Bachelorarbeit

Renko Nölken

Evaluierung von Usability-Testmethoden und Heuristiken für Computerspiele

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Renko Nölken

Evaluierung von Usability-Testmethoden und Heuristiken für Computerspiele

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Olaf Zukunft
Zweitgutachter: Prof. Dr. Stefan Sarstedt

Abgegeben am 25. August 2009

Renko Nölken

Thema der Bachelorarbeit

Evaluierung von Usability-Testmethoden und Heuristiken für Computerspiele

Stichworte

Usability, Game Usability, Usability-Heuristiken, Usability-Test, Computerspiele

Kurzzusammenfassung

In dieser Arbeit wird eine Evaluierung von Usability-Testmethoden und Heuristiken für Computerspiele durchgeführt.

Anfangs wird die Planung und Durchführung von Usability-Tests mit unterschiedlichen Computerspielen beschrieben. Auf Basis der Test-Erfahrungen werden die verwendeten Testmethoden beurteilt.

Die Ergebnisse der Tests dienen als Grundlage für die Evaluierung von Usability-Heuristiken für Computerspiele. Diese Evaluierung führt zu der Erstellung einer erweiterten Liste von Heuristiken.

Renko Nölken

Title of the paper

Evaluation of usability test methods and heuristics for computer games

Keywords

Usability, Game Usability, Usability Heuristics, Usability Test, Games

Abstract

In this report the evaluation of usability test methods and heuristics for computer games is described.

Initially the planning and the execution of usability tests with different computer games are described. The applied test methods are evaluated based on the experience gained through the test execution.

The results of the usability tests provide a base for the evaluating of game usability heuristics. At the end, an expanded set of heuristics is created.

Inhalt

1 Einleitung

Usability ist ein wichtiges Qualitätsmerkmal von Software. Praktisch einsetzbare Usability-Untersuchungsmethoden sind daher für die Software-Entwicklung von entscheidender Bedeutung um einen hohen Grad an Usability sicherzustellen. Für den Bereich der klassischen Desktop-Anwendungen, und für Webanwendungen, gibt es inzwischen auch eine Vielzahl an Veröffentlichungen die Untersuchungsmethoden beschreiben und Richtlinien formulieren. Die beschriebenen Untersuchungsmethoden orientieren sich oft an der aufgabenorientierten Gestaltung von Anwendungssoftware, die für Computerspiele nicht üblich ist. Anwendungssoftware soll grundsätzlich dazu in der Lage sein, effizient die benötigten Funktionen zur Erfüllung bestimmter Aufgaben zur Verfügung zu stellen. Computerspiele hingegen werden üblicherweise zu Unterhaltungszwecken erstellt, und nicht um bestimmte funktionale Anforderungen abzudecken. Gerade wenn eine Anwendung unterhalten soll, ist eine einfache Benutzung, also ein hoher Grad an Usability, jedoch von großer Bedeutung.

Im Rahmen dieser Arbeit wird untersucht in welcher Art und Weise weitläufig akzeptierte Usability-Testmethoden für die Untersuchungen von Computerspielen geeignet sind. Hierfür soll anhand von exemplarischen Tests von Computerspielen unterschiedlicher Genres eine Evaluierung von Testmethoden vorgenommen werden. Zusätzlich soll ein Vergleich zwischen den erzielten Ergebnissen mit Heuristiken aus dem Bereich der Game Usability erfolgen. Am Ende sollen Aussagen über die Eignung der verwendeten Verfahren und untersuchten Heuristiken gegeben werden.

In Kapitel 2 werden die notwendigen Grundlagen für ein allgemeines Verständnis dieser Arbeit erläutert. Die behandelten Themengebiete sind Computerspiele, Usability und Usability-Testmethoden, und die Game Usability.

In Kapitel 3, 4 und 5 werden die durchgeführten Usability-Tests näher beschrieben. In Kapitel 3 wird die Planung der durchgeführten Usability-Tests erläutert. Dazu gehören die Auswahl der Spiele, die Auswahl der Testmethoden und der Entwurf eines Testleitfadens. In Kapitel 4 werden die Durchführung der Tests und die Testumgebung beschrieben. Kapitel 5 widmet sich anschließend der Auswertung.

In Kapitel 6 werden die benutzten Usability-Testmethoden auf Basis der Erfahrungen aus den durchgeführten Tests bewertet.

Die Kapitel 7 und 8 widmen sich den Heuristiken für Game Usability. In Kapitel 7 wird eine Evaluierung von veröffentlichten Heuristiken anhand der Erfahrungen und Ergebnisse der durchgeführten Tests vorgenommen. Dies beinhaltet eine Aufbereitung der Testergebnisse, die Auswahl zu evaluierender Heuristiken und die Bewertung der ausgewählten Heuristiken. In Kapitel 8 wird anhand der Ergebnisse aus Kapitel 7 eine modifizierte und erweiterte Liste von Heuristiken formuliert.

2 Grundlagen

In diesem Kapitel werden die Grundlagen für das Vorgehen und das Verständnis dieser Arbeit erläutert. Dazu gehören Computerspiele und ihre Besonderheiten, Usability inklusive Usability-Untersuchungsmethoden, und Besonderheiten der Game Usability.

2.1 Computerspiele

Computerspiele werden genau wie der Großteil an Software und Webanwendungen für einen bestimmten Zweck erstellt. Bei Anwendungssoftware ist dieser Zweck meistens die Unterstützung bei der Ausführung von Arbeitsaufgaben, Software dient also als Werkzeug zur Arbeitserleichterung. Dies kann in gewisser Weise auch auf Computerspiele zutreffen, wenn sie z. B. zu Lern- oder Ausbildungszwecken eingesetzt werden. Meistens ist der Zweck eines Computerspiels jedoch die Unterhaltung des Anwenders.

2.1.1 Genres

Computerspiele lassen sich, genau wie andere Unterhaltungsmedien, in Genres einteilen. Die Einteilung erfolgt dabei meistens anhand des Gameplays und des User Interfaces eines Computerspiels. Oft werden auch technische Faktoren berücksichtigt.

Für Computerspiele gibt es keine wissenschaftlich anerkannte Einteilung, wodurch es oft an subjektiven Faktoren liegt zu welchem Gerne ein Spiel gehört, oder auch welche Genres es überhaupt gibt. Erschwerend kommt hinzu, dass die Entwicklung von Computerspielen stark durch den technischen Fortschritt geprägt ist, und daher immer neue Spielmodelle entstehen. Viele Spiele setzen inzwischen ganz bewusst auf eine Mischung aus Spielelementen, die aus unterschiedlichen Genres bekannt sind, so z. B. *Diablo*, das eine Mischung aus Rollen- und Actionspiel darstellt.

2.1.2 Plattformen

Computerspiele werden für eine Vielzahl unterschiedlicher Plattformen entwickelt. Neben dem PC, gibt es eine Vielzahl von Spielekonsolen (Playstation, Xbox, Wii) und Handheld-Konsolen (Gameboy, N-Gage, Playstation Portable). Die einzelnen Plattformen haben jeweils unterschiedliche Standard-Eingabegeräte. Für den PC wird oft eine Kombination aus Maus und Tastatur verwendet, bei Konsolen wie der Playstation und der Xbox ist die Verwendung eines Gamepads typisch. Der Standard-Controller für die Wii, die Wii Remote, ist eine Fernbedienung mit Bewegungs- und Beschleunigungssensoren. Dadurch kann auf Bewegung und Drehung der Wii Remote reagiert werden. Dies ermöglicht es Steuerungskonzepte umzusetzen, die auf Bewegungen des Spielers reagieren. Die Benutzung eines Spiels wird also auch durch die Wahl der Plattform geprägt.

2.1.3 Struktur und Komponenten von Computerspielen

Die Struktur eines Computerspiels besteht im Wesentlichen aus den Kernmechanismen, dem Gameplay und dem User Interface. Das Zusammenspiel dieser Komponenten wird in Abb. 1 veranschaulicht.

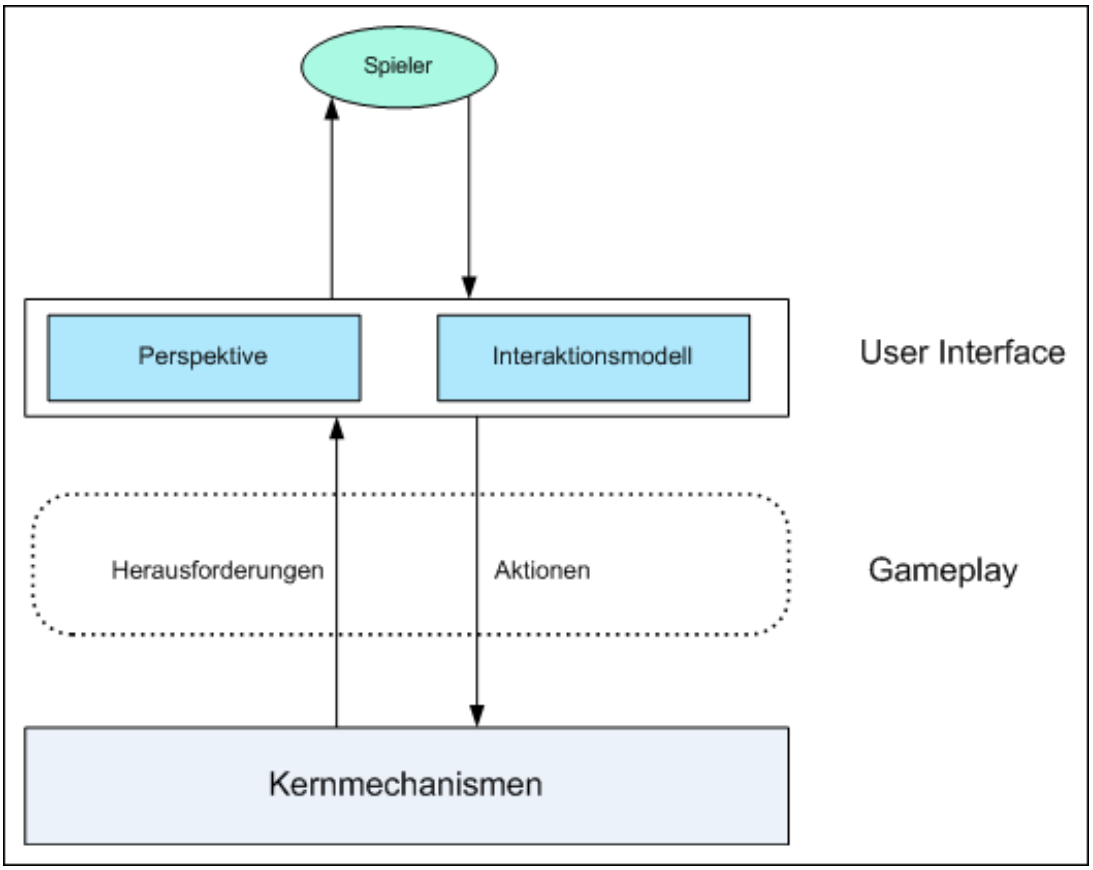

Abb. 1: Komponentenmodell eines Computerspiels [vgl. Adams06]

Im Folgenden werden die einzelnen Komponenten des Modells nach [Adams06] erläutert.

2.1.3.1 Kernmechanismen

Die Kernmechanismen eines Computerspiels sind die in ein symbolisches und mathematisches Modell konvertierten allgemeinen Spielregeln. Dieses Modell ist daher spezifischer als die allgemeinen Spielregeln. Zum Beispiel könnte eine Regel besagen, dass sich Raupen schneller bewegen als Schnecken, während die Kernmechanismen die genauen Bewegungsgeschwindigkeiten der Raupen und Schnecken in Zentimeter pro Minute spezifizieren. Die Umsetzung der Spielregeln hat auch die Generierung des Gameplays zur Folge, da die Kernmechanismen die Herausforderungen des Spiels und mögliche Aktionen des Spielers definieren. [Vgl. Adams06]

2.1.3.2 Gameplay

Gameplay beschreibt die Struktur in der Spieler mit dem Spielsystem und Mitspielern interagieren. Dazu gehören die möglichen Aktionen des Spielers und die Herausforderungen die an den Spieler gestellt werden.

