. Der Autor soll die Entscheidung treffen, wie die „Zustandsabhängige“ mit einem Zustand der „Zentralen Zustandskomponente“ verbunden werden sollen. Dabei wird zwischen zwei Klassen unterschieden: „direkte“ und „indirekte Abhängigkeit“.

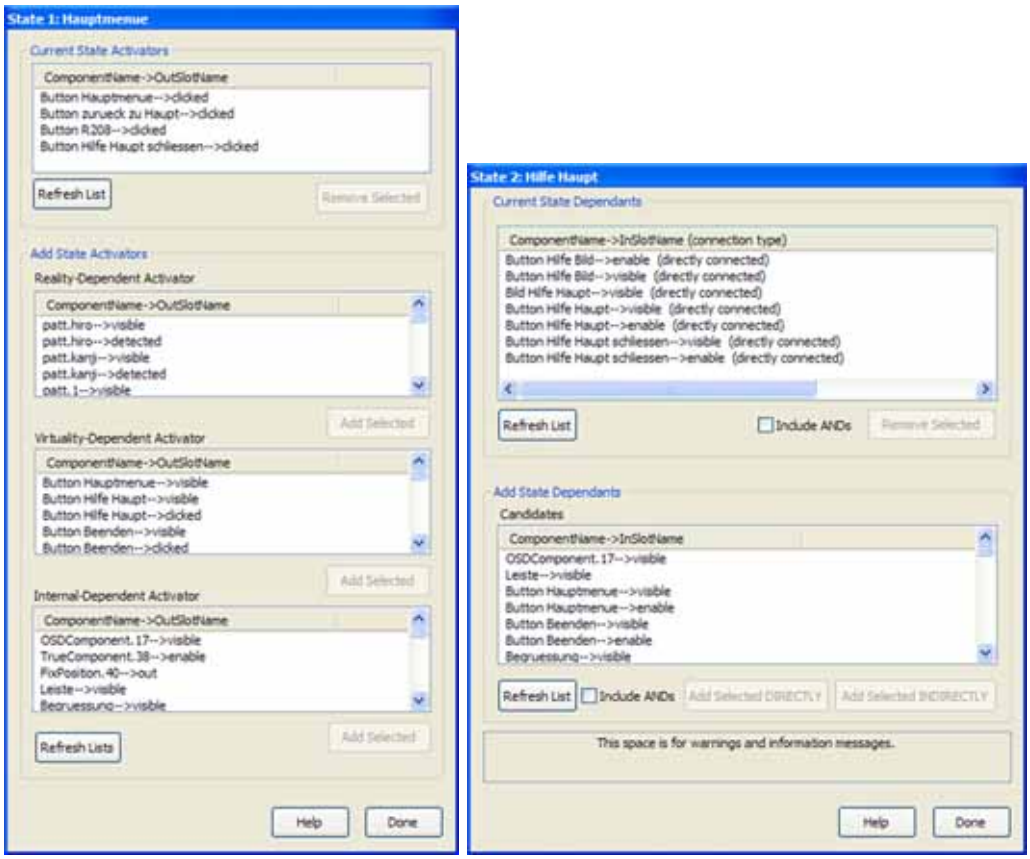


Abb. 6.10: Zustandsabhängige und Zustandsaktivierer

- Die „direkte Abhängigkeit“ wird dann gewählt, wenn allein der Zustand der ZZK der Auslöser für eine Aktion einer abhängigen Komponente ist. Für die Erstellung von Menü-Buttons in der „MR-Referenzanwendung“ wurde diese „direkte Abhängigkeit“ gewählt. Wenn beispielsweise „Hauptmenü-Button“ gedrückt wird, sollen die Untermenüs erscheinen.

- Die „indirekte Abhängigkeit“ wird in Situationen verwendet, in denen die Aktion einer „Zustandsabhängigen“ vom Zustand der ZZK und einem anderen Ereignis abhängig ist. Dabei wird durch die indirekte Abhängigkeit vom System automatisch eine logische „AND-Komponente“ erzeugt, die diese Verbindung ermöglicht. Diese Situation tritt zum Beispiel bei der Anzeige des Lageplans in der „MR-Referenzanwendung“ auf. Dabei soll der Lageplan abhängig von einem bestimmten Zustand und zusätzlich abhängig vom Erkennen eines oder zweier bestimmten Marker angezeigt werden.

Ein weiterer wichtiger Funktionsbutton ist der „Activate Selected State“, mit dem ein gewünschter Zustand aktiviert werden kann. Dies unterstützt den Autor bei der Erstellung und Validierung der Anwendung, weil er damit in jeden beliebigen Zustand wechseln kann. Für die „MR-Referenzanwendung“ wurden insgesamt 48 Zustände gebraucht und es wurden 363 Komponenten eingesetzt.

### Microsoft Visio

AMIRE setzt Microsoft Visio [Vis04] als ein Werkzeug ein, um MR-Anwendungen schematisch darzustellen. Die Integration von Microsoft Visio kann in zwei Richtungen stattfinden. Dabei können MR-Anwendungen, die mit dem AMIRE-Autorensystem erstellt wurden, in Microsoft Visio importiert, verändert und wieder gespeichert werden. Somit können MR-Anwendungen je nach Bedarf in Microsoft Visio oder in die AMIRE-Autorenumgebung erstellt, beziehungsweise bearbeitet werden. Die Entscheidung, ob Microsoft Visio zur Erstellung eingesetzt wird oder nicht, bleibt dem Autor überlassen. Durch den Einsatz von Microsoft Visio können die drei Arbeitsprozessphasen von AMIRE „Qualifikation“, „Adaption“ und „Kombination“ einfach durchgeführt werden. Darin kann der Autor die Szene in verschiedene Ebenen aufteilen und Teilszenen kommentieren [ADG04].

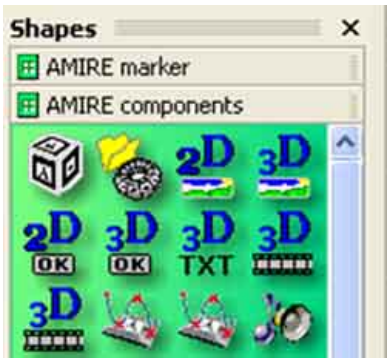


Abb. 6.11: AMIRE-Komponenten in Microsoft Visio

Wie Abawi [ADG04] weiterhin erklärt, können AMIRE-Komponenten bei Microsoft Visio als ikonisierte Symbole zur Verfügung gestellt werden, siehe Abbildung 6.11. Dadurch wählen die Autoren mit Hilfe von Drag & Drop die für die MR-Anwendung einzusetzenden Komponenten aus und ordnen sie auf einem Arbeitsblatt an. Die beiden Phasen „Adaption“ und „Kombination“ können mit speziell für Microsoft Visio entwickelten Werkzeugen realisiert werden. Um Veränderungen von Eigenschaften einzelner Komponenten zu ermöglichen, wird für die Adaptionsphase ein integriertes Modul angewandt, welches durch eine GUI bedient werden kann. In der

Kombinationsphase können Verbindungen zwischen den Komponenten innerhalb von Microsoft Visio mit Hilfe von Linien hergestellt werden.

Microsoft Visio eignet sich allerdings nicht besonders für die Kalibrierungsphase, da hier ein so genanntes „Previewproblem“ existiert, wie im Unterkapitel 3.3 beschrieben.

## 6.2 Umsetzung

Hier wird erläutert, welche Vorgehensweise für die Erstellung der „MR-Referenzanwendung“ ausgewählt wird und welche Vorbereitungen diesbezüglich durchgeführt werden müssen. Die Prinzipien der „Verbalisierung“ und „Dokumentation“ spielen bei dieser Umsetzung eine essentielle Rolle. „Verbalisierung“ bedeutet, Gedanken und Vorstellungen in Worten auszudrücken und damit ins Bewusstsein zu bringen [Bal00]. Bezogen auf die Entwicklung der „MR-Referenzanwendung“ soll „Verbalisierung“ dazu dienen, die Ideen und Konzepte der Umsetzung möglichst gut sichtbar zu machen und zu dokumentieren. Anschließend werden die während der Erstellung der „MR-Referenzanwendung“ gesammelten Erfahrungen vorgestellt.

### 6.2.1 Vorgehensweise

Für die Umsetzung der spezifizierten „MR-Referenzanwendung“ empfiehlt es sich, zunächst einige Vorbereitungen durchzuführen, bevor mit der Produktion der Anwendung begonnen wird.

Zunächst werden die für die Anwendung benötigten Marker erstellt und getestet, siehe Unterkapitel 2.2. Für die „MR-Referenzanwendung“ kommen insgesamt 50 Marker zum Einsatz, weil AMIRE maximal so viele unterstützt. Da AMIRE vom Hause aus nur 28 Marker mitbringt, werden zusätzliche Marker mit Hilfe des Werkzeugs *„mk_patt.exe“* erstellt. Dieses Werkzeug ist im Installationsumfang von AMIRE. Ferner werden Marker von Hannigan [Han04] von der Universität in Texas eingesetzt.

Es sind verschiedene Tests nötig, um herauszufinden, in welcher Größe die einzusetzenden Marker vom System am besten erkannt werden. Dabei spielen Eigenschaften wie Lichtverhältnisse im Gebäude und die Entfernung der Kamera eine wichtige Rolle. Die Ergebnisse dieser Tests zeigen, dass die Marker mit einer Größe von zehn Zentimetern für die „MR-Referenzanwendung“ optimal sind, weil sie einerseits vom System gut erkennbar, andererseits nicht zu groß sind, um die Aufmerksamkeit des Benutzers zu sehr in Anspruch zu nehmen. Ferner werden die Marker so positioniert, dass die „MR-Referenzanwendung“ nicht von der Körpergröße des Benutzers abhängt. Sie werden auf ein Prisma mit einem rechtwinkligen Dreieck als Grundfläche angebracht, damit die Marker mit einem Winkel von 45 Grad im Sichtfeld der Kamera stehen. Dadurch werden sie möglichst gut von beiden Richtungen während des Durchgangs durch das

Gebäude erkannt. Abbildung 6.12 zeigt eine Skizze für ein solches Prisma mit zwei Markern.

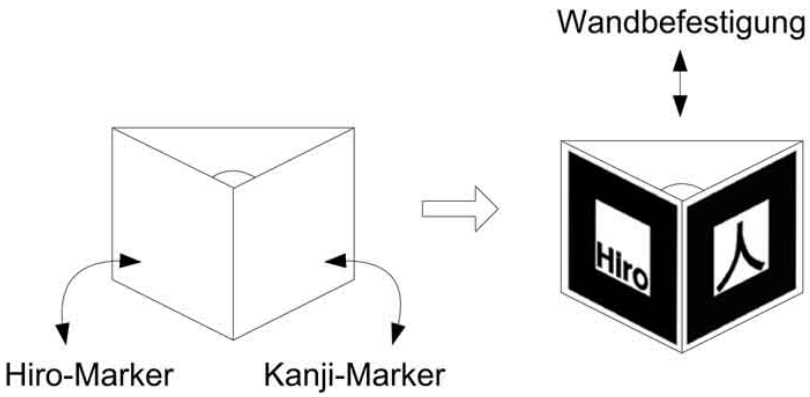


Abb. 6.12: Entwurf eines Prismas

Als nächstes wird ein Video erstellt, das einen Rundgang durch das Gebäude simuliert. Dieses Video wird bei der Erstellung und Validierung der „MR-Referenzanwendung“ eingesetzt.

Nun könnte der Autor die „Anwendungslogik“ planen, indem er diese anhand von Diagrammen auf Papier festlegt oder elektronisch verbalisiert. Darin sollten die verschiedenen Zustände und ihre Übergänge der „MR-Referenzanwendung“ aufgezeichnet werden. Zusätzlich sollten die Komponenten eventuell in „Zustandsaktivierer“ und „Zustandsabhängige“ eingeteilt werden. Diese Aufteilung ist nicht endgültig und kann für Komponenten oder Zustände, die später während der Umsetzung hinzukommen, erweitert werden. Die Einhaltung dieses Schrittes ist nicht zwingend und daher nicht ein Teil der Arbeitsprozessphase in AMIRE. Er hilft lediglich, logische Aufbaufehler zu vermeiden.

Abbildung 6.13 zeigt ein Beispiel für eine solche Planung der „Anwendungslogik“, die eine zusammengefasste Darstellung der Funktion „Lageplan“ beinhaltet. Darin werden die einzusetzenden Komponenten und ihre Verbindungen dargestellt. Die gezeichneten Rechtecke symbolisieren die Anwendungskomponenten. Die Verbindungspfeile stellen die Beziehungen zwischen den Komponenten dar.

Der Button „Lageplan“ in der Abbildung 6.13 aktiviert den Zustand 03 der ZZK und wird als „Zustandsaktivierer“ gekennzeichnet. Dieser Zustand ist in der Abbildung zufällig gewählt und kann in der „MR-Referenzanwendung“ vom Autor beliebig ausgesucht und benannt werden. Die Komponente „Image3D“ für den Lageplan wird als „Zustandabhängige“ mit dem Zustand 03 indirekt verbunden. Dies ist insofern wich-

tig, weil der Lageplan nur dann angezeigt werden soll, wenn der Zustand 03 aktiviert und mindestens einer von den zwei Markern „patt.11“ und „patt.12“ erkannt wird. Diese indirekte Verbindung erleichtert den Erstellungsprozess für den Autor, in dem die logische „AND-Komponente“ automatisch eingefügt wird. Die Verbindung zu diesen Markern bewirkt, dass der richtige Lageplan beziehungsweise die richtige Graphik-Datei, in der das Sekretariat mit blauer Markierung gekennzeichnet ist, angezeigt wird. Die „FixPosition-Komponente“ ist über den „transformation-Slot“ mit der „Image3D-Komponente“ verbunden, um dem Lageplan eine feste Position auf dem Bildschirm zuzuweisen.