Herausforderungen sind nichttriviale Aufgaben an den Spieler. Zur Lösung muss also geistiger und/oder körperlicher Aufwand betrieben werden. Dies bedeutet aber nicht, dass Herausforderungen schwierig sein müssen. Computerspiele für junge Kinder oder unerfahrene Spieler bestehen bewusst oft aus relativ einfach zu lösenden Herausforderungen. Die Art und Weise der Herausforderungen die in Computerspielen auftreten sind sehr unterschiedlich, wie die folgende Tabelle mit Beispielen zeigt:

Herausforderung	Computerspiel-Beispiel
Körperliche Koordination	
Geschwindigkeit und Reaktionszeit	Tetris, Sonic the Hedgehog
Takt und Rhythmus	Dance Dance Revolution
Erlernen von Bewegungskombinationen	Street Fighter, Tekken, Soul Calibur
Wiedererkennung	
Bewegungs- und Veränderungsmechanismen	Zelda (Verhalten der Gegner, Veränderung der Umgebung)
Statische Muster	Solitaire
Zeitdruck	
Gegen die Uhr	Frogger,Mirror's Edge, Colin McRae Rally
Schneller als andere sein	Need for Speed
Erforschung/Erkundung	
Entdecken versteckter Gegenstände/Levelabschnitte/Belohnungen	Mirror's Edge, Doom, Ultima
Konflikte	
Strategie und Taktik	Warcraft, Command & Conquer
Überleben	Pac-Man
Feindliche Einheiten eliminieren	Half-Life
Beschützen von wertvollen Einheiten oder Gegenständen	ICO
Agieren im Verborgenen	Splinter Cell, Thief
Wirtschaft	
Aufbau eines stabilen Wirtschaftssystems	Anno-Serie, Sim City
Kümmern um Individuen	Die Sims, Creatures
Kreatives	
Ästhetischer Erfolg	The Sims (Einrichtung eines schönen Hauses)

Tabelle 1: Herausforderungen in Computerspielen [Vgl. Adams06]

Aktionen sind in den Spielregeln spezifizierte Handlungsmöglichkeiten des Spielers, die hauptsächlich dazu dienen die Herausforderungen des Spiels zu bewältigen. Diese sind von Spiel zu Spiel sehr unterschiedlich: Bei *Pac-Man* ist es lediglich möglich die Bewegungsrichtung des Avatars zu verändern um vor Gegnern zu flüchten und Gegenstände für bestimmte Boni aufzusammeln. Bei *Grand Theft Auto* ist es hingegen nicht nur möglich sein Avatar zu

steuern, sondern es gibt eine Vielzahl von weiteren, teilweise unkonventionellen, Aktionsmöglichkeiten um die Herausforderungen zu bewältigen. So ist es z. B. möglich Autos zu klauen um schneller an einen Zielort zu kommen. Zusätzlich bieten viele Spiele auch Aktionen die nur der Unterhaltung dienen, bei dem gerade erwähnten *Grand Theft Auto* ist es z. B. möglich das Autoradio einzuschalten.

Die Kombination aus Aktionen und Herausforderungen sollte eine ausgewogene Beziehung darstellen. Interessante Herausforderungen sind nutzlos, wenn keine angemessenen Aktionen angeboten werden um diese zu bewältigen. Gleichzeitig sollte eindeutig erkennbar sein, welche Aktionen für eine Herausforderung bestimmt sind und das die Herausforderungen mit diesen gemeistert werden können [vgl. Adams06]. Die Ausgestaltung dieser Beziehung kann durchaus schwierig sein, da zu mächtige Aktionsmöglichkeiten die Herausforderung trivial erscheinen lassen. Dies wird Spieler schnell langweilen. Gleichzeitig werden zu schwere Herausforderungen den Spieler auf Dauer frustrieren. Des Weiteren ist ein Spiel uninteressant, wenn es zu eingeschränkte Aktionsmöglichkeiten bietet. Um einen hohen Unterhaltungswert zu erzielen, also dem Zweck eines Computerspiels gerecht zu werden, ist das Gameplay daher von entscheidender Bedeutung.

An dieser Stelle wird ein Unterschied von Computerspielen und Desktopanwendungen deutlich: Während Desktopanwendungen als Werkzeug dienen und die möglichen Aktionen daher so effizient wie möglich gestalten, wird versucht bei Computerspielen die Aktionen so uneffektiv wie nötig zu gestalten, um die Herausforderungen nicht zu trivialisieren.

2.1.3.3 User Interface

Ein User Interface dient im Allgemeinen dazu, dem Anwender die Interaktion mit einer Softwareanwendung zu ermöglichen. Der Anwender kann Eingaben machen, z. B. über Tastatur und Maus, und so bestimmte Aktionen ausführen. Anschließend werden die Ergebnisse grafisch, textuell und auditiv präsentiert. Zusätzlich werden oft auch weitere Informationen, z. B. über den Systemstatus, angezeigt. Das User Interface dient also zur Steuerung der Anwendung durch den Benutzer. Übertragen auf Computerspiele bedeutet dies, dass das User Interface als Vermittler zwischen dem Spieler und den Kernmechanismen des Spiels agiert. Ein Teil der Spielwelt wird grafisch dargestellt und mit Sound unterlegt. Dadurch können die Herausforderungen und möglichen Aktionen vom Spieler wahrgenommen werden. Gleichzeitig führen die Eingaben des Spielers, also Tastendrücke auf einem Gamepad oder der Tastatur und Bewegungen mit der Maus oder einem Joystick, zu einer Aktion innerhalb der Spielwelt. Das User Interface besteht also aus einer Perspektive auf die Spielwelt und einem Interaktionsmodell welches Eingaben des Spielers interpretiert um Aktionen in der Spielwelt auszulösen.

Ein **Interaktionsmodell** beschreibt die Beziehung zwischen den Eingaben des Spielers und den daraus resultierenden Aktionen innerhalb der Spielwelt. In der Praxis lassen sich viele

unterschiedliche Typen von Interaktionsmodellen identifizieren von denen fünf im Folgenden beispielhaft erläutert werden:

- **Avatar basierend**: Der Spieler kontrolliert meistens einen einzelnen Charakter („Avatar“) in der Spielwelt. Interaktionen mit der Spielwelt werden durch den Avatar ausgeführt, und Aktionen betreffen auch nur die Region der Spielwelt in der sich der Avatar gerade befindet. Daher wird bei diesem Modell relativ viel Aufwand für die Bewegung und Navigation nötig sein. Des Weiteren kann ein Avatar sehr unterschiedlichen Rollen entsprechen, ein Auto in einem Autorennspiel ist genauso ein Avatar wie ein menschlicher Zauberer in einem Rollenspiel.
- **Omnipräsent**: Dieses Modell erlaubt es dem Spieler mit mehreren, unterschiedlichen Teilen der Spielwelt zur gleichen Zeit zu interagieren. Daher ist eine Perspektive auf die Spielwelt nötig, die dem Spieler Übersicht verschafft. Üblicherweise findet man dieses Interaktionsmodell bei Aufbau-, Strategie- oder Taktik-Spielen. Relativ viel Aufwand geht bei diesem Modell in die Umsetzung der Möglichkeiten Einheiten oder Objekte zu selektieren und zu befehligen.
- **Gruppen basierend**: Der Spieler kontrolliert eine Gruppe von Charakteren. Dieses Modell wird häufig in Rollenspielen genutzt. Eine Schwierigkeit bei diesem Modell ist die Wahl der passenden Perspektive auf die Spielwelt, da die Übersicht über das Spielgeschehen leicht verloren geht.
- **Kandidatenmodell**: Der Spieler beantwortet Fragen wie in einer Spielshow im Fernsehen. Dieses Interaktionsmodell ist im Vergleich zu den anderen sehr simpel, da nur die Eingabe oder Auswahl der möglichen Antworten sichergestellt werden muss.
- **Arbeitsplatzmodell**: In diesem Modell wird ein (Computer-)Arbeitsplatz simuliert. Dies findet sich in verschiedenen Wirtschaftssimulationen und Fußball-Manager-Spielen wieder. Oft können hier alle Aktivitäten mit der Maus angesteuert werden.

[Vgl. Adams06]

Die **Perspektive** eines Computerspiels stellt eine Art virtuelle Kamera dar, durch die es dem Spieler ermöglicht wird die Spielwelt wahrzunehmen. Meistens ist nur ein Teilausschnitt der Spielwelt sichtbar. Im Folgenden werden die Ego-, die Drittperson- und die Vogel-Perspektive vorgestellt.

In der **Ego-Perspektive** entspricht der dargestellte Ausschnitt der Spielwelt einem Blick durch die Augen einer Spielfigur. Daher ist diese Perspektive oft in Kombination mit dem Avatar basiertem Interaktionsmodell anzutreffen. Computerspiele mit Ego-Perspektive sind z. B. die Adventure-Reihe *Myst*, die Rollenspiele-Reihe *Morrowind*, das Plattform-Spiel *Mirror’s Edge* oder Ego-Shooter wie *Half-Life*.

Abb. 2: Ego-Perspektive in *Mirror's Edge*

In der **Drittperson-Perspektive** befindet sich die Kamera außerhalb des Avatars und stellt diesen im sichtbaren Ausschnitt der Spielwelt dar. Die Position der Kamera kann dabei hinter dem Avatar („Verfolgerperspektive“) oder an festen Punkten der Spielwelt fixiert sein. Die Drittperson-Perspektive bietet sich in Kombination mit einem Avatar oder Gruppen basierten Interaktionsmodell an, und wird daher oft für Adventures, Rollen- und Actionspiele verwendet. Die Verfolgerperspektive wird z. B. in der Action-Adventure-Reihe *Tomb Raider* und bei dem Online-Rollenspiel *World of Warcraft* verwendet. Feste Kamerapositionen finden sich häufig in Action-Adventure-Spielen mit Horror-Elementen wie *Resident Evil* und *Silent Hill.*

Abb. 3: Drittperson-Perspektive in *World of Warcraft* (Verfolgerperspektive)

Vogelperspektive: Die Kamera befindet sich typischerweise in einem Winkel schräg über der Spielwelt, um einen großen Teil des Spielgeschehens und viele Charaktere oder Einheiten auf einmal darzustellen. Die Priorität liegt im Gegensatz zu der Ego- oder Drittperson-Perspektive nicht mehr auf einem oder mehreren Avataren, sondern auf der Spielwelt. Die Vogelperspektive bietet sich daher für ein omnipräsentes oder Gruppen basiertes Interaktionsmodell an. Bei Strategie-, Taktik- und Aufbausimulationen ist die Vogelperspektive Standard und auch in Rollenspielen findet sie oft Verwendung. Computerspiele mit Vogelperspektive sind z. B. die Strategiespiele der *Warcraft*-Reihe, die *Anno*-Reihe, oder das Rollenspiel *Baldur's Gate*.

Abb. 4: Vogelperspektive in *Anno 1701*

2.2 Usability und Usability-Testmethoden

Usability beschreibt die Gebrauchstauglichkeit eines Softwareproduktes. Die Norm ISO 9241-11 [ISO9241] bestimmt hierfür drei Leitkriterien:

- Effektivität zur Lösung einer Aufgabe,
- Effizienz der Handhabung des Systems,
- Zufriedenheit der Nutzer einer Software.

[Nielsen93] weist zudem darauf hin, dass es wichtig sei zu verstehen, dass Usability keine eindimensionale Eigenschaft eines User Interfaces darstellt, sondern aus mehreren komplexen Komponenten bestehe. Dazu gehören die Erlernbarkeit, die Effizienz, die Einprägsamkeit und die Fehlerbehandlung des Systems, genau wie die Zufriedenheit des Nutzers.

Um die Usability einer Anwendung zu untersuchen gibt es verschiedene Ansätze. Im Folgenden werden die für diese Arbeit relevanten Methoden erläutert.

2.2.1 Heuristische Evaluierung

Die heuristische Evaluierung ist eine systematische und trotzdem schnell und einfach einzusetzende Methode um mögliche Usability-Probleme in einem User Interface zu finden. Hierbei wird das User Interface betrachtet und analysiert ob bestimmte Richtlinien („Heuristiken“) eingehalten wurden und ob durch Abweichungen Usability-Probleme entstehen können. Die Richtlinien können dabei aus unterschiedlichen Quellen kommen: Es gibt Veröffentlichungen und Literatur die Richtlinien enthalten, Design-Richtlinien für bestimmte Systemtypen („Styleguides“) und auch die eigene Intuition und Vernunft kann als Quelle dienen [Vgl. Nielsen 1993].

Ein Vorteil dieser Methode ist, dass ein lauffähiges System nicht zwingend erforderlich ist. Daher bietet sich die heuristische Evaluierung für Untersuchungen in einem frühen Entwicklungsstadium und in einem iterativen Prozess an, da z. B. auch „Papier-Prototypen“ geprüft werden können. Des Weiteren ist dieses Verfahren sehr flexibel einsetzbar, da durch Austauschen und Modifizieren der verwendeten Heuristiken eine Anpassung an eine spezielle Domäne möglich ist.

Ein Nachteil der heuristischen Evaluierung ist, dass keine tatsächliche Benutzung analysiert wird, sondern lediglich eine regelbasierte Analyse stattfindet. Dies hat zur Folge, dass lediglich potenzielle Usability-Probleme gefunden werden. Das tatsächliche Auftreten von Probleme kann nur in einer Benutzungssituation ermittelt werden.

2.2.2 Usability-Test mit Anwendern

Das Testen der Usability einer Anwendung mit Anwendern oder Personen aus der jeweiligen Zielgruppe ermöglicht es Informationen über die tatsächliche Benutzung des User Interfaces zu bekommen und ist der einzige sichere Weg um garantiert vorhandene Usability-Probleme zu identifizieren.

Die Testpersonen werden aufgefordert typische Aufgaben mit dem Testobjekt zu erledigen. Während der Bearbeitungsphase werden die Testpersonen beobachtet um Probleme in der Bedienung, also Schwachstellen der Usability, aufzudecken. Ein typischer Testablauf besteht dabei aus vier Phasen [Vgl. Nielsen 1993]:

1. **Vorbereitung:** Der Testleiter kontrolliert ob das Testobjekt im Startzustand ist. Ist dies nicht der Fall, wird dieser wieder hergestellt, im Falle einer Webanwendung könnte z. B. das Löschen des Verlaufs und Zwischenspeichers des Browser nötig sein. Des Weiteren wird kontrolliert ob alle benötigten Unterlagen und Materialien, z. B. Aufgabenzettel und Fragebögen für die Testperson oder ein Leitfaden für den Testleiter vorliegen.
2. **Einführung**: In der Einführungsphase werden der Testperson die Testumgebung und der Testablauf erläutert. Wichtig ist hierbei auch auf Faktoren einzugehen, die nicht Teil der Aufgaben sind. Oft ist es sehr hilfreich die Testperson darauf hinzuweisen,

dass es um die Evaluierung der Software und nicht um die Fähigkeiten der Testperson geht, damit die Testperson sich im Laufe des Tests möglich natürlich verhält und keine Angst davor hat Fehler zu machen. Falls eine Aufzeichnung des Tests erfolgt, sollte auch dies erwähnt werden und bei Bedarf eine Rechtserklärung über die Verwendung der Aufzeichnungen bereit liegen. Des Weiteren kann die Einführungsphase genutzt werden, um die Erfahrung und Fähigkeiten des Anwenders hinsichtlich des Testobjekts abzufragen. Dies kann bei der Analyse und Bewertung der auftretenden Auffälligkeiten hilfreich sein.

3. **Test**: Während des eigentlichen Tests sollte der Testleiter sich der Testperson gegenüber zurückhaltend verhalten, damit diese die Aufgaben selbstständig löst. Auch auf Nachfrage sollte auf Hilfestellungen verzichtet werden. Nur wenn die Testperson festhängt oder frustriert ist, sollte eine Ausnahme gemacht werden. Des Weiteren ist es sinnvoll Auffälligkeiten zu notieren, damit diese in der Nachbesprechung angesprochen werden können.
4. **Nachbesprechung**: In der Nachbesprechung kann die Testperson zu ihrer Meinung über das Testobjekt befragt werden. Dies kann in Form eines vorbereiteten Fragebogens oder eines Interviews mit dem Testleiter geschehen. Wenn ein Fragebogen vorbereitet wurde, sollte dieser vor dem Interview ausgefüllt werden, damit die Meinung und Empfindung der Testperson nicht durch die Diskussion über die einzelnen Auffälligkeiten verfälscht wird. Im Rahmen des Interviews kann auf die notierten Auffälligkeiten und Unklarheiten die im Laufe des Tests aufgetreten sind eingegangen werden.

2.2.3 Laut Denken

Laut Denken ist eine Untersuchungsmethode, die oft bei Usability-Tests mit Anwendern eingesetzt wird. Die Testperson wird dabei aufgefordert seine Gedanken und Erwartungen über das Testobjekt auszusprechen. In [Hoonhout08] werden zwei Varianten beschrieben:

1. **Nebenläufig**: Während der Benutzung des Systems.
2. **Retrospektiv**: Im Anschluss an die Benutzung des Systems.

Beide Varianten ermöglichen es einen Eindruck davon zubekommen, wie die Testperson bestimmte Teile des Testobjekts versteht und interpretiert. Falsche Vorstellungen über das Testobjekt können auf diese Weise eindeutig identifiziert werden.

Ein Vorteil des nebenläufigen lauten Denkens ist es, dass die Testperson während sie bestimmte Aktionen ausführt, darüber spricht was für Aktionen und Reaktionen sie erwartet und warum sie bestimmte Aktionen ausführt und andere nicht. Würde man bestimmte Dinge im Nachinterview ansprechen, hätte sie womöglich ihre ursprüngliche Intention schon wieder vergessen. Auch kann es passieren, dass Probleme im Nachhinein nicht mehr als solche gesehen werden, da umständliche Aktionen auch zum Ziel führten.

Des Weiteren stellt das ständige Verbalisieren der Gedanken für die meisten Menschen ein unnatürliches Verhalten dar. Viele Testpersonen fühlen sich dabei unwohl. Grundsätzlich verhalten sich die Probanden beim Einsatz von nebenläufigem Denken oft unnatürlich. Es kann daher zu Problemlösungen führen, die der Testperson in einer natürlichen Umgebung unentdeckt geblieben wären. Ebenfalls kann es passieren, dass die Testperson sich mehr auf die Suche nach Fehlern konzentriert als auf die Arbeit mit dem System. Bei diesen zwei Verhaltensmustern verlieren die Ergebnisse der Tests an Aussagekraft, da die Bearbeitungsdauer und die Arbeitsweise durch das laute Denken beeinflusst sind.

Beim retrospektiven lauten Denken wird der Prozess des lauten Denkens im Anschluss an die Testdurchführung angestoßen. Das Verhalten der Testpersonen während des Tests wird aufgezeichnet. Die Aufnahme wird dann anschließend vom Testleiter und der Testperson gesichtet, und die Testperson soll versuchen seine Gedanken zu der Benutzung des Systems zu verbalisieren. Die Probleme durch das unnatürliche Verhalten beim nebenläufigen lauten Denken werden so vermieden, es besteht aber das Risiko, dass die Testperson sich nicht mehr an alle Gedanken erinnern kann.

Eine dritte Variante des lauten Denkens stellt die **Constructive Interaction** [Vgl. Nielsen93] dar. Dabei wird dem Unwohlsein auf Grund des unnatürlichen Verhaltens entgegengewirkt, in dem die Untersuchung mit zwei Probanden durchgeführt wird. Wenn zu zweit an einem Problem gearbeitet wird, ist es natürlich das die Partner sich über mögliche Lösungsstrategien unterhalten. Dadurch werden oftmals mehr Gedanken geäußert, als bei Untersuchungen mit nur einer Person. Wenn der Umgang mit Computern unterschiedlich ist, oder die Probanden nicht miteinander arbeiten können, kann diese Methode jedoch zu Problemen führen. Die Art und Weise wie das Testobjekt genutzt wird kann dann häufig wechseln, und der Lernfortschritt für beide wird dadurch behindert.

2.2.4 Fragebögen und Interviews

Fragebögen und Interviews stellen eine weitere Usability-Untersuchungsmethode dar. Durch die Befragung von Anwendern oder Testpersonen können Informationen über die Benutzung und die Meinung über das Testobjekt gewonnen werden.

Fragebögen und Interviews sind grundsätzlich sehr ähnlich, da beide aus einer Menge von Fragen bestehen. Der Vorteil von Fragebögen ist, dass diese selbstständig ausgefüllt werden können. Daher kann der Befragte auch nicht beeinflusst werden, und seine Meinung ehrlich wiedergeben. Trotzdem kann es bei der Auswertung zu Problemen kommen, da Antworten bei offenen Fragen unklar formuliert sein können.

Bei einem Interview hingegen werden dem Befragten von einem Interviewer die Fragen vorgelesen. Dadurch ist es möglich bei unklar formulierten Antworten nachzufragen, aber auch auf Unklarheiten in der Fragestellung einzugehen. Generell ist ein Interview flexibler als ein Fragebogen, da z. B. bei interessanten oder unzureichenden Antworten Folgefragen gestellt

werden können. Der Nachteil hierbei ist aber, dass bei übermäßigem Gebrauch dieser Möglichkeit die Interviews schwer zu analysieren sind.

Eine allgemeine Begrenzung der Möglichkeiten der Befragung steckt in der Methodik selbst: Es wird nur die Meinung oder Auffassung der Anwender oder Testpersonen erfragt, und es findet keine Evaluierung des eigentlichen User Interfaces statt. Dieses Problem zeigt sich z. B. bei der Bewertung einzelner Features: Hierfür werden z. B. oft Zahlenskalen verwendet. Es ist interessant zu erfahren, welche Features beliebt sind und welche nicht, für die Weiterentwicklung und Verbesserung des Systems ist aber eigentlich wichtig warum ein bestimmtes Feature unbeliebt ist.

2.2.5 Kombination von Testmethoden

Die unterschiedlichen Usability-Untersuchungsmethoden haben alle individuelle Stärken und Schwächen. Daher ist es sinnvoll in der Praxis eine Kombination der Testmethoden zu bilden um die möglichen negativen Effekte zu verringern.

Eine mächtige Kombination stellt z. B. die Ergänzung des Usability-Tests mit lautem Denken und einem Nachinterview dar. Bei Tests mit lautem Denken entstehen leicht Situationen, an denen man die Testperson gerne über bestimmte Dinge ausfragen würde, sie aber auch nicht in ihrem Arbeits- oder Redefluss unterbrechen möchte, da dies den geplanten Testablauf durcheinander bringt. Durch die Kombination mit einem Nachinterview ist es möglich den Test weiter laufen zu lassen und im Nachinterview Fragen zu den interessanten Auffälligkeiten zu stellen, die auf diese Punkte und weitere Auffälligkeiten einzugehen. Ebenfalls denkbar ist zusätzlich oder alternativ zu dem Nachinterview ein Fragebogen. Dies ist sinnvoll wenn ausgeschlossen werden soll, dass die Testperson durch das Gespräch im Nachinterview beeinflusst wird.

2.3 Game Usability

Im Computerspiele-Kontext kann der standardisierte Usability-Begriff leicht falsch verstanden werden, da die effiziente Aufgabenlösung bei Spielen grundsätzlich nicht angestrebt wird. Bei Computerspielen bietet es sich an nicht von Effizienz zur Aufgabenlösung, sondern von Kontrolle zur Bewältigung von Herausforderungen zu sprechen. Die Interaktion des Spielers mit der Spielwelt erfolgt durch unterschiedliche Interaktionsmodelle (vgl. 2.1.3.2). In jedem dieser Modelle wird dem Spieler die Kontrolle über Teile der Spielwelt verliehen. Dies kann ein Avatar sein, eine Gruppe von Charakteren oder eine riesige Armee. In dem Spiel *Popolous* ist der Spieler sogar in der Lage Landschaften zu transformieren und Naturkatastrophen auszulösen. Des Weiteren gibt es aber auch Gemeinsamkeiten zwischen Computerspielen und Desktopanwendungen. Die Erlernbarkeit und die Verständlichkeit sind bei Computerspielen ebenfalls von großer Bedeutung. Ein Computerspiel sollte es einem Spieler ermöglichen schnell in das Spielgeschehen einzusteigen. Des Weiteren sollte es nicht nötig sein, dass ein Spieler nach einer längeren Unterbrechung alles neu erlernen muss. Dafür ist

es wichtig, dass die einzelnen Aspekte des Spiels verständlich sind und die Bedienungsmuster einprägsam gestaltet sind.

Eine Definition für die Usability von Computerspielen ist *der Grad in dem ein Spieler ein Computerspiel erlernen, kontrollieren und verstehen kann* [Pinelle081]. Diese Definition umfasst die oben genannten Punkte, und schließt ästhetische (z. B. Grafikgestaltung und Musik) und technische (z. B. Grafik- und Audioqualität) Probleme aus. Dies ist auch sinnvoll, da diese Aspekte zwar für den Unterhaltungsgrad eines Computerspiels von großer Bedeutung sind, aber keine Usability-Aspekte darstellen. Die Zufriedenheit des Benutzers ist in dieser Definition aber nicht berücksichtigt. Da der Zweck eines Computerspiels die Unterhaltung ist, sollte die Zufriedenheit des Benutzers aber nicht nur berücksichtigt werden, sondern ein wichtiger Bestandteil der Game Usability sein. Da das Gameplay für die Unterhaltung des Spielers von entscheidender Bedeutung ist (vgl. 2.1.3.2), ist dieses für die Zufriedenheit des Benutzers ebenfalls von großer Bedeutung. Die Game Usability muss also auch Aspekte des Gameplays berücksichtigen, z. B. wenn das Spielerlebnis durch ungewollte Herausforderungen getrübt wird die im Game Design nicht vorgesehen sind. Abb. 5 verdeutlicht die für die Game Usability relevanten Bereiche im beschriebenen Komponentenmodell (vgl. 2.1.3).