Bisher wurden die virtuellen Objekte, wie beispielsweise der Lageplan, nicht zwangsweise in ihrer endgültigen Version benötigt. Da diese Objekte auch später in der Anwendung aktualisiert werden können, kann der Autor sogar mit Dummy-Objekten die Anwendung erstellen und sie später durch die Originale ersetzen.

### 6.2.2 Erfahrungen

Es sind Erfahrungen während der Erstellung der „MR-Referenzanwendung“ gesammelt worden, die zum Teil auf Fehler und Beschränkungen von AMIRE hinweisen. Die folgende Tabelle beschreibt sie und bietet gleichzeitig Lösungsvorschläge an.

| **Fehler/Beschränkung** | **Lösungsvorschlag** |
|---|---|
| Die Eigenschaften von ähnlichen Komponenten müssen jedes Mal erneut eingegeben werden. | Wenn beispielsweise mehrere Buttons die gleichen Eigenschaften, wie Farbe und Größe besitzen, sollten diese Eigenschaften untereinander kopiert, beziehungsweise in Form einer Formatvorlage weitergegeben werden können. |
| Bei größeren Anwendungen kann die Liste der Komponenten und Zustände in der ZZK unübersichtlich werden. | Es sollte eine Möglichkeit implementiert werden, womit Komponenten und Zustände alphabetisch geordnet werden können, damit das Suchen vereinfacht wird. |
| Es ist für nicht erfahrene Benutzer schwierig, Komponenten zu finden, wenn nur der Name bekannt ist. | Es sollte ermöglicht werden, alle vorhandenen Komponenten unabhängig von ihrer hierarchischen Anordnung alphabetisch aufzulisten. Dadurch können Komponenten schneller ausgesucht werden. |
| Fortsetzung folgt ... | |

| **Fehler/Beschränkung** | **Lösungsvorschlag** |
|---|---|
| Die Bezeichnung „Prototypes and Tools“ für dieses Fenster ist verwirrend. | Um Missverständnisse zu vermeiden, kann dieses Fenster in „Komponentenbibliothek“ umbenannt werden. |
| Die Positionierung von „Button2D-Komponente“ ist durch Eingabe von X-, Y-Koordinaten umständlich und benutzerunfreundlich. | Der existierende „Matrix-Editor“ könnte erweitert werden, so dass er für alle GUI-Elemente inklusive „Button2D-Komponente“ eingesetzt werden kann. |
| Um die Farbe von Buttons zu ändern, müssen diese durch Farbnummern eingegeben werden, was sehr benutzerunfreundlich ist. | Die Eingabe der Farbe der Buttons sollte durch eine graphische Farbpalettenauswahl erweitert werden. |
| Videos, die als Objekte in die MR-Anwendung eingebunden werden, können vom Anwender nicht zurück oder vorgespult werden. | Diese sollten steuerbar gemacht werden. |
| Die Anzahl der Marker ist auf 50 beschränkt. | Diese Einschränkung sollte aufgehoben werden, damit auch größere Anwendungen erstellt werden können. |
| Transparente Bilder in „Button2D-Komponente“ zeigen nicht den Ihnen direkt untergeordneten Hintergrund an, sondern das Video, beziehungsweise die Kameraansicht. | Durch Einführung einer Anordnung analog zur ähnlichen Standardsoftware könnte dieses Problem behoben werden. |
| In Microsoft Visio besteht die Möglichkeit, eine MR-Anwendung in verschiedene Ebenen aufzuteilen, die ein- und ausgeblendet werden können. Dies ist für große MR-Anwendungen sehr hilfreich. Allerdings besteht das Problem, dass man die Verbindungen nicht in Abhängigkeit von gewählten Ebenen anzeigen kann, sondern alle Verbindungen werden angezeigt. | Die Verbindungen sollten nur für die ausgewählte Ebene sichtbar sein. |
| | Fortsetzung folgt ... |

| **Fehler/Beschränkung** | **Lösungsvorschlag** |
|---|---|
| MR-Anwendungen, die mit AMIRE erstellt werden, weisen keinen Datenschutz vor Dritten auf. | Durch Verschlüsselung von MR-Anwendungen und deren Inhalte könnte die Sicherheit der sensiblen Daten erhöht werden. |
| AMIRE ist während der Erstellung der „MR-Referenzanwendung“ stetig langsamer geworden. So dass gegen Ende, jede Erweiterung der Anwendung die Arbeitsgeschwindigkeit erheblich beeinträchtigt hat. | Ein Lösungsvorschlag ist aufgrund fehlender technische Kenntnisse über das System nicht möglich. |

Tabelle 6.1: Verbesserungsvorschläge für AMIRE

## 6.3 Dokumentation

Nach Balzert [Bal00] ist eine geeignete Dokumentation ein wichtiger Bestandteil jedes Softwaresystems. Die Dokumentation für die „MR-Referenzanwendung“ wird vom Autor erstellt, da ohne sie die wichtigen Informationen, die während der Entwicklung gesammelt wurden, später verloren gehen können. Entwicklungsentscheidungen, wie zum Beispiel der Grund für die „indirekte Verbindung“, die zwischen Lageplan und „Zentraler Zustandskomponente“ besteht, müssen dokumentiert werden. Dies ist insofern wichtig, weil bei einer Modifikation oder Neuentwicklung die bereits gesammelten Erfahrungen eingesetzt werden können. Sinnvollerweise sollte diese Dokumentation vom Anwendungsautor erstellt werden, um die Aufwände der Dokumentationserstellung zu reduzieren.

Für die Dokumentation der „MR-Referenzanwendung“ wird das in AMIRE integrierte Werkzeug Microsoft Visio eingesetzt. Dieses Werkzeug eignet sich hervorragend für die Erstellung der Dokumentation. Dabei kann der Autor die Funktionen von Microsoft Visio uneingeschränkt benutzen. Dazu gehört zum Beispiel die Aufteilung der Szene auf verschiedenen Ebenen und die Kommentierung von Teilszenen.

Die Dokumentation der gesamten Anwendung sprengt den Rahmen dieser Arbeit. Deswegen werden hier exemplarisch nur zwei Funktionen der „MR-Referenzanwendung“ (die Funktionen „Start“ und „Navigation“) als Sonderfälle für die Umsetzung schematisch dargestellt. Diese Darstellung befasst sich mit den Zuständen, die den Funktionen untergeordnet sind. Bei dem ersten Zustand handelt es sich um den „Start-Zustand“, der gleichzeitig der erste der Anwendung ist und dadurch einen von AMIRE vorgegebenen „Zustandsaktivierer“ besitzt. Dieser „Start-Zustand“ erlaubt dem Autor nur das Bestimmen der Zustandsabhängigen. Bei dem anderen Zustand geht es um den Navigationszustand, der in der „MR-Referenzanwendung“ umfangreich und kompliziert ist.

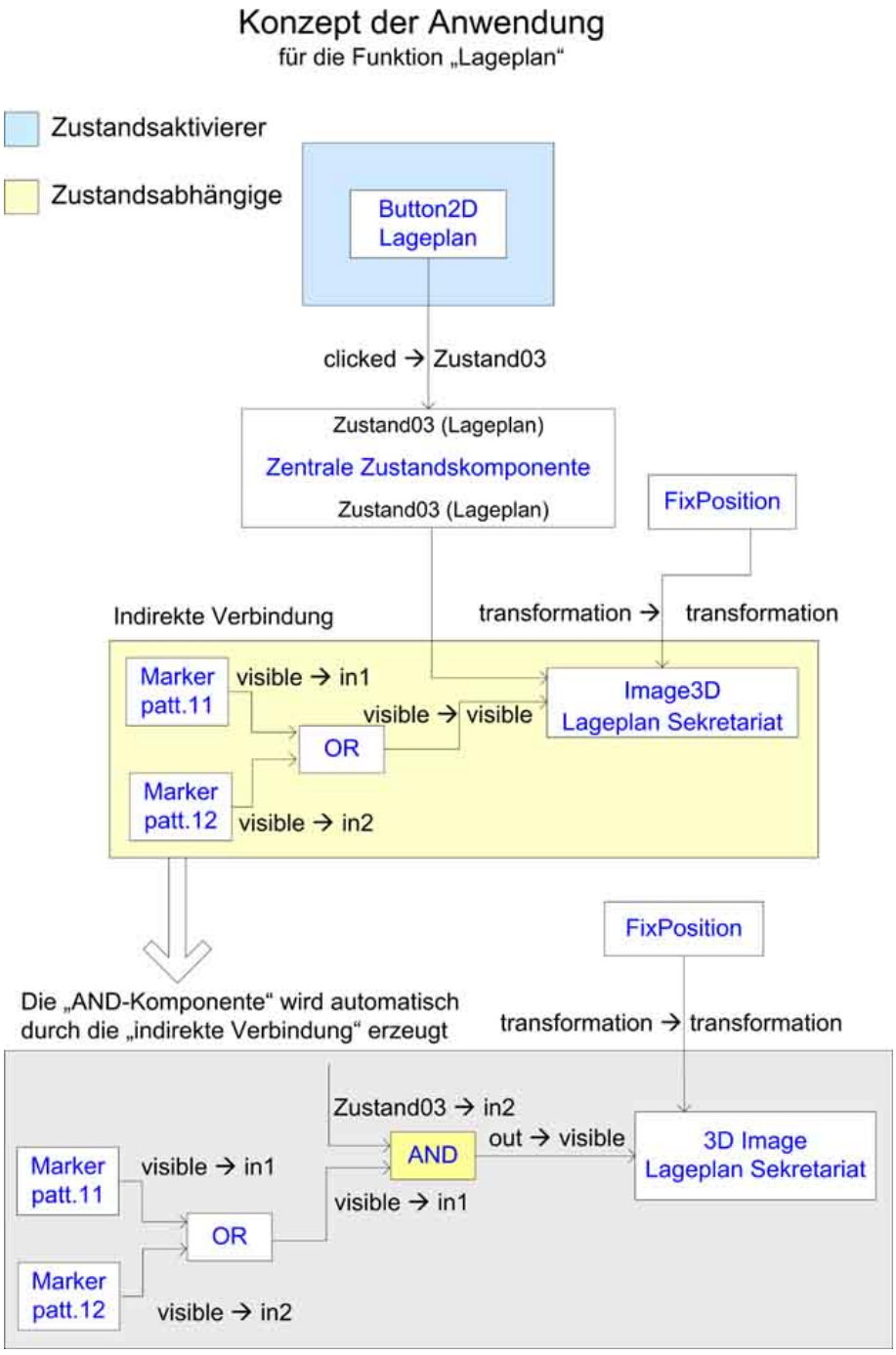


Abb. 6.13: Planung der Anwendungslogik für die Funktion Lageplan

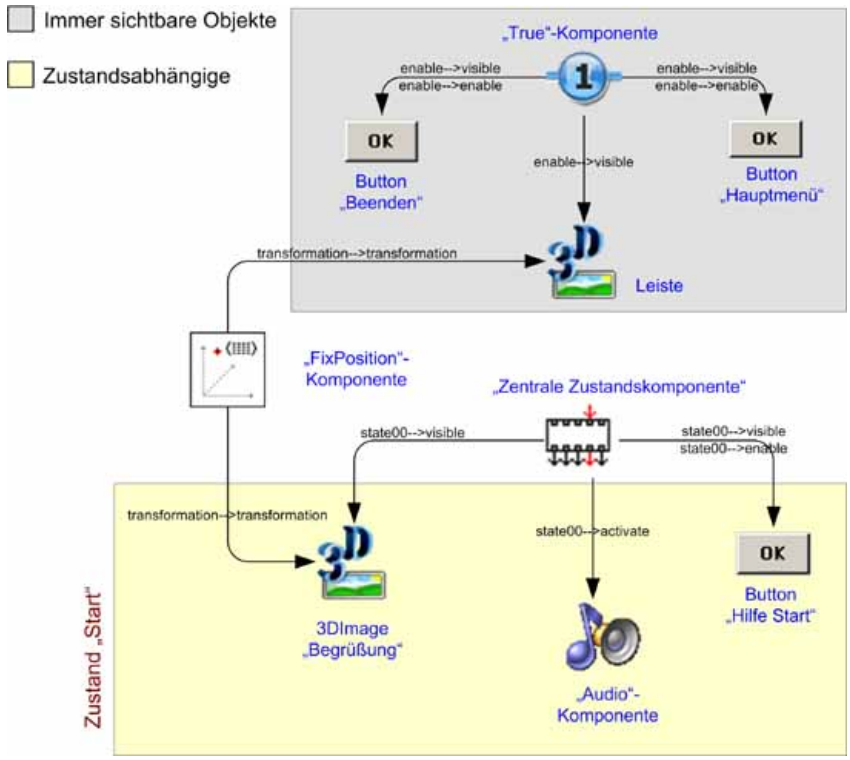


Abb. 6.14: Microsoft Visio-Diagramm des Start-Zustandes

Diese zwei Zustände sind gute Referenzen und repräsentativ für alle vorkommenden Zustände der „MR-Referenzanwendung".

Nun folgt eine kurze Erläuterung des in Abbildung 6.14 angezeigten Diagramms, das den „Start-Zustand" visualisiert. Die Hintergrundfarben grau und gelb sind lediglich dafür da, die immer sichtbaren Objekte und die „Zustandsabhängige" einfacher zu erkennen und wurden extra für dieses Dokument in die Abbildung eingefügt. Sie sind kein Bestandteil der Anwendung und besitzen daher keine technischen Funktionen.

Um den Navigationszustand graphisch darzustellen, müssen die verschiedenen dazugehörigen Zustände, wie sie im Unterkapitel 4.4.2 als Kategorien von „Navigation" spezifiziert wurden, einzeln aufgezeichnet werden. Hier werden jedoch nur die Zustände „Räume" und „Raum 208" schematisch dargestellt.