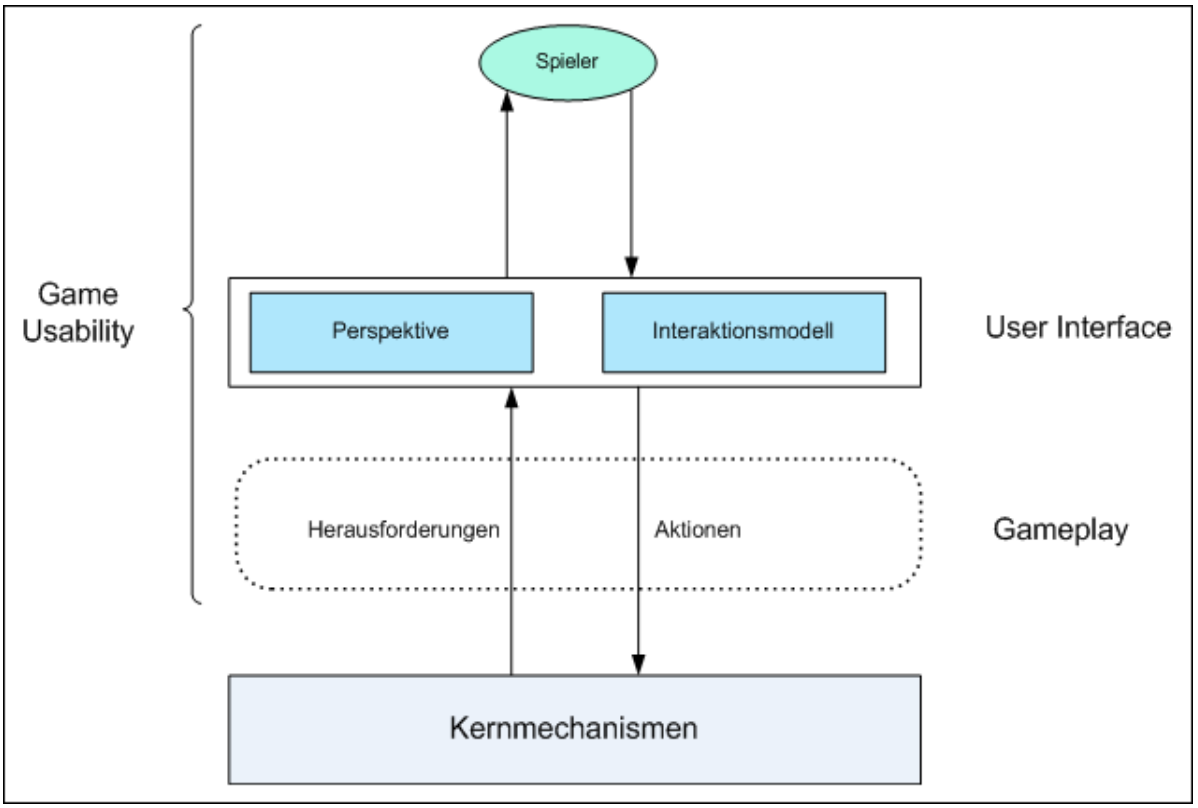

Abb. 5: Game Usability im Komponentenmodell

2.3.1 Problemverteilung in unterschiedlichen Genres

Wie unter 2.1.1 beschrieben, gibt es bei mehreren Spielen des gleichen Genres oft sehr viele Gemeinsamkeiten hinsichtlich des Gameplays und des User Interfaces. Daher ist davon auszugehen, dass Spiele des gleichen Genres ähnliche Usability-Probleme aufweisen. Dieser Ansatz wurde von [Pinelle082] verfolgt. Diese Veröffentlichung analysiert das Auftreten von zwölf Problemkategorien in den sechs Genres Action, Adventure, Shooter, Sport, Strategie und Rollenspiel. Das Ergebnis zeigt, dass es je nach Genre tatsächlich eine starke Verschiebung der Problemschwerpunkte gibt, wie an dem Beispiel des Action- und Strategie-Genres auf den folgenden Grafiken deutlich wird:

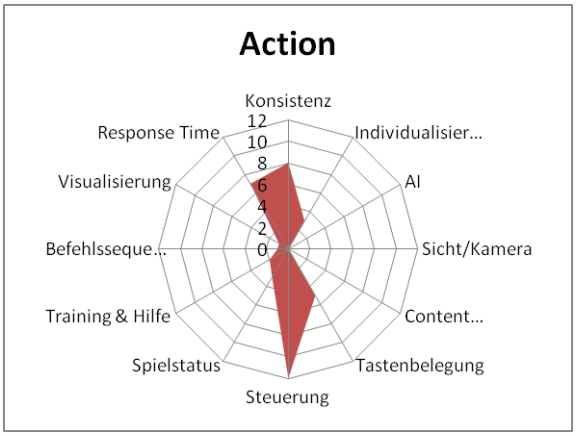

Abb. 6: Problemverteilung Action-Genre

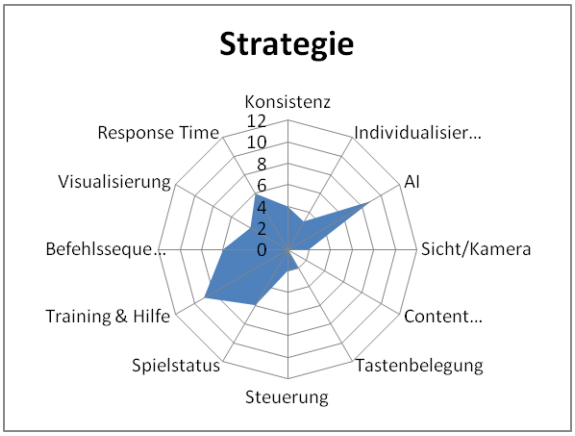

Abb. 7: Problemverteilung Strategie-Genre

Demnach liegen die Usability-Probleme im Action-Genre hauptsächlich in der Steuerung und der Konsistenz des Spiels. Bei Strategiespielen häufen sich die Probleme hingegen auf Grund der künstlichen Intelligenz, dem Training und der Hilfe eines Spiels.

3 Entwurf der Usability-Tests

In diesem Kapitel wird der Entwurf der Usability-Tests die im Rahmen dieser Arbeit durchgeführt wurden näher beschrieben. Dazu werden die Auswahl der Spiele, der Entwurf der Testpläne, und die Auswahl der Test-Methoden erläutert.

3.1 Auswahl der Spiele

Es wurden Spiele aus unterschiedlichen Genres ausgewählt, damit die Ergebnisse nicht zu spezifisch werden und ein Vergleich von Gemeinsamkeiten und Unterschieden zwischen den Vertretern der unterschiedlichen Genres möglich ist.

Wie im vorherigen Kapitel beschrieben, lässt sich je nach Genre eine unterschiedliche Häufung von Usability-Problemen feststellen. Außerdem weisen die Vertreter eines Genres Gemeinsamkeiten im User Interface auf (vgl. 2.1.1), verwenden also häufig identische Interaktionsmodelle und Perspektiven. Daher ist anzunehmen, dass bereits die gewählten Interaktionsmodelle und Perspektiven zu unterschiedlichen Problemen oder zu einer unterschiedlichen Häufung von Problemen führen. Bei einem Strategiespiel mit einem omnipräsentem Interaktionsmodell, bei dem sehr viele Einheiten kontrolliert werden ist z. B. die künstliche Intelligenz zur Wegfindung der Einheiten wichtig, damit Bewegungsbefehle auch korrekt ausgeführt werden. In einem Action-Spiel in dem man nur einen Charakter steuert ist dieses Problem hingegen ausgeschlossen, da entsprechende Aktionen auf Grund des Avatar-basierten Interaktionsmodells nicht möglich sind. Genauso können Probleme durch die Wahl der Perspektive entstehen und ausgeschlossen werden. Wird eine Verfolgerperspektive gewählt, kann diese an manchen Stellen im Spielverlauf ungünstig positioniert sein und Nachjustierung erfordern. Bei Verwendung der Ego-Perspektive kann dieses Problem jedoch nicht auftreten. Im Gegenzug kann aber die Übersicht verloren gehen, da man keinen Überblick über das gesamte Spielgeschehen hat. Stattdessen ist stets nur der eingeschränkte Blickwinkel auf die Spielwelt sichtbar, der durch die virtuellen Augen des kontrollierten Avatars erzeugt wird. Auf Grund dieser Unterschiede wurde bei der Wahl der Spiele darauf geachtet, dass sie sich hinsichtlich der Gestaltung des User Interfaces unterscheiden und unterschiedliche Perspektiven und Interaktionsmodelle durch die Spiele abgedeckt werden.

Des Weiteren wurde ein Spiel für die Wii-Konsole ausgewählt, damit Besonderheiten, die durch die Benutzung durch die Wii Remote entstehen, aufgedeckt werden können.

Die ausgewählten Spiele sind *Anno 1701*, *Mirror's Edge, Drakensang* und *Boom Blox*. *Anno 1701* und *Mirror's Edge* enthalten Elemente aus mehreren Genres. Bei Strategie- und Actionspielen schien dies sinnvoll, da es in diesen Bereichen sehr viele Sub-Genres gibt die im Rahmen dieser Arbeit nicht abgedeckt werden können. Gleichzeitig ist aber anzunehmen, dass zumindest die Richtung der Probleme in den meisten Sub-Genres ähnlich ist, da Gemeinsamkeiten im Gameplay und User Interface vorhanden sind. Mit *Drakensang* wurde ein Rollenspiel ausgewählt. Hier wurde auf einen Genre-Mix bewusst verzichtet, da die häufig

anzutreffenden Action-Rollenspiele ein teilweise stark abweichendes Gameplay bieten. Tabelle 2 zeigt eine Übersicht über die ausgewählten Spiele. *Boom Blox* ist ein Wii-Spiel, das mit mehreren Personen gespielt werden kann. Wie für viele Wii-Spiele typisch ist das Spielprinzip relativ einfach gehalten. Es handelt sich also um ein „Multiplayer-Casual[1]"-Spiel.

Genre	Interaktionsmodell	Perspektive	Repräsentant (Plattform)
Strategie (Strategie, Aufbau- und Wirtschaftssimulation)	omnipräsent	Vogelperspektive	**Anno 1701 (PC)**
Action (Ego-Shooter, Jump'n'Run, Action-Adventure)	Avatar basiert	Ego-Perspektive	**Mirror's Edge (PC)**
Rollenspiel	Gruppen basiert	Drittperson-Perspektive	**Drakensang (PC)**
Multiplayer/Casual	omnipräsent	Vogelperspektive	**Boom Blox (Wii)**

Tabelle 2: Auswahl der Spiele

3.1.1 Anno 1701

Anno 1701 verbindet Elemente aus Strategiespielen mit dem Prinzip der Aufbau- und Wirtschaftssimulationen. Das Grundprinzip des Spiels ist die Besiedlung von Inseln und die Befriedigung der Bedürfnisse der Einwohner der Siedlungen. Hierfür ist die Handhabung eines komplexen Ressourcenmanagementsystems nötig, da Rohstoffe abgebaut und Güter produziert werden müssen. Diese Ressourcen werden teilweise direkt zur Befriedigung von Bedürfnissen benötigt (z. B. Nahrung um Hunger-Bedürfnis zu stillen) und teilweise weiterverarbeitet um andere Güter zu produzieren (z. B. Wolle um Stoffe zu produzieren) oder besondere Gebäude zu errichten. Da Ressourcen begrenzt sein können, ist es nötig mit anderen Mitspielern um rohstoffreiche Inseln zu konkurrieren, und womöglich mit diesen Krieg zu führen. Friedliche Lösungen in Form von Handel und Diplomatie sind ebenfalls möglich.

Das in Echtzeitstrategiespielen sehr wichtige *Micromanagement,* also die Kontrolle und Steuerung von vielen Einheiten gleichzeitig, ist in *Anno 1701* weniger stark vertreten. Wenn Kriegseinheiten oder Schiffe kontrolliert werden, ist es zwar vorhanden, hat aber nicht den gleichen Stellenwert wie in klassischen Echtzeit-Strategiespielen wie *Warcraft* und *Starcraft*, in denen die Steuerung und Kontrolle einer Vielzahl von Einheiten den Großteil des Spiels ausmacht.

Wie in Strategie- und Aufbauspielen üblich, bietet *Anno 1701* ein omnipräsentes Interaktionsmodell in Kombination mit der Vogel-Perspektive.

3.1.2 Mirror's Edge

Mirror's Edge ist eine Mischung aus Action-Adventure, Plattformspiel und Ego-Shooter. Der Großteil des Gameplays basiert auf den agilen Bewegungsmöglichkeiten des kontrollierten

[1] Casual Games („Gelegenheitsspiele"): Leicht zugängliche, oft wenig komplexe Spiele die schnell zu Erfolgserlebnissen führen.

Charakters mit deren Hilfe man sich auf und in den Gebäuden der Spielwelt fortbewegt. Die Bewegungsmöglichkeiten orientieren sich an dem Trendsport Parkour. Gelegentlich kommt es auch zu Kämpfen, der Spielcharakter beherrscht hierfür Nahkampftechniken und kann Gegner entwaffnen. Waffen der Gegner können auch aufgenommen werden, dann gleicht das Gameplay dem eines Ego-Shooters. Die Spielfigur verliert hierdurch aber deutlich an Bewegungsgeschwindigkeit, wodurch das Führen von Waffen im Spielverlauf normalerweise nur kurzzeitig vorkommt.

Das User Interface besteht aus der Ego-Perspektive und einem Avatar-basiertem Interaktionsmodell. Eine Besonderheit ist der Verzicht auf Statusanzeigen durch ein HUD[2]. Bei Ego-Shootern und Action-Adventures wird dies oft verwendet um Informationen über den Spielstatus bereitzustellen, z. B. durch die Anzeige von Lebenspunkten oder des Munitionsvorrats. Bei *Mirror's Edge* werden Informationen dieser Art über das Gameplay, die grafische Darstellung und den Sound vermittelt. Hat man z. B. ein relativ hohes Tempo aufgenommen, so wird der äußere Rand des sichtbaren Ausschnitts der Spielwelt mit Bewegungsunschärfe dargestellt und der Luftzug wird deutlich hörbar.

3.1.3 Drakensang

Bei *Drakensang* handelt es sich um ein klassisches Computer-Rollenspiel. Am Anfang des Spiels erstellt der Anwender sich einen Charakter, bei dem die Rassen-, und Klassenwahl, sowie die Attributverteilung anhand des Regelwerks von das Schwarze Auge[3] vorgenommen wird. Im Verlauf des Spiels können sich Gefährten dem Spielcharakter anschließen, wodurch der Spieler die Kontrolle über mehrere Charaktere gewinnt. Mit dieser Gruppe werden dann Aufgaben gelöst, über die auch die Geschichte im Spiel vorangetrieben wird. Durch das Lösen von Aufgaben und meistern vom Kämpfen erhalten die Charaktere Erfahrungspunkte, die zur Verbesserung von Attributen oder zum Erlernen von neuen Fertigkeiten verwendet werden können.

Das User Interface bietet eine Verfolgerperspektive, bei der die Kamera-Position vom Spieler veränderbar ist, und ein Gruppen-basiertes Interaktionsmodell.

3.1.4 Boom Blox

Bei dem Wii-Spiel *Boom Blox* handelt es sich um ein Geschicklichkeitsspiel, wobei der Spielablauf eine Mischung aus Jenga und *Tetris*-ähnlichen Spielen darstellt. Es gibt mehrere Spielmodi, z. B. ein Jenga-orientierten Modus in dem es darum geht aus einem Stapel von Blöcken möglichst viele herauszuziehen ohne den Stapel umzuwerfen. In einem anderen Modus hingegen muss der Stapel mit gezielten Würfen zu Fall gebracht werden. Das Spiel enthält ein realistisches Physik-System, wodurch der Winkel indem ein Gegenstand geworfen oder gezogen wird entscheidenden Einfluss auf das Resultat der Aktion hat. Gleiches gilt

[2] Head-Up-Display: Ein Anzeigesystem, bei dem Informationen in das Sichtfeld des Nutzers projiziert werden.
[3] Das Schwarze Auge (DSA): Ein Pen-&-Paper-Rollenspiel.

für die Geschwindigkeit und Präzision der Bewegung des Spielers. Die Steuerung des Spiels erfolgt mit der Wii Remote.

Das User Interface stellt eine Kombination aus omnipräsenten Interaktionsmodell und einer anpassbaren Vogel-Perspektive dar. So ist es möglich die Kamera in der Spielwelt zu drehen um eine möglichst optimale Perspektive sicherzustellen.

3.2 Auswahl der Testmethoden

Die Auswahl der anzuwendenden Testmethoden orientierte sich an klassischen Usability-Tests und war abgesehen von der Anwendung der Methode des lauten Denkens für alle Spiele identisch.

Es wurde geplant, die Probanden während des Tests zu beobachten und Auffälligkeiten zu notieren. Im Anschluss an die Spielphase sollte ein Nachinterview durchgeführt werden um auf besondere Auffälligkeiten einzugehen und die Meinung des Probanden über das Testobjekt zu erfahren. Zusätzlich sollten die Tests aufgezeichnet werden, um neben den notierten Auffälligkeiten eine weitere Datenquelle für die Auswertung heranziehen zu können. Dies kann hilfreich sein falls Probleme während der Beobachtung übersehen oder nicht verstanden werden.

Im Folgenden wird die Verwendung der Methode des lauten Denkens für die einzelnen Spiele besprochen.

3.2.1 Strategie

Der Vertreter für das Strategie-Genre ist *Anno 1701*. Dieses Spiel bietet im Vergleich zu reinen Echtzeit-Strategiespielen einen relativ ruhigen Spielablauf, der in dem hohen Anteil der Wirtschafts- und Aufbauelemente am Gameplay begründet ist. Daher wurden die Probanden zu einem nebenläufigen lauten Denken aufgefordert. Allerdings sollte nicht jeder Gedanke geäußert werden, sondern nur Gedanken zu besonderen Auffälligkeiten. Da Besonderheiten für den Testleiter nicht unbedingt Besonderheiten für den Probanden darstellen müssen, wurde bei Bedarf der Proband durch den Testleiter zum lauten Denken aufgefordert. Es ist zu erwarten, dass trotzdem ein natürlicher Spielablauf gewährleistet ist, da das laute Denken nur gelegentlich eingesetzt werden soll.

3.2.2 Action

Der Vertreter des Action-Genres ist *Mirror's Edge*. Da das Gameplay von *Mirror's Edge* sehr temporeich gestaltet ist, und die Bewegungsmöglichkeiten des kontrollierten Avatars eine gewisse Übung und Konzentration erfordern, wurde auf ein begleitendes lautes Denken verzichtet. Bei diesem Spiel würde dies einen sehr starken Eingriff in den natürlichen Spielablauf darstellen. Der Spieler würde vermutlich Fehler machen, die ohne Unterbrechung und Ablenkung durch das laute Denken nicht auftreten. Stattdessen wird ein retrospektives lautes Denken durchgeführt. Im Anschluss an die Spielphase wird die Videoaufnahme gesichtet und

der Proband aufgefordert, seine Gedanken und Erinnerungen zu auffälligen und interessanten Spielsituationen zu äußern.

3.2.3 Rollenspiel

Der Vertreter des Rollenspiel-Genres ist *Drakensang*. Das Gameplay von *Drakensang* ist rollenspieltypisch. Die Story des Spiels spielt eine wichtige Rolle, und es gibt viele Dialoge zwischen den Spieler-Charakteren und Nicht-Spieler-Charakteren. Die Kämpfe können pausiert werden um die Aktionen der kontrollierten Charaktere sorgfältig zu planen. Der Spielablauf ist im Gegensatz zu einem Action-Spiel daher ruhig. Dies sollte es ermöglichen die Probanden zu einem nebenläufigen lauten Denken aufzufordern, und trotzdem einen natürlichen Spielablauf beobachten zu können.

3.2.4 Multiplayer

Als Vertreter für ein Multiplayer-Spiel wurde *Boom Blox* ausgewählt. Die Tests werden mit jeweils zwei Probanden durchgeführt. Daher ist zu erwarten, dass Probleme zwischen diesen beiden diskutiert werden. Auf eine Aufforderung zu lautem Denken wird daher verzichtet.

3.2.5 Übersicht der ausgewählten Testmethoden

Die folgende Tabelle 3 zeigt eine Übersicht über die ausgewählten Testmethoden.

Genre (Vertreter)	Beobachtung	Aufzeichnung	Nachinterview	Lautes Denken
Strategie (Anno 1701)	Ja	Ja	Ja	Nebenläufig
Action (Mirror's Edge)	Ja	Ja	Ja	Retrospektiv
Rollenspiel (Drakensang)	Ja	Ja	Ja	Nebenläufig
Casual/Multiplayer (Boom Blox)	Ja	Ja	Ja	Nein

Tabelle 3: Auswahl der Testmethoden

3.3 Probanden

Die bereits für die Auswahl der Spiele relevanten Gemeinsamkeiten der Vertreter eines Genres sind auch für die Wahl der Probanden von Bedeutung. Wenn ein Proband den Umgang mit Spielen des Genres erlernt hat, können die Kenntnisse auf Grund der vergleichbaren Bedienung und Spielinhalte auf andere Spiele übertragen werden. Das Erlernen der Bedienung anderer Spiele des Genres fällt dann leichter, als bei Probanden, die noch nie ein vergleichbares Spiel gespielt haben. Daher sollen geübte und ungeübte Spieler für die Spiele als Probanden ausgewählt werden.

Nach [Nielsen93] ist für die meisten Projekte eine Anzahl von fünf Tests ausreichend. Man kann sich bei der geringen Anzahl an Probanden zwar sicher sein, dass nicht jedes Problem gefunden wird, der Großteil der Problemfelder wird aber auch bei der geringen Anzahl an Probanden abgedeckt sein. Da das Ziel der Tests das Aufdecken von typischen Problemen ist, und nicht die Verbesserung der untersuchten Computerspiele, ist die Aufdeckung von allen

denkbaren Problemen nicht zwingend erforderlich. Daher wurden fünf Tests pro Spiel geplant. Dies ergibt eine Gesamtanzahl von 25 Probanden: Jeweils fünf für die Tests mit *Anno 1701*, *Drakensang* und *Mirror's Edge* und zehn für die Tests mit Boom Blox.

3.4 Testleitfaden

Das Ziel der Usability-Tests, die im Rahmen dieser Arbeit durchgeführt wurden, war es zum einen Erkenntnisse über die verwendeten Usability-Testmethoden im Computerspiele-Kontext zu gewinnen, und zum anderen Usability-Probleme von Computerspielen zu identifizieren.

Um die Ergebnisse so wenig wie möglich durch die künstliche Testsituation zu verfälschen, sollte ein natürlicher Spielablauf sichergestellt werden. Daher wurde auf eine Vorauswahl von zu evaluierenden Spielaspekten verzichtet, damit keine ungewollte Filterung der Usability-Probleme durch die Wahl der Spielaspekte entsteht. In diesem Zusammenhang wurde auch auf eine Formulierung einer sequenziell abzuarbeitenden Aufgabenliste verzichtet, die bei Usability-Tests von Anwendungssoftware und Webapplikationen oft verwendet wird. Vorgegeben wurde den Probanden höchstens der Spielmodus. Daraus resultierte ein allgemeiner Ablaufplan mit den vier Phasen Einstieg, Einführung/Tutorial, freies Spiel und Nachbesprechung:

1. **Einstieg**: In dieser Phase wählt der Spieler seinen bevorzugten Spielmodus oder modifiziert Einstellungen wie die Tastaturbelegung oder Grafikoptionen. Er befindet sich also noch nicht im eigentlichen Spiel. Diese Phase sollte relativ zügig durchlaufen werden und nur einen kleinen Teil des Usability-Tests ausmachen. Mögliche Probleme in dieser Phase sind ungünstige Bezeichnungen und fehlende Informationen zu Auswahlmöglichkeiten und Spielmodi.
2. **Einführung/Tutorial**: Diese Phase tritt optional auf, wenn das Spiel eine Einführung oder ein Tutorial bietet und die Testperson dieses auswählt. Das Ziel einer Einführung ist die Vermittlung der grundlegenden Aktions- und Steuerungsmöglichkeiten. Daher ist die Verständlichkeit der gegebenen Erklärungen und Anweisungen von entscheidender Bedeutung.
3. **Freies Spiel**: Diese Phase sollte den zeitlich größten Teil des Usability-Tests in Anspruch nehmen, da erst in dieser Phase das eigentliche Spiel gespielt wird. Die Probleme die in dieser Phase auftreten sollten durch das Interaktionsmodell, die gewählte Perspektive und der spezifischen Umsetzung des Spiels begründet sein.
4. **Nachbesprechung**: Unabhängig ob lautes Denken eingesetzt wird oder nicht, wird im Nachhinein nochmal auf Auffälligkeiten und Äußerungen der Testperson eingegangen. Dadurch sollen Unklarheiten und Verständnisprobleme auf Seiten des Testleiters ausgeräumt werden, um die folgende Auswertung zu erleichtern. Des Weiteren ist es an dieser Stelle möglich, die Meinung des Probanden über das Spiel und bestimmte Spielaspekte zu erfragen. Dies kann zum einen Problemfelder offenlegen die sonst

unbeachtet geblieben wären, und zum anderen Ansatzpunkte für eine mögliche Klassifizierung der gefundenen Probleme geben.

Der Ablaufplan stellte die Grundlage für die Erstellung des Testleitfadens dar. Dieser dient dem Testleiter als Hilfe für die Testdurchführung, da sich zum einen der Testablauf wiederfinden lässt und zum anderen Ansatzpunkte für mögliche Auffälligkeiten und die Nachbesprechung enthalten sind. Ergänzt wurde der Testleitfanden noch um die Erfassung der Vorkenntnisse und Spiele-Affinität der Probanden. Der Aufbau des Testleitfadens sieht wie folgt aus:

1. **Allgemeines**
   1.1. Testdaten: Zeitpunkt und Nummer des Tests
   1.2. Proband: Alter, Computerspiele-Erfahrung, bevorzugte Plattformen und Genres
2. **Beobachtung**
   2.1. Menü/Spieleinstieg: Auffälligkeiten vor dem Einstieg ins eigentliche Spiel
   2.2. Tutorial oder Charaktererstellung: Auffälligkeiten falls ein Tutorial angeboten und gespielt wurde. Alternativ bei Drakensang die Phase der Charaktererstellung.
   2.3. Freies Spiel: Auffälligkeiten während des eigentlichen Spiels
3. **Nachgespräch**
   3.1. Allgemeines: Eindruck des Spiels
   3.2. Spieleinstieg und Steuerung: Qualität des Tutorials, Probleme beim Spieleinstieg und mit der Steuerung des Spiels
   3.3. User Interface und Spielablauf: Gestaltung des User Interfaces, der verwendeten Symbole und die Vollständigkeit der bereitgestellten Funktionen, Besonderheiten im Spielablauf
   3.4. Sonstiges: Auffälligkeiten und Besonderheiten die nicht in die anderen Kategorien passen

In der Beobachtungsphase gibt es eine Variation für das Spiel *Drakensang*. In diesem gibt es keine Möglichkeit vor dem eigentlichen Spieleinstieg ein Tutorial zu spielen. Dafür ist die Charaktererstellung aufwendig. Daher wurde in der Beobachtungsphase das Tutorial, das bei den anderen drei Spielen möglich ist, durch die Charaktererstellung ersetzt. Weitere Modifikationen waren für die einzelnen Spiele nicht nötig. Der methodische Unterschied zwischen den Tests, die unterschiedliche Anwendung des lauten Denkens, hat keine direkte Auswirkung auf den Testleitfaden. Das nebenläufige laute Denken bei *Anno 1701* und *Drakensang* erfolgt in der Beobachtungsphase, in der Auffälligkeiten notiert werden sollen. Die Kommentare des Probanden ebenfalls zu notieren, ist daher möglich. Die Sichtung der zu erzeugenden Aufzeichnung beim retrospektiven lauten Denken ist dem Nachgespräch zugeordnet, da der Übergang zu diesem fließend ist.