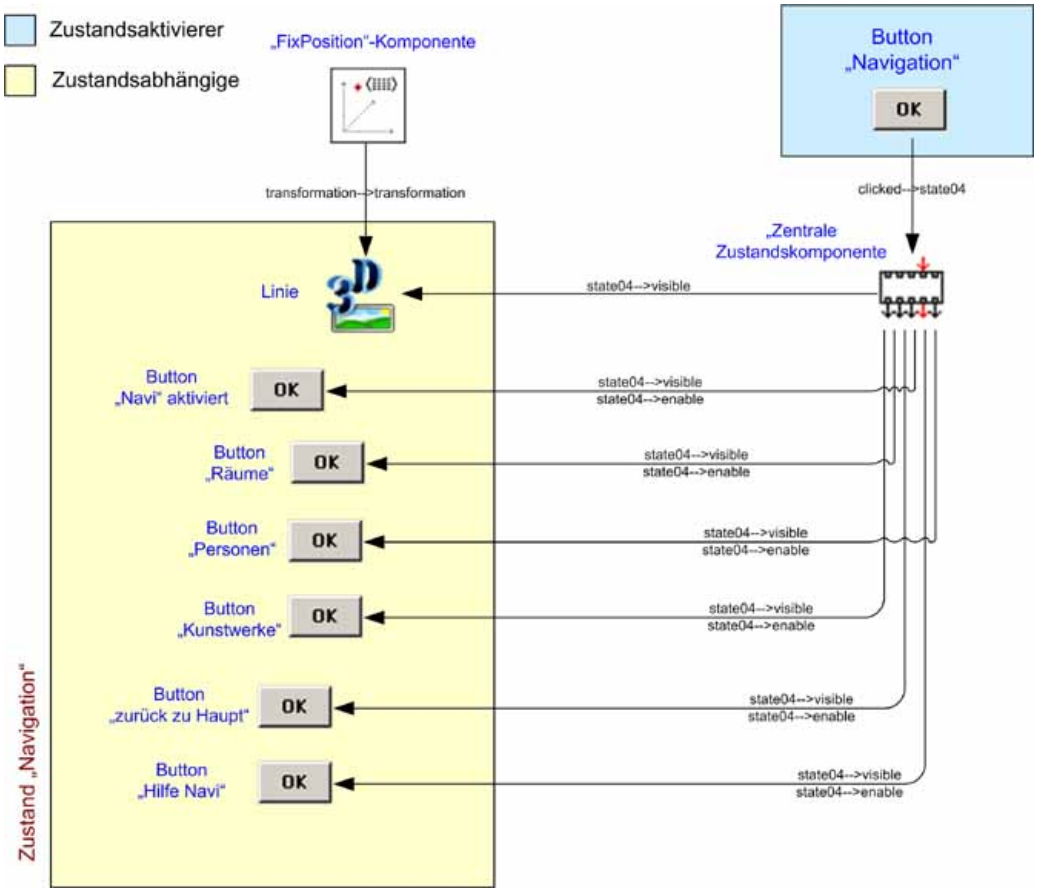


Abb. 6.15: Microsoft Visio-Schema zum Navigationszustand

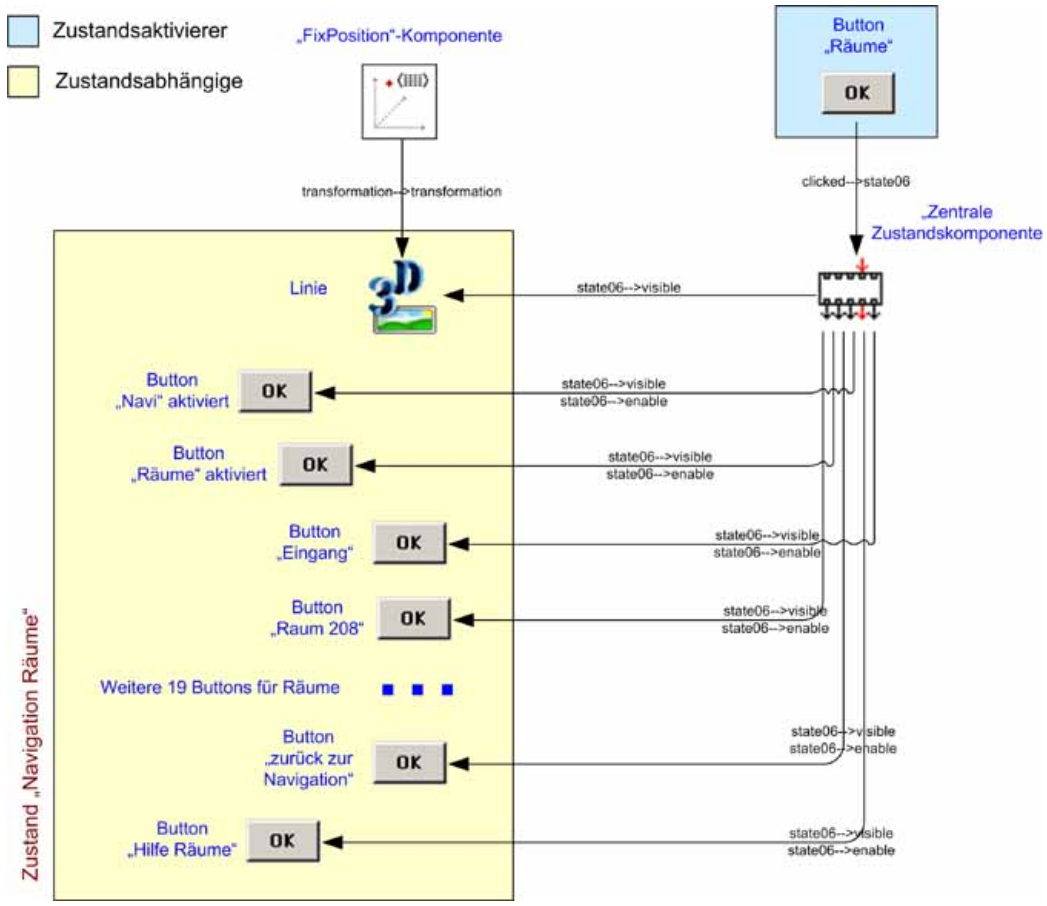


Abb. 6.16: Microsoft Visio-Diagramm des Räume-Zustandes

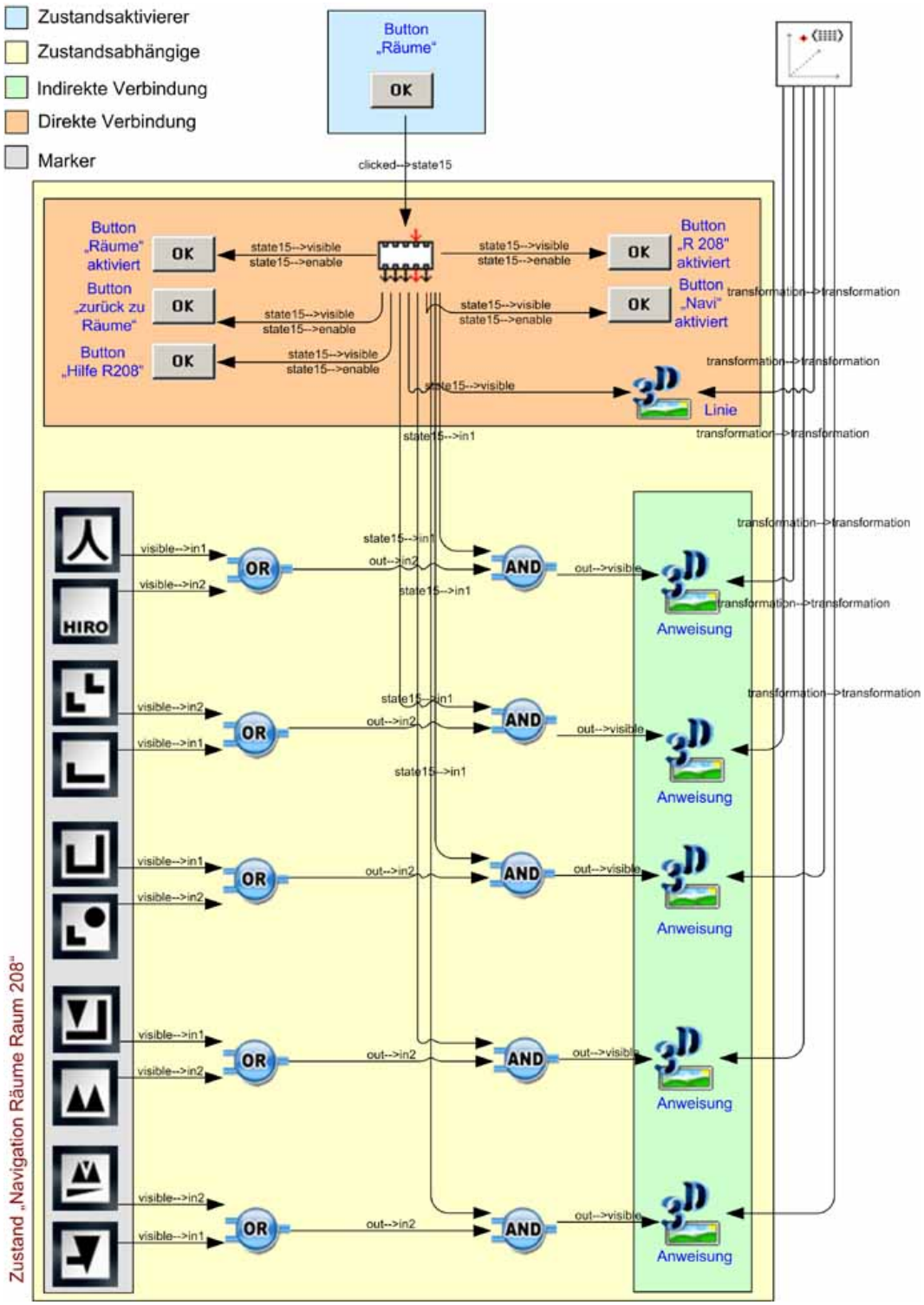


Abb. 6.17: Microsoft Visio-Diagramm des Raum 208-Zustands

## 6.4 Testen

Die Überprüfung des Ein-/Ausgabeverhaltens eines Programms anhand der Spezifikation und der Vergleich des beobachtbaren Verhaltens mit dem erwarteten Verhalten des Programms wird als Test bezeichnet. Die Notwendigkeit des Tests begründet sich darin, dass zu jedem Zeitpunkt der Entwicklung sichergestellt werden muss, dass das System sowohl der Spezifikation als auch den Vorstellungen des Kunden entspricht [Dro03]. Für die „MR-Referenzanwendung“ bedeutet dies, dass der Autor sich bei der Umsetzung an die Spezifikation aus Kapitel 4 im Detail halten muss.

AMIRE bietet die Möglichkeit an, während des Erstellens der „MR-Referenzanwendung“ das Ergebnis zu testen, indem die Anwendung stets im „Camera View-Fenster“ angezeigt wird. Das hilft dem Autor zu jedem Zeitpunkt, den Fortschritt der Produktion mit den Spezifikationen zu vergleichen.

Aus psychologischen Aspekten, die Drobnik [Dro03] erwähnt, sollen Tests von einer fremden Person abgewickelt werden, weil unter Umständen der Autor daran interessiert sein könnte, keine Fehler zu finden. Aus diesem Grund ist es sinnvoll, die Tests der „MR-Referenzanwendung“ in drei Phasen durchzuführen. Für alle drei Phasen gilt, dass Fehler nach Korrektur erneut getestet werden müssen.

- Die erste Phase namens „Entwicklungstests“, findet parallel zur Entwicklung der „MR-Referenzanwendung“ statt. Diese Tests werden auf der Basis der Videos vom Autor durchgeführt, die während der Umsetzung im Unterkapitel 6.2 zum Einsatz kamen. Wenn diese Tests fehlschlagen, dann ist der Grund meistens die aktuell bearbeitete Änderung in der „MR-Referenzanwendung“. Das vereinfacht die Ursachenfindung für einen solchen Fehlschlag. Falls der Grund nicht die zuletzt bearbeitete Änderung ist, dann muss die Ursache systematisch gesucht werden. Das systematische Vorgehen bei der Ursachenfindung wird weiter unten in diesem Abschnitt erläutert. Abbildung 6.18 zeigt den Test der „Lageplan-Funktion“ während der Entwicklung der „MR-Referenzanwendung“ im „Camera View-Fenster“.
- Die zweite Phase namens „vor Ort-Tests“ findet nach Fertigstellung einzelner Hauptfunktionen, wie „Lageplan“, „Navigation“ und „Objekt Info“ statt. Diese Tests stellen sicher, dass die „MR-Referenzanwendung“, die auf Basis von Videos entwickelt wurde, auch in der realen Umgebung des Einsatzgebietes funktioniert. Wenn Fehler vor Ort auftauchen, dann muss der Fehler und die Umstände, die dazu geführt haben, protokolliert und Vermutungen zur Ursache aufgenommen werden. Dies erfolgt, um die Fehler im Nachhinein durch systematische Ursachenfindung zu identifizieren und beheben zu können. Bei den „vor Ort-Tests“ ist es wichtig, dass sie sowohl vom Autor, als auch von einer fremden Person durchgeführt werden können. Abbildung 6.19 zeigt die Autorin der „MR-Referenzanwendung“ beim „vor Ort-Tests“.