4 Durchführung der Usability-Tests

In diesem Kapitel wird die Testumgebung vorgestellt und die Durchführung der Usability-Tests beschrieben, dazu gehören die Aufbauten für die einzelnen Tests, die gewählten Probanden und eine Beschreibung der Testabläufe.

4.1 Testumgebung: Das Usability-Labor der HAW Hamburg

Durchgeführt wurden alle Tests in dem Usability-Labor des Departments Informatik der HAW Hamburg. Das Usability-Labor ist in zwei Räume aufgeteilt. Im Testraum befinden sich u. a. der Testrechner, auf dem das Testobjekt üblicherweise ausgeführt wird, und sechs Kameras. Die Bilder des Testrechners und der Kameras werden in den Regieraum weitergeleitet, wo diese aufgezeichnet und auf einem Beobachtungsmonitor ausgegeben werden (Vgl. Abb. 8).

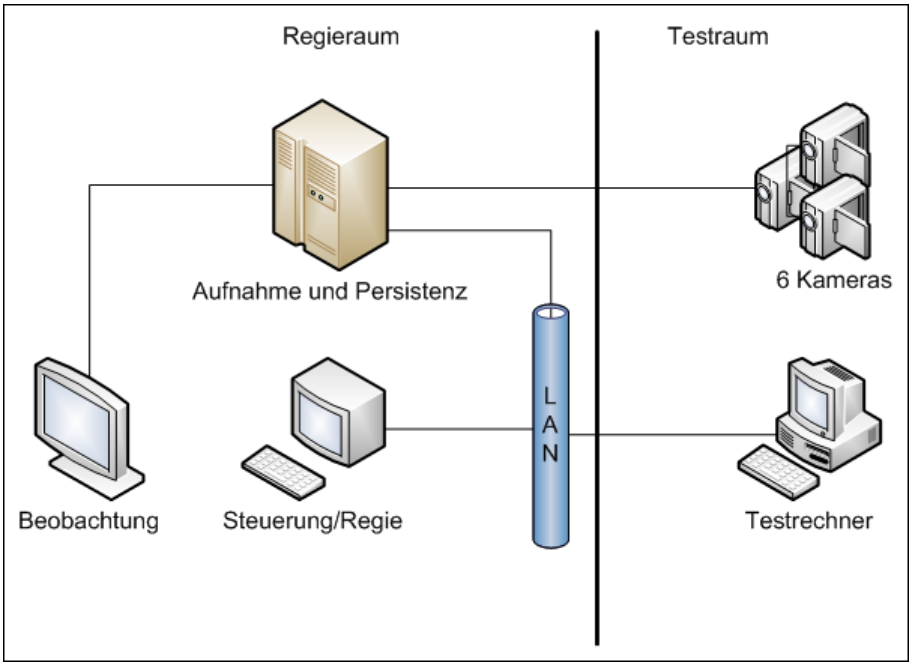

Abb. 8: Architektur des Usabiliy-Labors (vereinfacht)

4.1.1 Aufbau PC-Spiele

Für die Durchführung der Tests mit den PC-Spielen *Anno 1701, Drakensang* und *Mirror's Edge* konnte der Standardaufbau des Usability-Labors verwendet werden. Die technische Vorbereitung bestand aus der Installation der Spiele auf dem Testrechner und der anschließenden Prüfung der Testumgebung in Form von Probeaufnahmen. Dabei traten keine Probleme auf.

4.1.2 Aufbau Wii-Spiel

Mit *Boom Blox* wurden Tests mit einem Spiel für die Nintendo Wii durchgeführt. Da eine Konsole im Usability-Labor zur Verfügung steht, konnte diese genutzt werden. In der Vorbereitung zeigten sich aber zwei Probleme: Die Wii ließ sich nicht problemlos in den Aufnahmeprozess des Labors integrieren, da die technische Ausstattung des Labors auf PCs ausge-

legt ist. Des Weiteren sind die üblicherweise im Labor verwendeten Monitore relativ klein, was zu einem schlechten Spielerlebnis mit der Wii führt. Die Probleme wurden durch einen Umbau der üblichen Umgebung gelöst. Das normalerweise als Beobachtungs-Monitor verwendete TV-Gerät aus dem Regie-Raum wurde im Testraum verwendet. Eine Kamera wurde auf das Bild des TV-Gerätes ausgerichtet. Die Aufzeichnung des Bildes der Wii erfolgte daher aus einer seitlichen Perspektive. Da die Komplexität der einzelnen Spiele vom *Boom Blox* relativ gering ist, stellte dies kein Problem in der späteren Auswertung dar.

Abb. 9: Testaufbau Boom Blox

4.2 Probanden

Die Probanden wurden aus meinem persönlichen und studentischen Umfeld rekrutiert. Die unter 3.3 beschriebenen Auswahlkriterien konnten fast vollständig erfüllt werden. Im Folgenden wird für jedes Spiel ein Überblick über die Probanden gegeben.

4.2.1 Anno 1701

Insgesamt wurden mit *Anno 1701* fünf Tests durchgeführt. Alle Probanden wiesen Vorkenntnisse mit Vorgängern oder ähnlichen Spielen auf. Da es sich um ein beliebtes Genre und ein bekannten Spiel handelt, war es nicht möglich Probanden ohne Vorkenntnisse zu finden.

Nr	Alter	Geschlecht	Spiele-Affinität	Vorkenntnisse
1	31	m	Hoch	Genre bekannt
2	29	m	Hoch	Vorgänger und Genre bekannt
3	28	w	Mittel	Vorgänger und Genre bekannt
4	38	m	Mittel	Genre bekannt
5	23	m	Sehr Hoch	Vorgänger und Genre bekannt

Tabelle 4: Probanden Anno 1701

4.2.2 Mirror's Edge

Insgesamt wurden fünf Tests mit *Mirror's Edge* durchgeführt. Zwei Probanden hatten keine Erfahrung mit der Ego-Perspektive in Spielen. Ein Proband kannte die Ego-Perspektive und

das entspreche Bedienkonzept, allerdings hatte dieser Proband längere Zeit kein entsprechendes Spiel mehr gespielt. Zwei weitere Probanden wiesen Kenntnisse von aktuellen, vergleichbaren Spielen auf.

Nr	Alter	Geschlecht	Spiele-Affinität	Vorkenntnisse
1	38	m	Mittel	Leichte Kenntnis des Genres
2	23	m	Sehr Hoch	Genre bekannt
3	31	m	Hoch	Genre nicht bekannt
4	28	w	Mittel	Genre nicht bekannt
5	29	m	Hoch	Genre bekannt

Tabelle 5: Probanden Mirror's Edge

4.2.3 Drakensang

Insgesamt wurden mit *Drakensang* fünf Tests durchgeführt. Zwei Probanden wiesen Vorkenntnisse durch andere Spiele des Genres auf und zwei Probanden hatten keine Erfahrung mit Computer-Rollenspielen. Ein Proband kannte das Genre grundsätzlich, hatte aber keine Kenntnisse aktueller Genre-Vertreter.

Nr	Alter	Geschlecht	Spiele-Affinität	Vorkenntnisse
1	29	m	Hoch	Genre bekannt
2	38	m	Mittel	Leichte Kenntnis des Genres
3	28	w	Mittel	Genre bekannt
4	23	m	Sehr hoch	Genre unbekannt
5	42	m	Mittel	Genre unbekannt

Tabelle 6: Probanden Drakensang

4.2.4 Boom Blox

Insgesamt wurden mit *Boom Blox* vier Tests mit jeweils zwei Probanden durchgeführt. Die insgesamt acht Probanden setzten sich aus zwei regelmäßigen, drei gelegentlichen und drei ungeübten Wii-Spielern zusammen. Aus terminlichen Gründen musste der geplante fünfte Test ausfallen. Auf Grund der vergleichbaren Probleme der ersten vier Tests wurde auf ein Nachholtermin verzichtet, da keine grundlegend neuen Erkenntnisse mehr zu erwarten waren.

Team-Nr.	Alter	Geschlecht	Spiele-Affinität	Vorkenntnisse
1	29	m	Hoch	Regelmäßiger Wii-Spieler
	28	w	Mittel	Gelegentlicher Wii-Spieler
2	23	m	Sehr hoch	Gelegentlicher Wii-Spieler
	30	m	Mittel	Ungeübt
3	25	m	Sehr hoch	Regelmäßiger Wii-Spieler
	22	w	Gering	Ungeübt
4	26	w	Gering	Ungeübt
	23	w	Mittel	Gelegentlicher Wii-Spieler

Tabelle 7: Probanden Boom Blox

4.3 Testdurchführung

Der geplante Testablauf (vgl. 3.4) konnte ohne abweichende Vorkommnisse umgesetzt werden. Die Testdurchführung folgte demnach bei allen Spielen dem gleichen Schema und bestand aus drei Phasen:

1. **Vorbereitung**: Vor dem Start des Tests wurde sichergestellt, dass die Testplattform im Ursprungszustand ist, damit alle Probanden mit den Standardeinstellungen starten und nicht den Spielstand eines anderen Tests fortführen. Des Weiteren wurde darauf geachtet, dass ein Testleitfaden bereit liegt. Vor dem Start der Testphase wurde den Testpersonen noch die Testumgebung erläutert. Anschließend wurde die Aufzeichnung gestartet.
2. **Testphase:** Die eigentliche Testdurchführung anhand des unter 3.4 erläuterten Testleitfadens. Die Dauer betrug jeweils eine Stunde. Bei *Mirror's Edge* wurde die Spielphase abhängig vom Eindruck des Testleiters verkürzt, damit trotz dem retrospektiven lauten Denken der Zeitrahmen eingehalten wird.
3. **Nachbereitung:** Nach Beendigung der Testphase wurde die Aufzeichnung gestoppt, und der Ursprungszustand der Testplattform wieder hergestellt.

5 Auswertung

Die Auswertung setzte bereits während der Testphase an: Während der Testdurchführung wurden bereits viele Auffälligkeiten im Testleitfaden notiert. Diese Dokumente dienten daher zusammen mit den Videoaufnahmen der einzelnen Tests als Grundlage für die Auswertung.

Im ersten Schritt der Auswertung wurden die in den Testleitfäden notierten Auffälligkeiten aller Tests gesammelt. Anschließend wurden die Videoaufnahmen gesichtet, um weitere Auffälligkeiten zu identifizieren. Die einzelnen Auffälligkeiten wurden jeweils mit einer kurzen, stichpunktartigen Erklärung in einer Tabelle gesammelt.

Des Weiteren wurde eine Gewichtung der Auffälligkeiten vorgenommen, da z. B. fundamentale Steuerungsprobleme ein deutlich größeres Usability-Problem darstellen als eine lückenhafte Beschreibung, die nur von wenigen Anwendern gelesen wird. Hierfür wurden aus meinen Erfahrungen mit Usability-Untersuchungen fünf Problemklassen bestimmt (Tabelle 8) und anschließend die gesammelten Auffälligkeiten diesen zugeordnet.

Problemklasse	Beschreibung
1	Katastrophales Problem: Die effektive Bedienung ist nicht möglich
2	Schwerwiegendes Problem: Die effektive Bedienung des Spiels ist eingeschränkt
3	Offensichtliches Problem: Die effiziente Bedienung des Spiels ist eingeschränkt
4	Leichtes Problem: Die effiziente Bedienung des Spiels ist teilweise eingeschränkt
5	Kosmetisches Problem: Auffälligkeiten die vernachlässigt werden können

Tabelle 8: Problemklassen

Im Folgenden werden die identifizierten Usability-Probleme und die Verteilung der Auffälligkeiten auf die Problemklassen der einzelnen Spiele erläutert.

5.1 Anno 1701

Insgesamt wurden bei *Anno 1701* 36 unterschiedliche Probleme identifiziert. Katastrophale Probleme, das heißt Probleme der Klasse 1, wurden nicht gefunden (vgl. Abb. 10).