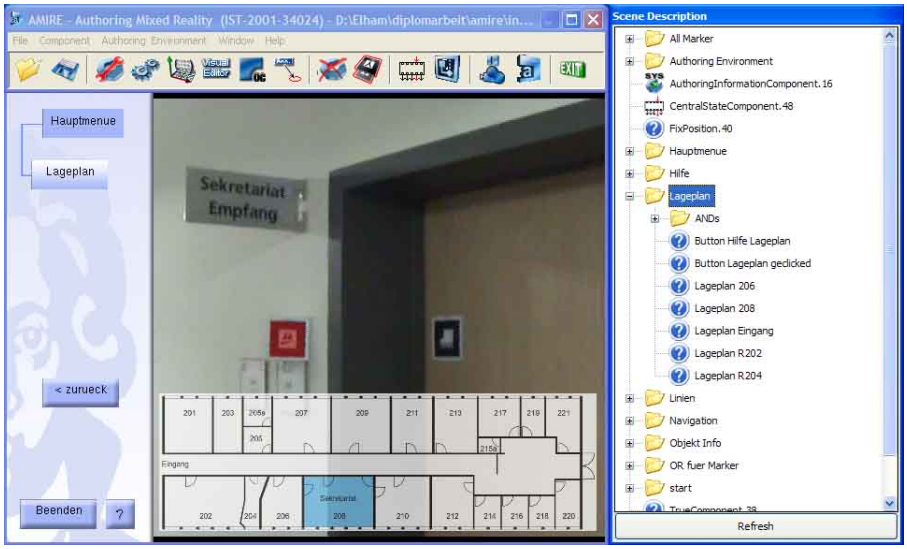


Abb. 6.18: Screenshot des Lageplans am Sekretariat

Abb. 6.19: Eine Anwenderin beim Test der MR-Referenzanwendung

- Die dritte Phase, hier „finaler Test“ genannt, findet nach Abschluss der Entwicklung statt. Die Voraussetzung für diesen Test ist, dass alle anderen Tests erfolgreich abgewickelt wurden. Dieser Test findet vor Ort statt und sollte von einer fremden Person druchgeführt werden, um dadurch die Fehler am besten zu erkennen. Beim Auftauchen von Fehlern sollte analog zu den „vor Ort-Tests“ vorgegangen werden, um eine möglichst effektive Ursachenfindung durchzuführen.

**Systematische Vorgehensweise zur Ursachenfindung:**

Die systematische Vorgehensweise bei der Ursachenfindung von Fehlern in der „MR-Referenzanwendung“ enthält folgende Schritte, die nicht notwendigerweise in dieser Reihenfolge durchgeführt werden müssen. Zudem sind diese nur Empfehlungen, die durch Erfahrungen des Autors zustande kamen. Daher werden nur die wichtigsten Schritte bei der Fehlersuche hier aufgelistet und erklärt.

- „Zustandsabhängige“ und „Zustandsaktivierer“ beim Auftreten eines Fehlers kontrollieren. Im Zweifelsfall sollten ihre Verbindungen getrennt und erneut erstellt werden.
- Falls neue Komponenten, kurz vor dem Auftreten eines Fehlers, eingesetzt wurden, sollten diese entfernt werden, um die Reaktion zu beobachten.
- Überprüfung von Eigenschaften der Komponenten. Wenn eine Graphik-Datei beispielsweise nicht angezeigt wird, kann der Fehler in einer fehlerhaften Konfiguration der Eigenschaften dieser Komponente liegen.
- Komponenten, die durch indirekte Verbindungen mit einem Zustand der „Zentralen Zustandskomponente“ verbunden werden, dürfen nicht manuell aus der „Scene“ gelöscht werden. Diese müssen durch den Editor, der die Zustandskomponenten verwaltet, wieder entfernt werden. Falls dies nicht berücksichtigt wird, können Fehler entstehen, deren Korrektur sehr aufwendig sein kann.

## 6.5 Ergebnisse

Für die im Unterkapitel 6.3 dokumentierten Funktionen „Start“ und „Navigation“ werden im Folgenden die Ergebnisse in Form von „Screenshots“ dargestellt. Für weitere Funktionen der Anwendung werden „Sreenshots“ im Anhang dieser Arbeit präsentiert.

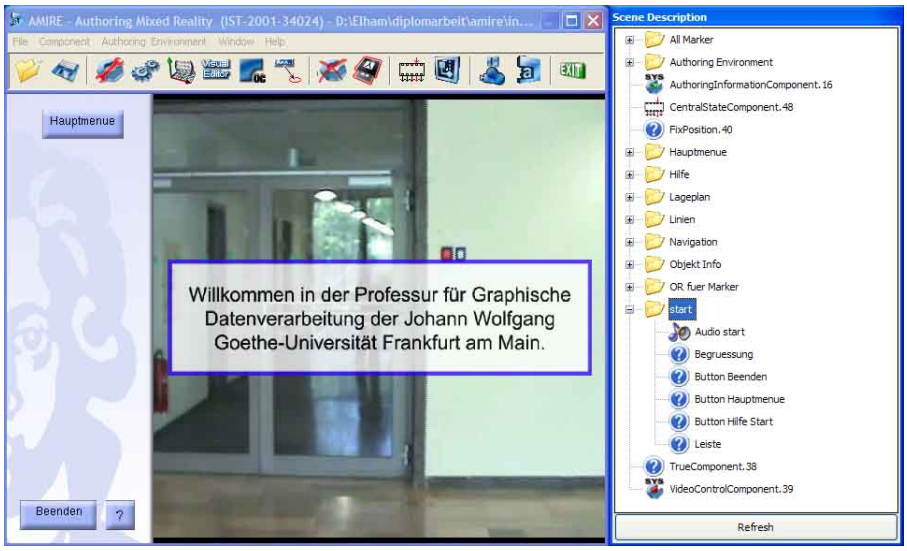


Abb. 6.20: Screenshot des Start-Zustandes

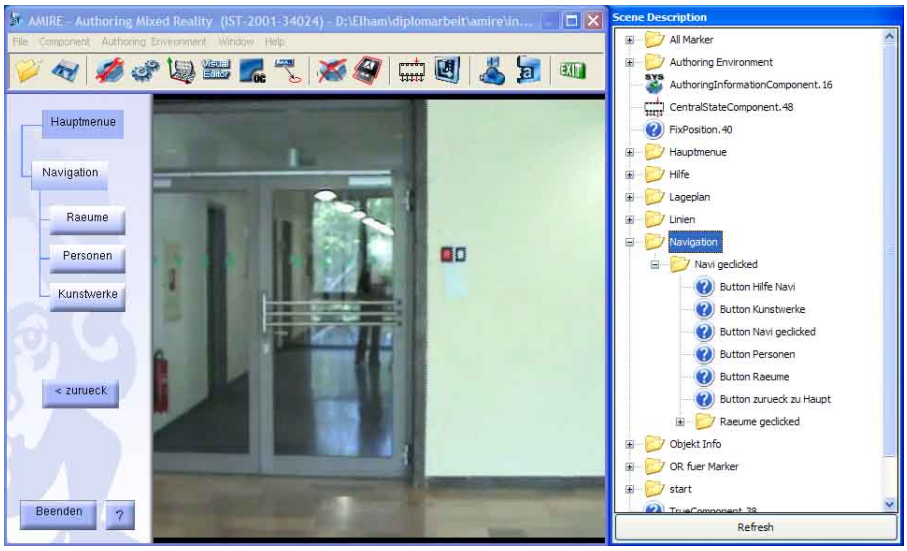


Abb. 6.21: Screenshot des Navigation-Zustandes

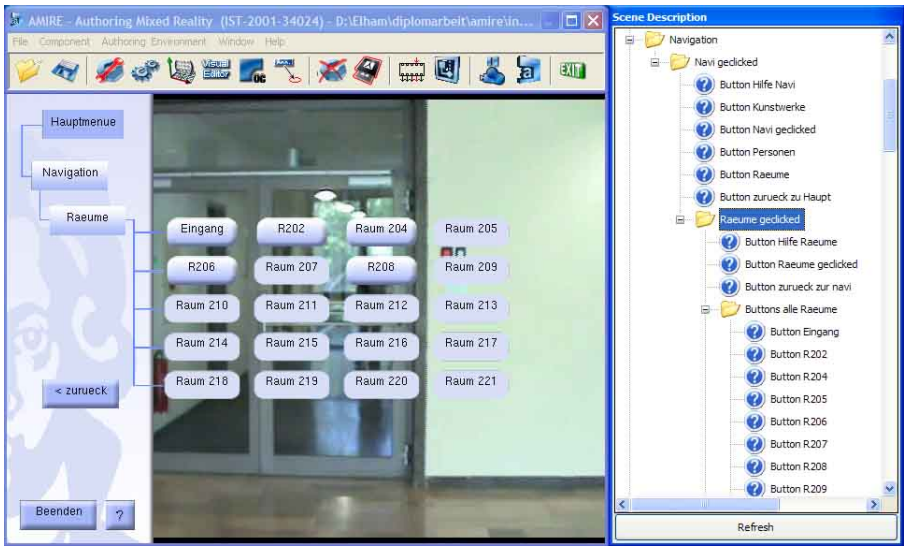


Abb. 6.22: Screenshot des Räume-Zustandes

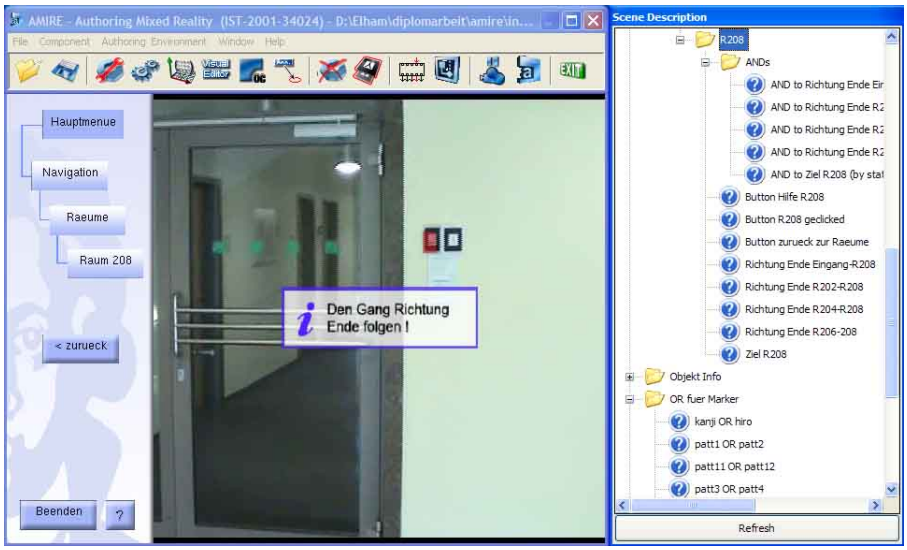


Abb. 6.23: Screenshot des Raum 208-Zustandes am Eingang

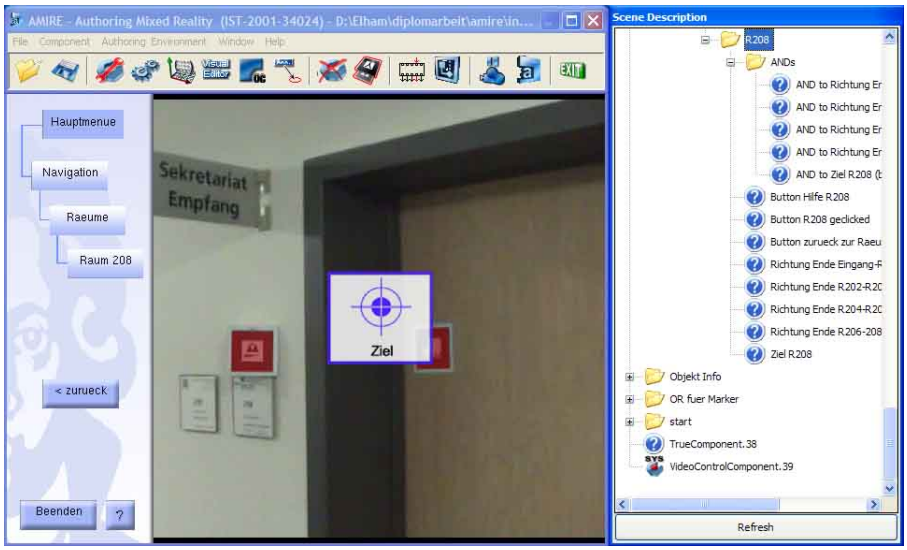


Abb. 6.24: Screenshot des Raum 208-Zustandes am Sekretariat

In diesem Kapitel wurden die wichtigsten Schritte bei der Realisierung der „MR-Referenzanwendung“ vorgestellt. Dabei wurden die AMIRE-Autorenumgebung und ihre Werkzeuge kurz vorgestellt. Danach erfolgte die Erläuterung der Vorgehensweise für die Umsetzung der Anwendung und der gesammelten Erfahrungen bei der Erstellung der „MR-Referenzanwendung“. Anschließend wurde die „MR-Referenzanwendung“ beispielhaft dokumentiert und es erfolgte eine Präsentation der Testphasen und der Ergebnisse für die dokumentierten Beispiele. Im nächsten Kapitel soll eine Evaluierung durchgeführt werden.

# Kapitel 7

# Evaluierung

In diesem Kapitel wird die vorliegende Arbeit hinsichtlich ihrer Zielsetzung kritisch beurteilt. Dabei werden die Ziele den erreichten Ergebnissen gegenübergestellt und bewertet. Durch diese Evaluierung soll einerseits klar gestellt werden, ob die Vorgehensweise in dieser Arbeit korrekt war und inwieweit sie optimiert werden kann und andererseits, ob die Ziele der Arbeit erreicht wurden.

Die Vorgehensweise dieser Arbeit hat sich im Nachhinein hinsichtlich der erreichten Ziele als richtig erwiesen und dazu beigetragen, die definierten Ziele innerhalb des gesetzten Zeitrahmens zu erreichen.

Die Hauptziele dieser Arbeit lagen darin, einerseits zu überprüfen, inwieweit die Erstellung von MR-Anwendungen mit Hilfe von MR-Autorensystemen vereinfacht wird und andererseits, ob MR-Anwendungen analog zur Software-Technik auf Basis von standardisierten Vorgehensmodellen, wie das Wasserfallmodell erstellt werden können. Des Weiteren sollte im Rahmen dieser Arbeit eine repräsentative MR-Anwendung erstellt werden, die bei weiteren Entwicklungen als Vorlage dient und dabei typische Elemente einer MR-Anwendung beinhaltet.

Wenn für die Erstellung von MR-Anwendungen kein Spezialwissen, wie Programmierkenntnisse im Bereich der MR-Technologie benötigt wird, wird der MR-Erstellungsprozess teilweise vereinfacht. Dadurch wird für Autoren, die keine Fachexperten sind, die Erstellung von MR-Anwendungen zugänglich gemacht. Da eines der Hauptziele dieser Arbeit die Überprüfung der tatsächlichen Vereinfachung des MR-Erstellungsprozesses war, wurde dies bei der Umsetzung der „MR-Referenzanwendung“ überprüft. Die „MR-Referenzanwendung“ wurde beispielsweise unabhängig von der einzusetzenden Erstellungsmethode spezifiziert, was dazu führte, dass bei der Spezifikation, ähnlich wie bei der Software-Technik nicht die technischen, sondern die fachlichen Aspekte im Vordergrund standen.

Des Weiteren wurde die „MR-Referenzanwendung“, wie aus Kapitel 6 entnommen werden kann, mit Hilfe von AMIRE durch intuitiv bedienbaren, GUI-basierten Werkzeugen ohne Programmierung erstellt. Diese zwei Indizien bestätigen im Rah-

men dieser Arbeit, dass für die Erstellung der „MR-Referenzanwendung“ kein bis kaum Spezialwissen über MR-Technologie benötigt wurde. Das kann eindeutig als eine Vereinfachung des MR-Erstellungsprozesses angesehen werden.

Eine weitere Vereinfachung des MR-Erstellungsprozesses ergab sich durch die klare Abgrenzung der Aufgabenbereiche mitwirkender Autoren. Die „MR-Referenzanwendung“, die bei dieser Arbeit erstellt wurde, ist zwar von nur einer Autorin erstellt worden, aber die Erstellung wurde in verschiedenen Rollen, die im Unterkapitel 5.2 ausführlich beschrieben sind, durchgeführt. Dabei ist es gelungen, eine klare Abgrenzung der Aufgaben zu erkennen. Die Autorin konnte bei der Übernahme der verschiedenen Autorenrollen keine Reibungspunkte zwischen ihnen feststellen. Dies ist eine erhebliche Vereinfachung, die nicht nur die Kommunikation zwischen den mitwirkenden Autoren, sondern auch den MR-Erstellungsprozess beeinträchtigt.

Ferner wurde im Rahmen dieser Arbeit das AMIRE-Autorensystem eingesetzt, das viele Technologien bei der Erstellung von MR-Anwendungen unterstützt. Diese Technologien sind jedoch in einem überschaubaren Rahmen gehalten, was Autoren bei der Auswahl ihrer multimedialen Objekte dabei hilft, diese in ihre Anwendungen zu integrieren. Dadurch fällt es Autoren einfacher, eine Übersicht über den Umfang der gesamten Entwicklung zu behalten, was wiederum die Vereinfachung des MR-Erstellungsprozesses bewirken kann.

Zusammenfassend kann man feststellen, dass diese Fakten gemeinsam eine gravierende Vereinfachung des MR-Erstellungsprozesses bewirken. Dieser Prozess wird dadurch insoweit vereinfacht, dass MR-Anwendungen auch ohne Programmierung erstellt werden können und dass mitwirkenden Autoren klar definierte Aufgabenbereiche zugeordnet werden können. Außerdem kann der Umfang einzusetzender Technologien zur Erstellung von MR-Anwendungen übersichtlicher gestaltet werden.

Ein weiteres Ziel der Arbeit war die Erstellung einer MR-Anwendung basierend auf einem standardisierten Vorgehensmodell aus der Software-Technik. Da diese MR-Anwendung in den vier Schritten Anforderungsanalyse, Spezifikation, Realisierung und Test erfolgreich umgesetzt wurde, war es nahe liegend, sich an das Wasserfallmodell [Dro03] zu orientieren. Wie aus den vorigen Kapiteln entnommen werden kann, ist diese Realisierung in den oben genannten Schritten tatsächlich gelungen, was eine Bestätigung für die erfolgreiche Übertragung des Wasserfallmodells auf eine typische MR-Anwendung angesehen werden kann.

Weiterhin wurde versucht, die „MR-Referenzanwendung“ als eine repräsentative MR-Anwendung zu erstellen. Bei ihrer Umsetzung wurden Elemente, wie Benutzerführung und Orientierungshilfe eingesetzt, die typischerweise in den meisten MR-Anwendungen vorkommen. Bei der Vorgehensweise wurde die Entwicklung aus der Sicht eines Benutzers, der keine Erfahrungen mit MR-Technologie hat, betrachtet. Die Realisierung

erfolgte aus dieser Perspektive, weil die meisten Autoren sich für diese Vorgehensweise entscheiden. Die „MR-Referenzanwendung“ wurde, wie bereits erwähnt, ähnlich wie bei der Software-Technik umgesetzt, was zur Folge hatte, dass sie unabhängig von Implementierungsmethoden konzipiert wurde. Diese Tatsachen legitimieren den Anspruch, dass diese „MR-Referenzanwendung“ repräsentativ ist und für weitere Entwicklungen durchaus eingesetzt werden kann.

Ganzheitlich betrachtet wurden alle Ziele dieser Arbeit mit positiven Ergebnissen erreicht, was diese Arbeit zu einer Referenz für den erfolgreichen Einsatz der MR-Technologie im Alltag auszeichnet.

# Kapitel 8

# Zusammenfassung und Ausblick

In diesem Kapitel werden die wichtigsten Ergebnisse dieser Arbeit kurz erläutert und die Erkenntnisse zusammengefasst. Zum Schluss wird ein Ausblick auf weiteren Forschungen im Bereich dieser Diplomarbeit geworfen.

## 8.1 Zusammenfassung

Das größte Problem bei der Erstellung von MR-Anwendungen besteht darin, dass sie meistens durch Programmierung erstellt werden. Daher muss ein Autor spezielles Fachwissen über MR-Technologie und zumindest allgemeine Programmierkenntnisse mitbringen, um eine MR-Anwendung erstellen zu können. Dieser Erstellungsprozess soll mit Hilfe von MR-Autorensystemen, die derzeit auf dem Markt existieren und in der Forschung entwickelt werden, vereinfacht werden. Dies war ein Grund, warum diese Arbeit sich zum Ziel erklärte, zu überprüfen, inwieweit die Erstellung von MR-Anwendungen durch Einsatz von MR-Autorensystemen vereinfacht wird.

Ein weiteres Hauptziel war die Erstellung einer repräsentativen MR-Anwendung, die in dieser Arbeit als „MR-Referenzanwendung“ bezeichnet wird. Sie sollte vor allem bei weiteren Entwicklungen als Vorlage dienen können und auf Basis von standardisierten Vorgehensmodellen, wie das Wasserfallmodell, erstellt werden. Ganz wichtig war es noch im Rahmen dieser Arbeit zu bestätigen, dass standardisierte Vorgehensmodelle auf MR-Anwendungen übertragbar sind. Um diese Ziele zu erreichen, sind in dieser Arbeit viele Schritte befolgt worden, die jeweils als Teilziele betrachtet werden können. Die „MR-Referenzanwendung“, die im Rahmen dieser Arbeit erstellt wurde, sollte mit Hilfe eines MR-Autorensystems umgesetzt werden. Um das richtige MR-Autorensystem dafür auszusuchen, wurden im Rahmen einer Analyse fakultative und obligatorische Anforderungen an MR-Autorensysteme definiert, worin auch Funktionen identifiziert wurden, die ein solches System bereitstellen sollte. Das Anbieten einer Vorschau ist ein Beispiel für diese Funktionen, die bei der Erstellung von MR-Anwendungen eine essentielle Rolle spielen können. Die obligatorischen Anforderungen sind welche, die jedes Softwaresystem erfüllen soll, während die fakultativen das Ziel der Verbesserung von Autorensystemen verfolgen. Mit Hilfe der Analyse wurde ein Vergleich zwischen

bekannten MR-Autorensystemen gezogen, dessen Ergebnis AMIRE als ein für die Ziele dieser Arbeit geeignetes MR-Autorensystem identifizierte.

Für die „MR-Referenzanwendung“, die ähnliche Funktionen aufweisen sollte wie andere typische MR-Anwendungen wurden Funktionen, Anwendungsfälle und Design der Oberfläche spezifiziert. Diese Spezifikation wurde unabhängig von dem ausgesuchten Autorensystem durchgeführt, um darin analog zur Software-Technik das Augenmerk auf fachliche und nicht auf technische Aspekte zu legen.

Um ans Ziel zu gelangen, wurde die „MR-Referenzanwendung“ durch AMIRE realisiert, jedoch musste zuvor ihre Spezifikation auf dieses MR-Autorensystem überführt werden. Bei der Überführung wurde die Realisierung aus technischer Sicht betrachtet, das heißt es wurden verschiedene Vorbereitungen, wie die Auswahl der benötigten Komponenten, die Planung der Anwendungslogik und die Aufteilung der Anwendung in verschiedenen Zuständen, durchgeführt.

Nach der gelungenen Realisierung und beispielhaften Dokumentation der „MR-Referenzanwendung“ konnte die Arbeit bewertet werden, worin die erzielten Resultate den Zielen der Arbeit gegenübergestellt wurden. Die Ergebnisse bestätigen, dass mit AMIRE die Entwicklung einer MR-Anwendung ohne Spezialwissen möglich ist und dass diese Arbeit alle ihrer Ziele innerhalb des festgelegten Zeitrahmens erreicht hat.

## 8.2 Ausblick

Es wurde mit der „MR-Referenzanwendung“ eine Basis geschaffen, die als Ausgangspunkt diverser Weiterentwicklungen dient. Dabei können die Dokumentationen, die Vorgehensweise oder die „MR-Referenzanwendung“ selbst als Arbeitsvorlage für weitere Entwicklungen dienlich sein.

Bei den Dokumentationen handelt es sich um nützliche Visio-Diagramme, die vielseitig eingesetzt, beziehungsweise angepasst werden können. Eine gute Einsatzmöglichkeit dafür ist beispielsweise bei jemanden, der eine Referenzanwendung oder einen Prototypen als Vorlage benutzen möchte, um in Erfahrung zu bringen, wie eine Anwendung in AMIRE überhaupt aufgebaut wird.

Ferner kann die Vorgehensweise unverändert übernommen werden, um eine andere MR-Anwendung zu erstellen. Dabei handelt es sich nicht nur um die groben Schritte der Anforderungsanalyse, Spezifikation, Realisierung und Test, sondern auch um die im Detail ausgeführten Schritte, die unter den groben eingeordnet sind. Ein Beispiel dafür sind die unterschiedlichen Rollen, die ein Autor während der Erstellung von MR-Anwendungen übernehmen muss.

Die „MR-Referenzanwendung“ selbst kann auch als eine Vorlage dienen, sofern eine ähnliche MR-Anwendung zu erstellen ist. Dadurch werden sich die Aufwände minimieren, die bei der Erstellung anfallen, weil sich hierdurch das Prinzip der Wiederverwendbarkeit durchleben lässt. Wenn beispielsweise die Marker und ihre Reihenfolge, wie sie in „MR-Referenzanwendung“ vorkommen, übernommen werden, verringern sich die anfallenden Aufwände um ein vielfaches. In diesem Fall brauchen nur die Graphiken oder Menübuttons an die neue MR-Anwendung angepasst zu werden.

Durch Erkenntnisse aus den Recherchen, die im Rahmen dieser Arbeit gewonnen wurden, kristallisiert sich die Aussage, dass MR-Anwendungen bisweilen immer wieder aufs Neue entwickelt werden müssen. Soweit bekannt, ist in dieser Arbeit erstmalig versucht worden, eine Analogie zwischen Erstellung von MR-Anwendungen und Erstellung von Software im Sinne der Software-Technik zu bilden. Das Gelingen dieser Analogiebildung eröffnet die Betrachtung weiterer Aspekte, die möglicherweise von Software-Technik auf MR-Technologie übertragen werden könnten.

Der Einsatz des Wasserfallmodells bei der Umsetzung der „MR-Referenzanwendung“ hat sich als brauchbar erwiesen, was durch die gelungene Realisierung bestätigt wurde. Daher könnten zukünftige Versionen von MR-Autorensystemen analog zur Entwicklungstools, die auf Software-Technik spezialisiert sind, ähnliche standardisierte Vorgehensweisen wie das Wasserfallmodell unterstützen und die Erstellung von MR-Anwendungen damit für Autoren vereinfachen.

Die Betrachtungen dieser Arbeit zeigen die vielfältigen Möglichkeiten beim Einsatz von AMIRE zur Erstellung von MR-Anwendungen. Die in dieser Arbeit realisierte „MR-Referenzanwendung“ dient als ein Beispiel für größere Anwendungen wie der Einsatz der MR-Technologie in Museen, Ausstellungen, Fabrikgeländen oder anderen Gebieten, in denen Benutzer eine elektronische Navigation oder Orientierung häufig einsetzen können. MR-Technologien können in verschiedensten Gebieten insbesondere für kulturelle, industrielle, medizinische, militärische Bereiche und in der Unterhaltungsbranche verwendet werden. Dieser Einsatz findet derzeit schon statt und zeigt ein enormes Wachstumspotential, da die Forschung in diese Richtung stark vorangetrieben wird. Es ist durchaus denkbar, dass in naher Zukunft die MR-Technologie im privaten Bereich zum Einsatz kommt. Man könnte beispielsweise vor dem Urlaub die Reiseführung aus dem Internet beziehen und eine elektronische Führung im Urlaubsort vorplanen.

# Anhang A

# Quellenverzeichnis


[3dS04] *Discreet, 3d Studio max zur Modellierung, Animation und Rendering*, 2004. `http://www.discreet.com`, letzter Zugriff September 2004. (Zitiert auf Seite 38.)

[ABB+01] Azuma, Ronald, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier und Blair MacIntyre: *Recent Advances in Augmented Reality.* IEEE Computer Graphics and Applications, 21(6):34–47, November/Dezember 2001. `http://www.cs.unc.edu/~azuma/cga2001.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix, 7, 10, 14 und 20.)