Leichte Probleme gibt es mit der grundlegenden Steuerung, sowie nicht auffindbaren und vermissten Funktionen, z. B. um den Spielablauf zu beschleunigen oder Häuser mit einer vom Standard abweichenden Ausrichtung zu bauen, damit diese besser in das ästhetische Erscheinungsbild der Siedlung passen.

Schwerwiegendere Probleme gab es beim Verständnis des Spiels, bestimmter Spielmechanismen und der Bedienung der Menüs. So war es mehreren Probanden nicht ersichtlich, wie genau die Entwicklung der Siedlung voranschreitet, welche Bedingungen hierfür erfüllt sein müssen, und wie diese erfüllt werden können. Probleme beim Verständnis der Spielmechanismen ergaben sich z. B. durch eine undeutliche Visualisierung des Wirkungsbereichs von einigen Gebäuden und fehlenden Informationen über den genauen Zusammenhang zwi-

schen Gebäuden. Zum Beispiel ist nicht ersichtlich, wie viele Schaffarmen eine Webstube auslasten. Eine weitere Häufung von Problemen gab es bei der Bedienung der unterschiedlichen Menüs im Spiel, unabhängig davon zu welchem Spielaspekt diese gehören. Im Bau-Menü waren z. B. einige Symbole unverständlich und in einem Menü mit einer Übersicht über die aktuelle Entwicklungsstufe der Siedler, deren Bedürfnisse und der Möglichkeit Steuern anzupassen, wurde jedes Element von mindestens einem Probanden nicht verstanden oder falsch gedeutet.

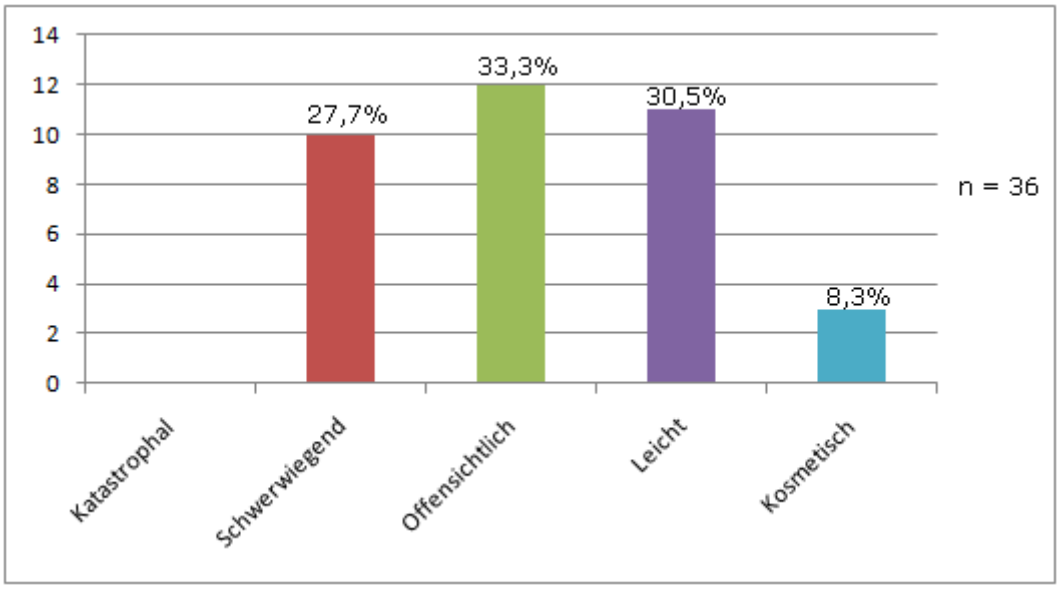

Abb. 10: Problemklassen Anno 1701

5.2 Mirror's Edge

Bei *Mirror's Edge* konnten 23 Auffälligkeiten unterschiedlicher Schwere identifiziert werden (vgl. Abb. 11). Es zeigte sich bereits während der Durchführung, dass die Dauer der Tests sehr knapp bemessen war. Die Probanden ohne spezifische Vorkenntnisse kamen kaum über das Tutorial hinaus, wodurch ein Großteil der gefundenen Probleme sich auch auf diesen Teilabschnitt des Spiels bezieht.

Vielfach waren die Beschreibungen der Übungen im Tutorial für die Probanden nicht verständlich. Mehrfach waren die Formulierungen nicht eindeutig und teilweise wurden Begriffe aus der Spielmechanik verwendet, die vorher nicht erläutert wurden. Zusätzlich wurden zwei Probanden durch die verwendeten Bezeichnungen der Tastatur- und Maustasten irritiert, z. B. wird die linke Maustaste als „Maustaste 1" bezeichnet. In Folge dieser Beschreibungen führte ein Ausprobieren von möglichen Aktionen zum Erfolg, wobei die Probanden durch die einleitende Beschreibung eher andere Aktionen vermutet haben.

Weitere Auffälligkeiten zeigten sich bei der Steuerung. Diese orientiert sich am Genre-Standard und ist größtenteils identisch mit anderen Spielen mit Ego-Perspektive. Die Probanden, die entsprechende Vorkenntnisse haben, kamen mit der Steuerung daher problem-

los zurecht. Ein komplett gegenteiliges Bild zeigte sich jedoch bei den Probanden ohne entsprechende Vorkenntnisse. Die grundlegende Steuerung, also das Bewegen des Avatars mit WASD[4], ist kein Bestandteil des Tutorials. Dadurch waren zwei Probanden direkt nach dem Einstieg in das Spiel ratlos, da sie die Standard-Tastaturbelegung nicht kannten. Beide Probanden haben anschließend die Tastaturbelegung geändert und die Bewegung auf die Pfeiltasten umgelegt. Diese Belegung ist aber sehr unhandlich, was dazu führte, dass die Probanden an den ersten Aufgaben auf Grund der Steuerung scheiterten und nach einigen Fehlversuchen wieder zur ursprünglichen Tastaturbelegung wechselten.

Ein weiteres Problem zeigte sich im Aufbau des Tutorials. Die einzelnen Übungen werden von einem computergesteuerten Avatar vorgeführt. Dies wird nach jedem Fehlversuch wiederholt, kann aber nicht unterbrochen werden. Einige der Übungen sind im Prinzip leicht zu verstehen, können aber etwas Übung seitens des Spielers erfordern, wodurch mehrere Versuche zum Bestehen der Aufgabe erforderlich sind. Daher versuchten mehrere Probanden erfolglos die Vorführung zu überspringen.

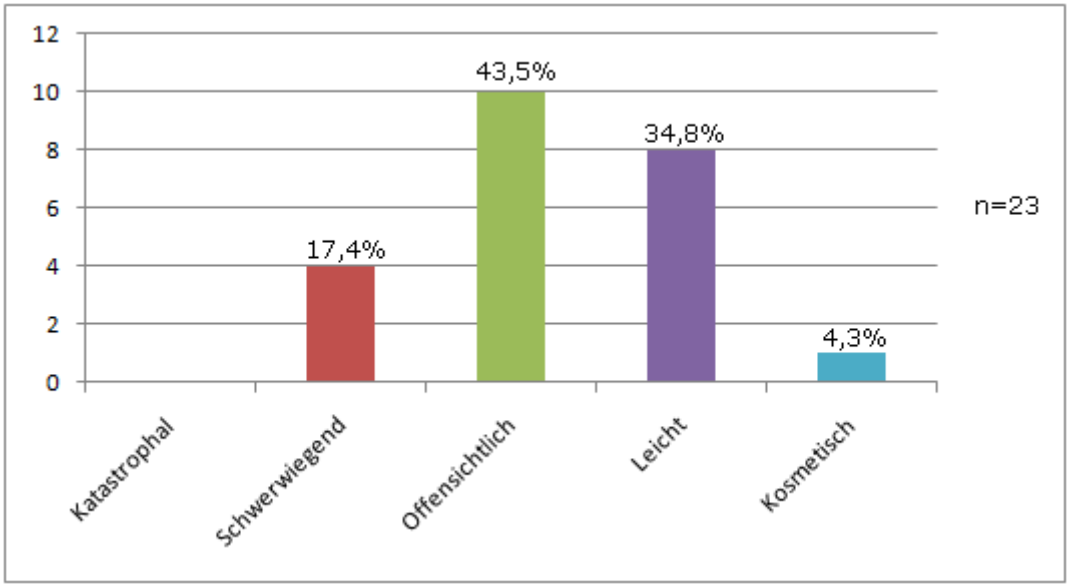

Abb. 11: Problemklassen Mirror's Edge

5.3 Drakensang

Insgesamt wurden bei Drakensang 23 Probleme identifiziert. Katastrophale Probleme wurden nicht gefunden, es konnten aber in unterschiedlichen Spielelementen einige schwerwiegende und offensichtliche Probleme identifiziert werden (vgl. Abb. 12).

Bei allen Probanden zeigten sich Probleme bei der Handhabung der Kamera, da diese regelmäßig eine Nachjustierung erfordert. Mehrere Probanden vermissten die Funktion einer intelligenten Kamera, also einer Perspektive die sich dynamisch an die Spielsituation anpasst.

[4]WASD bezeichnet eine für Spiele mit Ego-Perspektive häufig voreingestellte Tastaturbelegung. Die Bewegung des Avatars erfolgt dabei mit den Tasten W(vor), A(links), S(zurück) und D(rechts).

In Kampfsituationen ist es möglich charakterabhängige Fähigkeiten zu benutzen. Bei Nahkampfklassen besonders effektive Schläge oder Zauber bei magiebegabten Klassen. Die Benutzung dieser Fähigkeiten bereitete mehreren Probanden Probleme. Für die Benutzung von besonderen Nahkampfattacken muss eine Nahkampfwaffe angelegt sein. Zwei Probanden merkten dies nicht, und versuchten unbewaffnet und erfolglos entsprechende Fähigkeiten zu benutzen. Ein weiterer Proband dachte er hätte die Fähigkeiten benutzt, tatsächlich wurden aber nur automatische Schläge ausgeführt.

Weitere Probleme zeigten sich, sobald die Probanden die Kontrolle über mehr als einen Charakter erhielten. Oft war es nicht eindeutig, ob die gesamte Gruppe ausgewählt ist oder ob die anderen Gruppenmitglieder dem Hauptcharakter automatisch folgen. Ein Proband war mehrfach von einem Auswahlrechteck zur Auswahl mehrerer Charaktere irritiert.

Als Hilfe zum Spieleinstieg wird anfangs zu jedem Spielelement (Dialog, Kampf, Inventar, Charakterbogen usw.) das zum ersten Mal auftaucht oder genutzt wird eine Beschreibung („Tutorial“) eingeblendet. Diese erscheint relativ unscheinbar am linken unteren Bildschirmrand, wodurch viele Probanden diese anfangs nicht gesehen haben. Gleichzeitig blockiert das Auftauchen des Tutorials andere Aktionen. Um das Spiel fortzusetzen muss die Beschreibung weggeklickt werden. Dies störte mehrfach Probanden, gerade wenn an den Beschreibungen kein Interesse bestand. Ein ähnlicher Effekt konnte auch bei dem Eintritt in den Kampfmodus festgestellt werden, da mit dem Beginn des Kampfes standardmäßig automatisch eine Pause aktiviert wird. Der Hinweis auf die Pause ist relativ unscheinbar visualisiert, was dazu führte, dass mehrere Probanden den Hinweis nicht gesehen haben und nicht erklären konnten warum das Spiel nicht voranschreitet. Ein Proband dachte sogar, dass es sich um ein Problem mit dem Testrechner handelt.

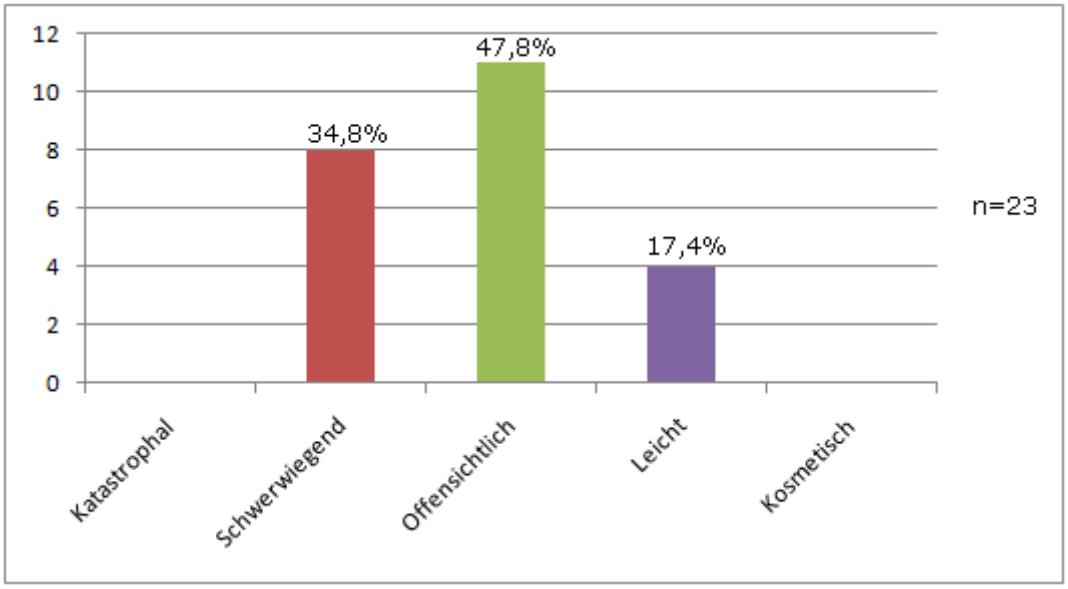

Abb. 12: Problemklassen Drakensang

5.4 Boom Blox

Bei *Boom Blox* wurden insgesamt 12 unterschiedliche Probleme identifiziert. Die geringe Anzahl verdeutlicht, dass die Bedienung grundsätzlich sehr gut funktionierte. Die Verteilung der Probleme auf die Problemklassen weist dabei ein vergleichbares Verhältnis mit den anderen Spielen auf (vgl. Abb. 13: Problemklassen Boom Blox.