[AD04] Abawi, Daniel F. und Ralf Dörner: *Creating Mixed Reality Content: Problems, Concepts and Solutions.* In: *Proceedings of the Computer Graphics International 2004 Conference (CGI 2004)*, Seiten 444–451. IEEE Press, 2004. ISBN: 0-7695-2171-1. (Zitiert auf den Seiten xi, 45, 98, 100, 101 und 102.)

[ADG04] Abawi, Daniel F., Ralf Dörner und Paul Grimm: *Werkzeuge für eine Mixed Reality Autorenumgebung.* In: Gausemeier und Grafe (Herausgeber): *Augmented und Virtual Reality in der Produktentstehung*, Band 149 der Reihe *Verlagsschriftreihe des Heinz Nixdorf Instituts*, Seiten 133–148, 2004. (Zitiert auf den Seiten ix, 1, 6, 7, 8, 37, 45, 58, 59 und 119.)

[ADHZ04] Abawi, Daniel F., Ralf Dörner, Michael Haller und Jürgen Zauner: *Efficient Mixed Reality Application Development.* In: *Proceedings of the 1st Conference on Visual Media Production (CVMP)*, Seiten 289–294. IEE, 2004. ISBN: 0-86341-391-9. (Zitiert auf Seite 58.)

[Ado04] *Offizielle Adobe Photoshop CS*, 2004. `http://www.adobe.com/products/photoshop/main.html`, letzter Zugriff September 2004. (Zitiert auf Seite 107.)

[alV04] *Offizielle alVRed Webseite*, 2004. `http://www.lmr.khm.de/alvred/`, letzter Zugriff September 2004. (Zitiert auf Seite 52.)

[AMI04a] AMIRE: *Authoring Framework Technical Reference.* Teil von Deliverable D6.4 (Restricted), Mai 2004. `http://ipx1.labein.es:8084/spa/prjc/amire/DocumentosWeb/publicaciones/Deliverables/restricted/D6.4.zip`, letzter Zugriff September 2004, Zugriff nur mit Benutzername/Passwort. (Zitiert auf den Seiten 97, 113 und 116.)

[AMI04b] AMIRE: *Authoring Tutorial.* Teil von Deliverable D6.4 (Restricted), Mai 2004. `http://ipx1.labein.es:8084/spa/prjc/amire/DocumentosWeb/publicaciones/Deliverables/restricted/D6.4.zip`, letzter Zugriff September 2004, Zugriff nur mit Benutzername/Passwort. (Zitiert auf den Seiten 97 und 116.)

[AMI04c] AMIRE: *Installation and First Steps.* Teil von Deliverable D6.4 (Restricted), Mai 2004. `http://ipx1.labein.es:8084/spa/prjc/amire/DocumentosWeb/publicaciones/Deliverables/restricted/D6.4.zip`, letzter Zugriff September 2004, Zugriff nur mit Benutzername/Passwort. (Zitiert auf den Seiten xi, 97, 112 und 116.)

[AMI04d] *Offizielle AMIRE Webseite*, 2004. `http://www.amire.net`, letzter Zugriff September 2004. (Zitiert auf den Seiten 45 und 58.)

[App04] *Offizielle Apple Macintosh Operating System Webseite*, 2004. `http://www.apple.com/de/macosx/`, letzter Zugriff September 2004. (Zitiert auf Seite 44.)

[ARC02] *Offizielle ARCHEOGUIDE Webseite*, 2002. `http://archeoguide.intranet.gr`, letzter Zugriff September 2004. (Zitiert auf Seite 18.)

[ARD04] Abawi, Daniel F., Silvan Reinhold und Ralf Dörner: *A Toolkit for Authoring Non-linear Storytelling Environments Using Mixed Reality.* In: Göbel, Stefan, Ulrike Spierling und Anja Hoffman et al. (Herausgeber): *Technologies for Interactive Digital Storytelling and Entertainment: Second International Conference (TIDSE 2004), Proceedings*, Band 3105 der Reihe *Lecture Notes in Computer Science*, Seiten 113–118. Springer-Verlag, Heidelberg, Juni 2004. ISBN: 3-540-22283-9. (Zitiert auf den Seiten ix, x, 40 und 59.)

[ARI04] *ARIS Toolset, Das professionelle Werkzeug für E-Business-Engineering*, 2004. `http://www.ids-scheer.de`, letzter Zugriff September 2004. (Zitiert auf Seite 38.)

[ARP04] *Offizielle AR-PDA Webseite*, 2004. `http://www.ar-pda.de/`, letzter Zugriff September 2004. (Zitiert auf Seite 55.)

[ARS04] *Offizielle ARSyS-Tricorder Webseite*, 2004. `http://www.arsys-tricorder.org`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix, 16 und 17.)

[Aut04] *Offizielle Macromedia Authorware Webseite*, 2004. `http://www.macromedia.com/software/authorware/`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix, 21 und 24.)

[Azu97] Azuma, Ronald T.: *A Survey of Augmented Reality.* Presence: Teleoperators and Virtual Environments, 6(4):355–385, August 1997. (Zitiert auf den Seiten ix, 7, 9, 10, 11, 16, 18 und 20.)

[Bal00] Balzert, Helmut: *Lehrbuch der Software-Technik I, Software-Entwicklung.* Spektrum Akademischer Verlag, Heidelberg Berlin, 2 Auflage, 2000. ISBN: 3-8274-0480-0. (Zitiert auf den Seiten 2, 27, 42, 70, 73, 75, 120 und 124.)

[BLM+04] Beckhaus, S., A. Lechner, S. Mostafawy, G. Trogemann und R. Wages: *alVRed – Methods and Tools for Storytelling in Virtual Environments.* Internationale Statustagung „Virtuelle und Erweiterte Realität" Leipzig, Februar 2004. `http://informatiksysteme.pt-it.de/vr-ar-3/projekte/alvred/paper_ALVRED.pdf`, letzter Zugriff Juli 2004. (Zitiert auf den Seiten x und 54.)

[Boe81] Boehm, Barry W.: *Software Engineering Economics.* Pearson International, Prentice Hall, 1981. ISBN:0-1382-2122-7. (Zitiert auf den Seiten ix und 28.)

[Bol95a] Boles, Dietrich: *Elektronisches Publizieren - Autorensysteme und Arbeitsumgebungen für Autoren.* In: *Proceedings Deutscher Dokumentartag 1995, Potsdam, Deutsche Gesellschaft für Dokumentation*, Nummer 5, Seiten 393–411. W. Neubauer, September 1995. `http://www-is.informatik.uni-oldenburg.de/~dibo/paper/ddt95/main.html`, letzter Zugriff September 2004. (Zitiert auf Seite 21.)

[Bol95b] Boles, Dietrich: *Vergleich und Bewertung von Autorensystemen.* Universität Oldenburg, Fachbereich Informatik, Berichte der Arbeitsgruppe Informatik-Systeme, Bericht AIS-23, November 1995. `http://www-is.informatik.uni-oldenburg.de/~dibo/paper/ais/ais.html`, letzter Zugriff September 2004. (Zitiert auf den Seiten 21 und 24.)

[CFG+04] Cheok, Adrian David, Siew Wan Fong, Kok Hwee Goh, Xubo Yang, Wei Liu, Farzam Farzbiz und Yu Li: *Human Pacman: A Mobile Entertainment System with Ubiquitous Computing and Tangible Interaction over a wide outdoor area.* In: *Personal and Ubiquitous Computing*, Band 8 der Reihe *2*, Seiten 71–81. Springer-Verlag, Mai 2004. `http://www.ipo.tue.nl/homepages/mrauterb/Movies/2003-cheok-et-al.pdf`, letzter Zugriff September 2004. (Zitiert auf Seite 19.)

[CS01] Claus, Volker und Andreas Schwill: *Duden Informatik, Ein Fachlexikon für Studium und Praxis*. Meyers Lexikonredaktion, Dudenverlag, Mannheim, 2001. ISBN: 3-411-05233-3. (Zitiert auf Seite 46.)

[DGHP02] Dörner, Ralf, Christian Geiger, Michael Haller und Volker Paelke: *Authoring Mixed Reality - A Component and Framework-Based Approach*. In: *First International Workshop on Entertainment Computing (IWEC)*, Seiten 405–413, Makuhari, Chiba, JAPAN, Mai 2002. `http://webster.fh-hagenberg.at/amire/research/amire@IWEC2002.pdf`, letzter Zugriff September 2004. (Zitiert auf Seite 58.)

[Dro03] Drobnik, Oswald: *Materialien zur Vorlesung Software-Technik SS 2003*, 2003. `http://www.tm.informatik.uni-frankfurt.de:8080/Plone/Lehre/05_SoSe03/mat_swt`, letzter Zugriff September 2004. (Zitiert auf den Seiten 1, 2, 3, 28, 29, 32, 33, 130 und 138.)

[EKFM02] Ebbesmeyer, Peter, Markus Knobel, Jürgen Fründ und Carsten Matyszock: *AR-PDA: Ein digitaler Assistant für VR/AR Inhalte*. In: *Internationale Statustagung „Virtuelle und Erweiterte Realität“ Leipzig*, November 2002. `http://informatiksysteme.pt-it.de/vr-ar-2/projekte/arpda/beitrag_ARPDA.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten 55 und 57.)

[Fai04] *Furniture Assembly Instructor in Mixed Reality (FaiMR) Webseite*, 2004. `http://webster.fh-hagenberg.at/staff/haller/research/projects/FaiMR/index.html`, letzter Zugriff September 2004. (Zitiert auf den Seiten x, 49 und 52.)

[Fis02] Fisher, Scott S.: *An Authoring Toolkit for Mixed Reality Experiences*. International Workshop on Entertainment Computing IWEC, 2002. `http://www.itofisher.com/PEOPLE/sfisher/Authoring_Toolkit_for_Mixed_Reality-IWEC2002.pdf`, letzter Zugriff September 2004. (Zitiert auf Seite 45.)

[Fla04] *Offizielle Macromedia Flash MX 2004 Webseite*, 2004. `http://www.macromedia.com/software/flash/`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix, 21 und 22.)

[FMS93] Feiner, Steven, Blair MacIntyre und Dorée Seligmann: *Knowledge-based Augmented Reality*. In: *Communications of the ACM*, Band 36 der Reihe 7, Seiten 52–62., 1993. (Zitiert auf den Seiten ix, 15 und 16.)

[GBB00] Grässle, Patrick, Henriette Baumann und Philippe Baumann: *UML projektorientiert - Geschäftsprozeßmodellierung, IT-System-*

*Spezifikation und Systemintegration mit der UML*. Galileo Computing, August 2000. ISBN 3-934358-58-6. (Zitiert auf Seite 6.)

[GBZK01] Grechenig, Thomas, Stefan Biffl, Wolfgang Zuser und Monika Köhle: *Software-Engineering mit UML und dem Unified Process*. Pearson Studium, 2001. ISBN: 3-8273-7027-2. (Zitiert auf den Seiten x und 76.)

[GEI04] *Offizielle GEIST Projekt Webseite*, 2004. `http://www.tourgeist.de`, letzter Zugriff September 2004. (Zitiert auf Seite 20.)

[GJM91] Ghezzi, Carlo, Mehdi Jazayeri und Dino Mandrioli: *Fundamentals of Software Engineering*. Prentice-Hall International, Inc., 1991. (Zitiert auf den Seiten 6 und 65.)

[GKR+01] Geiger, Christian, Bernd Kleinjohann, C. Reimann, Dirk Stichling und W. Rosenbach: *Interaktive VR/AR-Inhalte auf mobilen Endgeräten*. In: *Proceedings GI Tagung, Informatik 2001 Workshop 22, Synergien zwischen virtueller Realität und Computerspielen: Anforderungen, Design, Technologien*, Wien, Österreich, September 2001. `http://webster.fh-hagenberg.at/staff/haller/giocg/12_geiger.pdf`, letzter Zugriff Juli 2004. (Zitiert auf den Seiten x, 55 und 58.)

[Gli02] Glinz, Martin: *Skript: Software Engineering I - Grundlagen der Systementwicklung, Kapitel 7: Spezifikation von Anforderungen*, 2002. `http://www.ifi.unizh.ch/groups/req/ftp/se_I/kapitel_07.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten 5, 6 und 27.)

[GTB+04] Goebbels, G., K. Troche, M. Braun, A. Ivanovic, A. Grab, K. von Lübtow, H. F. Zeilhofer, R. Sader, F. Thieringer, K. Albrecht, K. Praxmarer, E. Keeve, N. Hanssen, Z. Krol und F. Hasenbrink: *ARSyS-Tricorder, Development of an Augmented Reality System for intra-operative navigation in maxillo-facial surgery*. BMBF Statustagung, Leipzig, 19.-20. Februar 2004. `http://informatiksysteme.pt-it.de/vr-ar-3/projekte/arsys/paper_ARSYS.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix, 16 und 17.)

[Hal02a] Haller, Michael: *Student projects using ARToolKit*. The First IEEE International Augmented Reality Toolkit Workshop, Darmstadt, September 2002. `http://webster.fh-hagenberg.at/staff/haller/research/publications/2002/fhh-art02.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix und 13.)

[Hal02b] Haller, Michael: *Virtual Reality (Einführung)*, 2002. `http://www.fhs-hagenberg.ac.at/staff/haller/mmp6_2002/04vrIntroduction_1.pdf`, letzter Zugriff September 2004. (Zitiert auf Seite 6.)

[Han04] Hannigan, J. Brendan: *Webseite*, 2004. `http://www.cc.gatech.edu/projects/ael/people/brendanh.html`, letzter Zugriff September 2004. (Zitiert auf Seite 120.)

[Har02] Haringer, Matthias: *Entwurf und Entwicklung eines XML basierten 3D-Präsentations- und Autorensystems*. Diplomarbeit, Fachbereich Technische Informatik, Fachhochschule Ulm, Januar 2002. (Zitiert auf den Seiten ix, x, 46, 47, 48 und 49.)

[Hau02] Hauser, Birgit: *MusicAR - ein Augmented Reality Spiel für Kinder basierend auf ARToolKit und OpenAL*. Diplomarbeit, Fachhochschule Hagenberg, Medientechnik und -design, Österreich, Juli 2002. (Zitiert auf Seite 12.)

[Hel00] Hellbardt, Günter: *Vorlesung Software-Ergonomie*. Institut für Informatik, Friedrich-Schiller-Universität Jena, 2000. `http://www1.informatik.uni-jena.de/Lehre/SoftErg/userbor.htm`, letzter Zugriff September 2004. (Zitiert auf Seite 31.)

[HP04] *Offizielle Human Pacman Webseite*, 2004. `http://mixedreality.nus.edu.sg/research-HP-infor.htm`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix und 19.)

[HR02] Haringer, Matthias und Holger T. Regenbrecht: *A Pragmatic Approach to Augmented Reality Authoring*. In: *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR'02)*, Seiten 237–246. IEEE, Oktober 2002. (Zitiert auf den Seiten 45 und 46.)

[HS04] Holweg, Danie und Oliver Schneider: *Mobile outdoor AR-Informationsystem for historical education with digital Storytelling*, 2004. `http://informatiksysteme.pt-it.de/vr-ar-3/projekte/geist/paper_GEIST.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix und 20.)

[JCJÖ92] Jacobson, Ivar, Magnus Christerson, Patrik Jonsson und Gunnar Övergaard: *Object-Oriented Software Engineering: A Use Case Driven Approach*. Addison-Wesley Professional, 1992. ISBN: 0-201-54435-0. (Zitiert auf Seite 75.)

[Kar98] Karat, Clare-Marie: *Guaranteeing Rights for the User*. In: *Communications of the ACM*, Band 41 der Reihe *12*, Seiten 29–31. ACM Press New York, NY, USA, Dezember 1998. (Zitiert auf Seite 31.)

[KBP00] Kato, Hirokazu, Mark Billinghurst und Ivan Poupyrev: *ARToolKit (version 2.33)*. Hiroshima City University, November 2000.

`http://www.hitl.washington.edu/people/grof/SharedSpace/Download/ARToolKit2.33doc.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix, 12, 13 und 45.)

[Krö04] Krömker, Detlef: *Unterlagen zur Vorlesung: Graphische Datenverarbeitung, Graphikbearbeitung und Rendering, Geometrische Transformationen*, 2004. `http://www.gdv.informatik.uni-frankfurt.de/lehre/ss2004/Folien/GDV/04.Rendering-GeometrischeTransformationen.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten 20 und 103.)

[LW04] *Land Warrior Webseite*, 2004. `http://www.fas.org/man/dod-101/sys/land/land-warrior.htm`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix und 18.)

[May04] *Alias Maya Produkt Webseite*, 2004. `http://www.alias.com/eng/products-services/maya/`, letzter Zugriff September 2004. (Zitiert auf Seite 38.)

[MED04] *Offizielle MEDARPA Webseite*, 2004. `http://www.medarpa.de`, letzter Zugriff September 2004. (Zitiert auf den Seiten 16 und 17.)

[Mic04] *Microsoft Operating Systems Webseite*, 2004. `http://www.microsoft.com`, letzter Zugriff September 2004. (Zitiert auf den Seiten 29, 30, 44 und 69.)

[MK94] Milgram, Paul und Fumio Kishino: *A Taxonomy of Mixed Reality Visual Displays*. IEICE Trans. Information Systems, E77-D(12):1321–1329, 1994. (Zitiert auf den Seiten ix und 7.)

[Mül01] Müller, Stefan: *Skript zur Vorlesung: Online Autorensysteme*. Technische Fachhochschule Berlin, Fachbereich Informatik, 2001. `http://www.tfh-berlin.de/~smueller/download/Online-Autorensysteme.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten 21, 22 und 24.)

[NYC+99] Neumann, Ulrich, Suya You, Youngkwan Cho, Jongweon Lee und Jun Park: *Augmented reality tracking in natural environments*. In: *International Symposium on Mixed Realities*, Tokyo, Japan, 1999. `http://graphics.usc.edu/cgit/pdf/papers/ismr99mac.pdf`, letzter Zugriff September 2004. (Zitiert auf Seite 11.)

[OMG04] *Object Management Group Webseite*, 2004. `http://www.omg.org`, letzter Zugriff September 2004. (Zitiert auf Seite 72.)

[Ope04] *The Open Group, The Unix System Webseite*, 2004. `http://www.unix-systems.org/`, letzter Zugriff September 2004. (Zitiert auf Seite 44.)

[PKFM04] Peter, Ebbesmeyer, Markus Knobel, Jürgen Fründ und Carsten Matyszock: *AR-PDA: Innovative Product Marketing for Innovative Products.* In: *Internationale Statustagung „Virtuelle und Erweiterte Realität“ Leipzig*, 2004. `http://informatiksysteme.pt-it.de/vr-ar-3/projekte/arpda/paper_ARPDA.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten x, 56, 57 und 58.)

[Pow04] *Microsoft PowerPoint Produkt Webseite*, 2004. `http://office.microsoft.com/home/office.aspx?assetid=FX01085797&CTT=6&Origin=EC010963431033`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix, 21, 22, 23 und 46.)

[PTB+01] Poupyrev, Ivan, Desney Tan, Mark Billinghurst, Hirokazu Kato, Holger Regenbrecht und Nobuji Tetsutani: *Tiles: A Mixed Reality Authoring Interface.* In: *Proceedings of INTERACT 2001 Conference on Human Computer Interaction*, 2001. (Zitiert auf Seite 45.)

[RAP04] *RAPTOR-Projekt*, 2004. `http://www.crcg.edu/research/pastprojects/vs.php3`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix und 15.)

[Red04] *RedHat Package Manager*, 2004. `http://www.redhat.com/docs/books/max-rpm/max-rpm-html/index.html`, letzter Zugriff September 2004. (Zitiert auf Seite 44.)

[RKHF02] Riser, Urs, Jürgen Keuneke, Bruni Hoffmann und Hans Freibichler: *Konzeption und Entwicklung interaktiver Lernprogramme, Kompendium und multimedialer Workshop Lernen Interaktiv.* Macromedia GmbH - Akademie für Neue Medien, Springer-Verlag, Berlin Heidelberg, 2002. ISBN: 3-540-67437-3. (Zitiert auf Seite 20.)

[RL01] Raskar, Ramesh und Kok-Lim Low: *Interacting with spatially augmented reality.* In: *Proceedings of the 1st international conference on Computer graphics, virtual reality and visualisation*, Seiten 101–108, Camps Bay, Cape Town, South Africa, 2001. ACM Press. ISBN: 1-58113-446-0. (Zitiert auf den Seiten ix, 10 und 12.)

[RN95] Rekimoto, Jun und Katashi Nagao: *The World Through the Computer: Computer Augmented Interaction with Real World Environments.* In: *Proceedings of the 8th annual ACM symposium on User interface and software technology*, Seiten 29–36, 1995. `http://www.csl.sony.co.jp/person/rekimoto/papers/uist95.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix, 11 und 14.)

[RP02] Rechenberg, Peter und Gustav Pomberger: *Informatik-Handbuch.* Hanser, 3. Auflage, 2002. ISBN: 3-4461-9601-3. (Zitiert auf Seite 20.)

[Sch04] Schmalstieg, Dieter: *Unterlagen zur Vorlesung Virtual Reality, Introduction to Augmented Reality.* Technische Universität Wien, Österreich, 2004. `http://www.ims.tuwien.ac.at/teaching/vr/folien/index.php`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix, 9, 10 und 12.)

[SG84] Schumann, Jörg und Manfred Gerisch: *Softwareentwurf Prinzipien, Methoden, Arbeitsschritte, Rechnerunterstützung.* VEB Verlag Technik, 1984. ISBN: 3-481-35711-7. (Zitiert auf Seite 65.)

[SKT03] Sugano, Natsuki, Hirokazu Kato und Keihachiro Tachibana: *The Effects of Shadow Representation of Virtual Objects in Augmented Reality.* In: *The Second IEEE and ACM International Symposium on Mixed and Augmented Reality*, Seite 76, Tokyo, Japan, Oktober 2003. (Zitiert auf den Seiten ix und 41.)

[Som01] Sommerville, Ian: *Software Engineering.* Pearson Studium, 6. Auflage, 2001. ISBN: 3-8273-7001-9. (Zitiert auf den Seiten ix, 5, 6, 28, 30, 34, 65, 75 und 84.)

[SRSS04] Schnaider, Michael, Sandra Roeddiger, Helmut Seibert und Bernd Schwald: *Implementation and Evaluation of an Augmented Reality System Supporting Minimal Invasive Interventions.* BMBF Statustagung, Leipzig, 19.-20. Februar 2004. `http://informatiksysteme.pt-it.de/vr-ar-3/projekte/medarpa/paper_MEDARPA.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten ix und 17.)

[TBE03] Tesch, Joachim, Oliver Bimber und L. Miguel Encarnação: *RAPTOR: Applying the Virtual Showcase to Augmented Paleontology.* In: *CG topics*, März 2003. `http://www.inigraphics.net/press/topics/2003/issue3/3_03a06.pdf`, letzter Zugriff September 2004. (Zitiert auf Seite 15.)

[Vis04] *Microsoft Visio Produkt Webseite*, 2004. `http://www.microsoft.com/office/visio/prodinfo/default.mspx/`, letzter Zugriff September 2004. (Zitiert auf den Seiten 38 und 119.)

[VKT+01] Vlahakis, Vassilios, John Karigiannis, Manolis Tsotros, Michael Gounaris, Luis Almeida, Didier Stricker, Tim Gleue, Ioannis T. Christou, Renzo Carlucci und Nikos Ioannidis: *Archeoguide: first results of an augmented reality, mobile computing system in cultural heritage sites.* In: *Proceedings of the 2001 conference on Virtual reality, archeology, and cultural heritage*, Seiten 131–140. ACM Press, 2001. ISBN: 1-58113-447-9. (Zitiert auf den Seiten ix und 19.)

[W3C04] *W3C: Extensible Markup Language (XML)*, 2004. `http://www.w3c.org`, letzter Zugriff September 2004. (Zitiert auf Seite 46.)

[WGT+02] Wages, R., B. Grützmacher, G. Trogemann, S. Mostafawy, M. Suttrop, R. Jain, F. Hasenbrink und S. Conrad: *alVRed – Nichtlineare Dramaturgie in VR-Umgebungen.* Internationale Statustagung „Virtuelle und Erweiterte Realität“ Leipzig, November 2002. `http://informatiksysteme.pt-it.de/vr-ar-2/projekte/alvred/beitrag_ALVRED.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten x, 1 und 53.)

[Zel03] Zeller, Andreas: *Skript: Software-Praktikum, Einführung in die Softwaretechnik, Software-Spezifikation*, 2003. `http://www.st.cs.uni-sb.de/edu/sopra/2003/11-spezifikation.pdf`, letzter Zugriff September 2004. (Zitiert auf Seite 65.)

[ZHBH03] Zauner, Jürgen, Michael Haller, Alexander Brandl und Werner Hartmann: *Authoring of a Mixed Reality Assembly Instructor for Hierarchical Structures.* In: *Proceedings of the SIGGRAPH 2003 conference on Sketches & applications*, Seiten 1–1. ACM Press, 2003. `http://webster.fh-hagenberg.at/staff/haller/research/publications/2003/assembling@ismar2003.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten x, 49, 50 und 51.)

[ZHHL03] Zauner, Jürgen, Michael Haller, Werner Hartmann und Thomas Luckeneder: *A generic framework for a training application based on Mixed Reality.* Technischer Bericht, MTD - Upper Austria University of Applied Sciences, Hagenberg, Österreich, 2003. `http://webster.fh-hagenberg.at/staff/haller/research/publications/2003/amireTR2003-1.pdf`, letzter Zugriff September 2004. (Zitiert auf den Seiten xi, 1, 102, 103 und 104.)

# Anhang B

# Abkürzungsverzeichnis

| Abkürzungsverzeichnis | |
|---|---|
| **AMIRE** | Authoring Mixed Reality |
| **AR** | Augmented Reality |
| **ARCHEOGUIDE** | Augmented Reality-based Cultural Heritage On-site GUIDE |
| **AVI** | Audio Video Interleaved, Multimedia Datei-Format |
| **BMP** | Bitmap-Format, Dateiformat für Graphiken |
| **CASE** | Computer Aided Software Entwicklung |
| **DVD** | Digital Versatile Disk |
| **EU** | Europäische Union |
| **GDV** | Graphische Datenverarbeitung |
| **GIF** | Graphics Interchange Format, Dateiformat für Graphiken |
| **GUI** | Graphical User Interface |
| **HMD** | Head Mounted Display |
| **HMS** | Helm Mounted Sights |
| **HUD** | Head Up Display |
| **HWD** | Head Worn Displays |
| **JPEG** | Joint Photographic Experts Group, Komprimierungsstandard für unbewegte Bilder |
| **MP3** | MPEG Audio Layer 3 |
| **MPEG** | Moving Pictures Experts Group, Standard zur Komprimierung von Bild- und Tondaten |
| **MR** | Mixed Reality |
| **MSI** | Microsoft Installer |
| **PC** | Personal Computer |
| **PDA** | Personal Digital Assistant |
| **RPM** | RedHat Package Manager |
| **UML** | Unified Modeling Language |
| **VBA** | Visual Basic for Applications |
| | Fortsetzung folgt ... |

| **Abkürzungsverzeichnis** | |
|---|---|
| **VCD** | Video Compact Disk |
| **VR** | Virtual Reality |
| **XML** | Extensible Markup Language |

Tabelle B.1: Abkürzungsverzeichnis

# Anhang C

# Screenshots der Anwendung

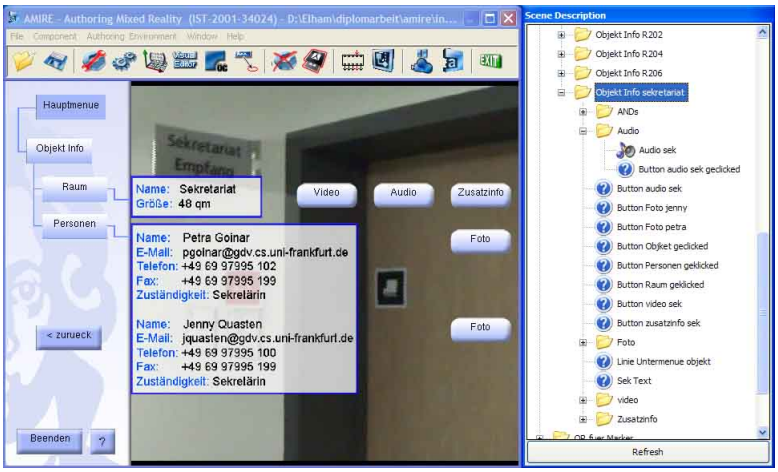


Abb. C.1: Screenshot der Objekt Info am Sekretariat

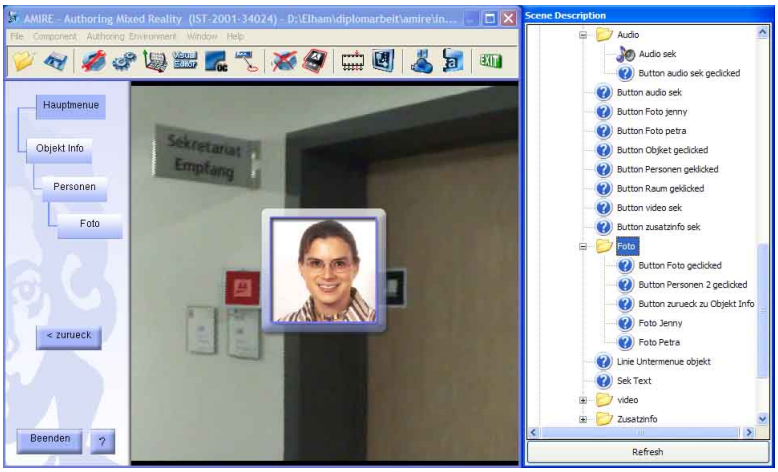


Abb. C.2: Screenshot von Mitarbeiterfotos am Sekretariat

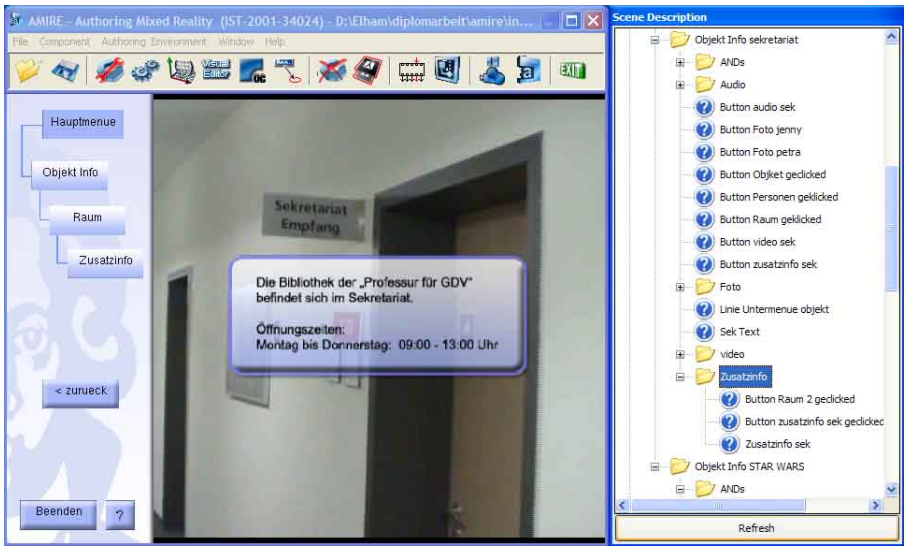


Abb. C.3: Screenshot von Zusatsinformation am Sekretariat

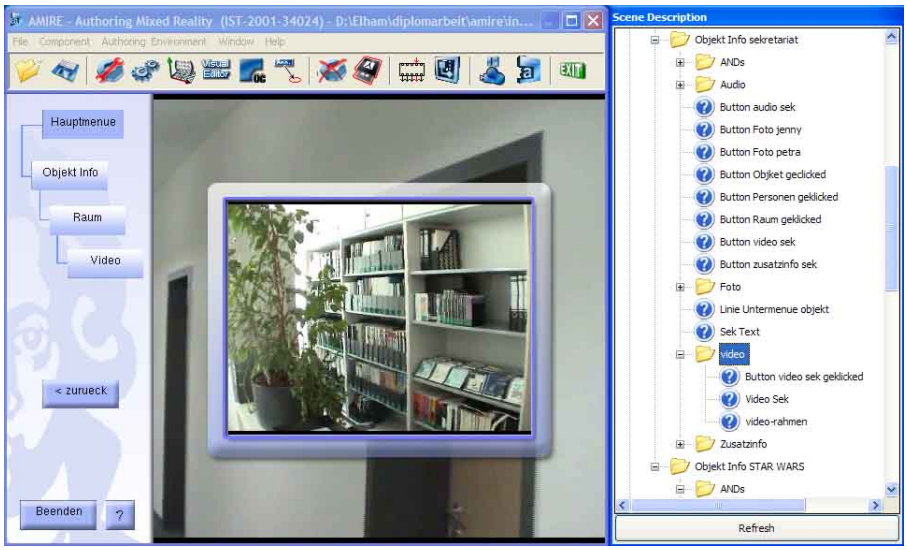


Abb. C.4: Screenshot von Anzeigen eines Videos am Sekretariat

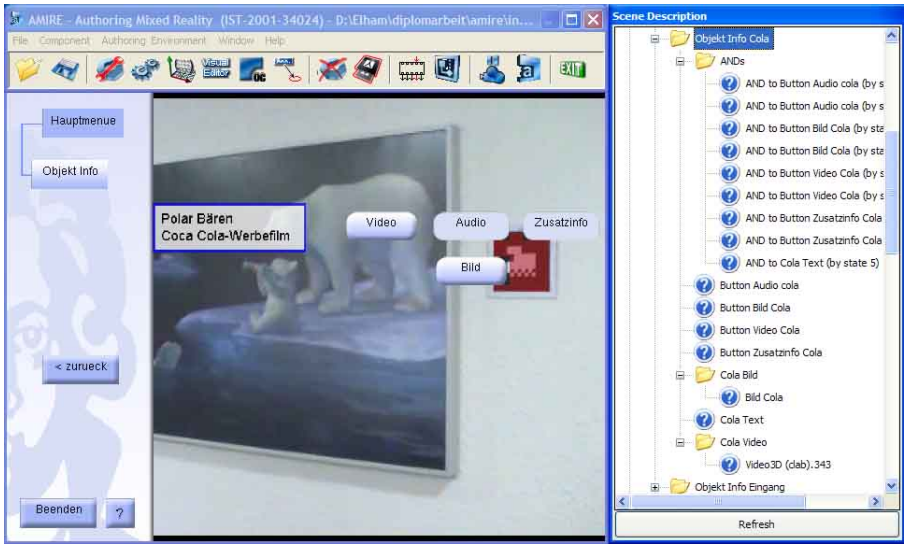


Abb. C.5: Screenshot der Objekt Info beim Polar Bären-Bild

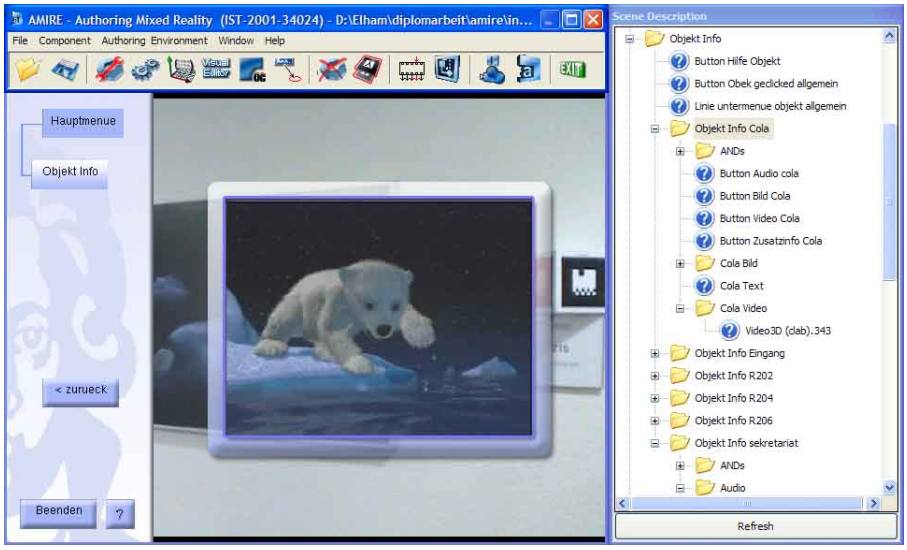


Abb. C.6: Screenshot der Objekt Info beim Anzeigen eines Videos

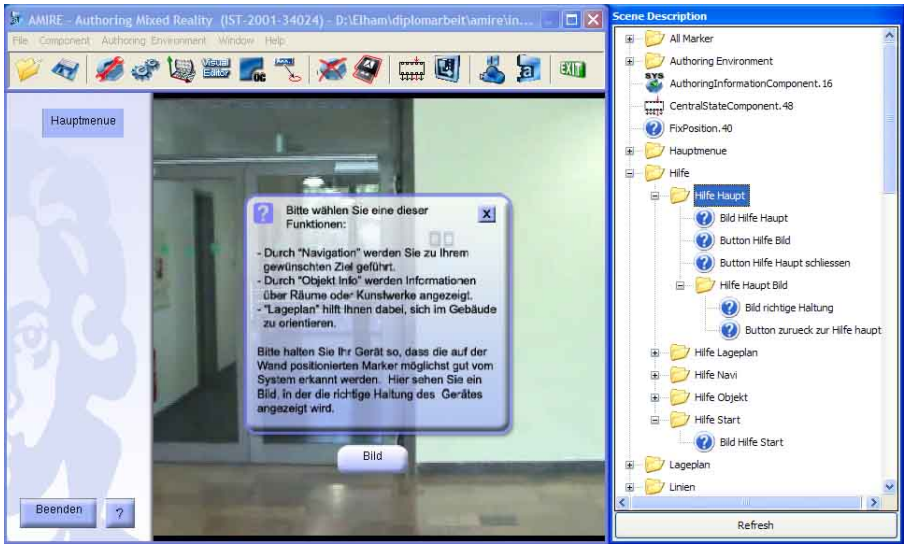


Abb. C.7: Screenshot der Hilfe zum Hauptmenü

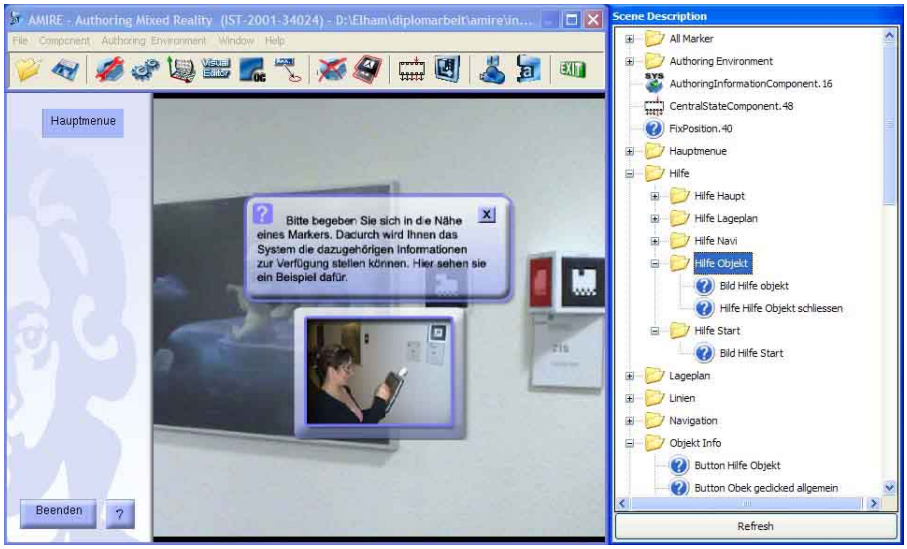


Abb. C.8: Screenshot der Hilfe zur Objekt Info

# Anhang D

# Inhalt der beigefügten DVD

Die beigefügte DVD enthält die Diplomarbeit in Druck- und Webqualität sowie die erstellte „MR-Referenzanwendung“. Die Inhalte der DVD sind in folgende Ordner aufgeteilt:

- **Dokumente als PDF (Portable Dokument Format):** Diplomarbeit in Druck- und Webqualität, Zwischenvortrag und Aufgabenstellung.
- **AMIRE_v1:** Die Version 1 des MR-Autorensystems AMIRE, die „MR-Referenzanwendung“ und alle dafür benötigten Ressourcen wie Graphiken, Audios und Videos.
- **LaTeX:** LaTeX-Quelldateien der Arbeit und alle verwendeten Grafiken.
- **Graphiken:** Alle Graphiken und Screenshots, die mit Hilfe von Microsoft Visio und Adobe Photoshop erstellt wurden.

Zum Starten der „MR-Referenzanwendung“ sollten die folgenden Schritte befolgt werden:

1. Der Ordner *AMIRE_v1* soll auf eine lokale Festplatte kopiert werden.
2. AMIRE kann über die Datei *„amirev1.bat“* im Ordner *AMIRE_v1/bin* gestartet werden.
3. In dem darauffolgend geöffneten Fenster soll die Video-Datei *„Anwendung.avi“* im Ordner *AMIRE_v1/data/videos* geöffnet werden.
4. Nach dem Start von AMIRE, kann nun die Anwendung, durch das Öffnen der XML-Datei *„Anwendung.xml“* im Ordner *AMIRE_v1/data/xml/AGC-Anwendung*, geladen werden.