Die meisten Probanden starteten mit Schieß-Spielen, bei denen Kisten mit der Wii Remote anvisiert und per Knopfdruck abgeschossen werden. Die Spiele dieser Kategorie sind relativ leicht verständlichen und daher führten Mängel in der Beschreibung anfangs nur zu leichten Problemen. Sobald die Testpersonen jedoch in einen Spielmodus wechselten, der mehr unterschiedliche Aktionen erfordert, wie die Werfen-Spiele oder ein Jenga-ähnlicher Spielmodus, entstanden schwerwiegende Probleme. Zwei Probanden ist nicht bewusst gewesen, wie der Bewegungsablauf für das Werfen eines Gegenstandes ist, und auch nicht ob es eine passende Hilfefunktion gibt. In einem anderen Test wechselten die Probanden in einen Jenga-ähnlichen Spielmodus, erwarteten aber nicht, dass die Kamera manuell positioniert werden kann. Dies führte zu einigen Fehlversuchen und leichter Frustration, da der Schwierigkeitsgrad ohne Anpassungen der Kamera deutlich über dem normalen liegt.

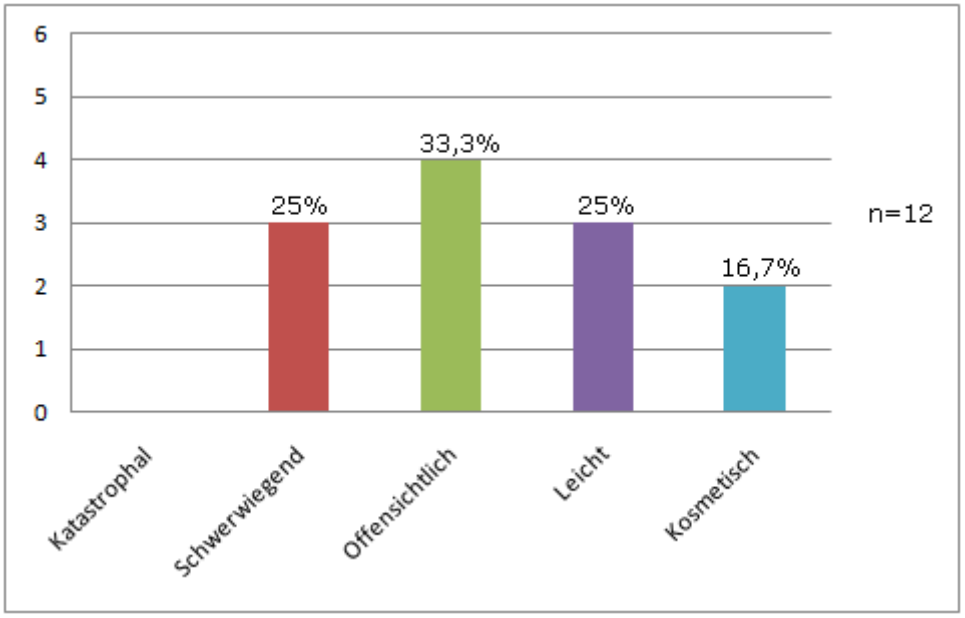

Abb. 13: Problemklassen Boom Blox

5.5 Unterschiede zwischen PC- und Wii-Spielen

Die Auswahl der Spiele (vgl. 3.1) berücksichtigte bewusst ein Spiel für die Nintendo Wii, um abweichende Probleme zwischen PC- und Wii-Spielen zu ermitteln. Signifikante Unterschiede konnten aber nicht identifiziert werden.

Die identifiziert Probleme bei *Boom Blox* sind mit denen der PC-Spiele vergleichbar. Mehrere Bewegungsabläufe waren einigen Probanden zwar nicht intuitiv verständlich, im Grunde stellt dies aber ein Steuerungsproblem dar, wie es z. B. auch bei *Mirror's Edge* festgestellt

werden konnte. Weitere Probleme bei *Boom Blox* waren unverständliche Aufgabenbeschreibungen und Kameraprobleme. Verständnisprobleme finden sich bei allen untersuchten Spielen und Kameraprobleme konnten bei *Drakensang* ebenfalls identifiziert werden. Auch die verhältnismäßig geringe Anzahl von Problemen, ist keine Besonderheit die Eindeutig der Bedienung mit der Wii Remote zugeordnet werden kann. Stattdessen liegt dies an dem Spiel selbst: Da *Boom Blox* ein Casual Game darstellt, gibt es einfach weniger komplexe Spielelemente als bei den anderen getesteten Spielen.

6 Evaluierung der verwendeten Testmethoden

Die verwendeten Testmethoden waren größtenteils bei allen Tests identisch und orientierten sich an den unter 2.2.2 beschriebenen Möglichkeiten des Usability-Tests mit Anwendern. Die Testpersonen wurden beim Spielen beobachtet und Auffälligkeiten wurden direkt notiert. Zusätzlich wurde der Test aufgezeichnet. Im Anschluss an die Spielphase wurde mit den Probanden ein Nachinterview geführt. Eine Variation gab es bei der Anwendung der Methode des lauten Denkens. Bei den Tests für *Anno 1701* und *Drakensang* wurde ein leichtes lautes Denken angewendet, bei *Mirror's Edge* ein retrospektives lautes Denken und bei *Boom Blox* wurde auf diese Methode verzichtet. Im Folgenden werden die Erfahrungen mit den einzelnen Methoden beschrieben.

6.1.1 Beobachtung des Tests

Durch die Beobachtung der Probanden und das direkte Notieren von Auffälligkeiten wurde insgesamt der größte Teil der Probleme identifiziert. Die Beschreibungen waren meistens sehr präzise und im Nachhinein noch gut nachvollziehbar, da die Probleme unmittelbar aufgeschrieben wurden. Durch das aufmerksame Beobachten ist ein Nachdenken über das genaue Geschehen zeitlich kaum möglich. Dies hat den Vorteil, dass keine vorschnelle Fehlinterpretation die Auswertung erschwert oder im schlimmsten Fall die Ergebnisse verfälscht.

Allerdings sind die identifizierten Probleme nicht vollständig. Wenn mehrere Auffälligkeiten zeitnah Auftreten, ist es für einen einzelnen Testleiter schwer alle zu notieren. Der Zeitfaktor, der positiverweise eine Fehlinterpretation verhindert, kann sich in dieser Hinsicht also negativ auswirken. Des Weiteren gibt es Probleme die erst bei genauerer Analyse sichtbar werden. Ein Beispiel hierfür sind Verständnisprobleme, da z. B. bei unklaren Aufgaben oder Visualisierungen viele Probanden alternative Möglichkeiten ausprobieren, die eigentlich nicht sinnvoll sind, auf Grund des Verständnisproblems aber sinnvoll erscheinen. Die ursprüngliche Intention und das eigentliche Problem sind in diesen Fällen durch die Beobachtung alleine oft nicht zu bestimmen.

Im eigentlichen Vorgehen unterscheidet sich die Beobachtung bei Computerspielen nicht mit der bei klassischen Informationssystemen. Bei Computerspielen kann es aber passieren, dass die Frequenz des Auftretens von Auffälligkeiten höher ist. Wie bereits erwähnt, gab es mehrfach Situationen, in denen zeitnah mehrere Probleme auftraten. Dies liegt auch an den Eigenschaften der getesteten Spiele: Meistens ist eine ständige Aufmerksamkeit des Benutzers erforderlich, um auf Aktionen von Gegnern, Mitspielern und der Spielwelt reagieren zu können. Wenn ein Problem auftritt, können dadurch weitere Probleme ausgelöst werden, da auf weitere Aspekte nicht mehr mit der erforderlichen Konzentration reagiert werden kann.

6.1.2 Aufzeichnung des Tests

Die Aufzeichnungen der Tests ermöglichte es auch nach der Testdurchführung weitere Probleme zu identifizieren. Die Anzahl der neuidentifizierten Auffälligkeiten lag deutlich unter

denen der Beobachtung. Viele Probleme die durch die Beobachtung gefunden wurden, konnten aber auch über die Aufzeichnungen gefunden werden.

Insgesamt erwies sich die Aufzeichnung, bzw. die Analyse der Aufzeichnung als eine gute Ergänzung zu der Beobachtung. Es konnten einige Probleme gefunden werden, die bei der Beobachtung übersehen wurden, z. B. weil mehrere Probleme zeitnah aufgetreten sind. Als alleinige Methode wäre die Analyse der Aufzeichnung des Usability-Tests für eine Auswertung aber vermutlich unzureichend, da erklärungsbedürftige Probleme komplett unbeachtet bleiben würden. Durch die Betrachtung der Aufnahme wird zwar ersichtlich, dass Probleme vorliegen, der Kern des Problems kann aber unklar bleiben. Beispiele hierfür sind Verständnisprobleme, die nur von der Testperson selbst beschrieben werden können.

6.1.3 Nachinterview

Im Rahmen der Nachinterviews konnten Erkenntnisse über den Eindruck des Probanden über das jeweilige Testobjekt gewonnen werden und es bot sich die Möglichkeit auf interessante Auffälligkeiten der Spielphase einzugehen.

Die Testpersonen wurden gebeten ihre Meinung über das Testobjekt zu äußern. Daraufhin gaben sie meistens bereitwillig eine ausführliche Auskunft über ihre Einschätzung einzelner Spielaspekte. Es wurden dabei nur vereinzelt neue Probleme identifiziert, die Aussagen der Probanden erwiesen sich aber als nützlich zur Gewichtung von Problemen.

Angesprochen auf interessante Auffälligkeiten aus der Spielphase, waren die Probanden in den meisten Fällen in der Lage sehr präzise zu formulieren, was sie in der jeweiligen Situation vermisst haben oder was sie gestört hat. Durch die Erläuterung des Probanden wurden zwar keine neuen Probleme aufgedeckt, die bereits gefundenen werden aber verständlicher und sind daher besser nachzuvollziehen. Gerade bei Verständnisproblemen die durch die Beobachtung nicht eindeutig zu klären sind, ist dies für die Auswertung der Tests sehr hilfreich. In Kombination mit der Beobachtung konnten daher nützliche Erkenntnisse im Nachinterview gewonnen werden.

Die Nachinterviews wurden nach den in [Nielsen93] beschriebenen Möglichkeiten durchgeführt (vgl. 2.2.2) Da es zu keinen negativen Vorkommnissen kam, sind die hier beschriebenen Erkenntnisse mit den von Nielsen geschilderten Vorteilen identisch. Unterschiede zwischen Computerspielen und klassischen Informationssystemen haben hier also keine Auswirkungen.

6.1.4 Nebenläufiges lautes Denken

Bei den Tests mit den Spielen *Anno 1701* und *Drakensang* wurde nebenläufiges lautes Denken verwendet. Die Probanden wurden im Vorfeld aufgefordert nicht permanent, sondern nur bei Problemen und positiven oder negativen Auffälligkeiten ihre Gedanken zu verbalisieren. Dennoch äußerten sich die Probanden unaufgefordert nur selten, da die Konzentration

auf dem Spielablauf lag. Dem konnte aber durch eine Aufforderung durch den Testleiter entgegengewirkt werden.

Ergänzend zu den beobachteten Auffälligkeiten konnten bei einigen Tests durch das laute Denken neue Probleme identifiziert werden. Diese waren zwar im Verhältnis zur Gesamtanzahl von Problemen nicht besonders zahlreich, dafür aber oft relativ bedeutend. Ein Proband erläuterte z. B. bei *Anno 1701*, dass er den Schieberegler für den Steuersatz eigentlich für die Zufriedenheit der Siedler hielt. Ohne die Anmerkung des Probanden wäre dieses Verständnisproblem unbemerkt geblieben.

Im Vergleich zu klassischen Informationssystemen wurde die übliche Methode des lauten Denkens modifiziert. Beim nebenläufigen lauten Denken sollen die Probanden alle ihre Gedanken über das Testobjekt verbalisieren [vgl. Nielsen93]. Im Rahmen der Tests sollten die Probanden dies nur bei besonderen Auffälligkeiten tun. Dadurch wurde gewährleistet, dass ein möglichst natürlicher Spielablauf gewährleistet ist. Ein vollständiger Verzicht auf lautes Denken hätte weniger Probleme offenbart und daher das Ergebnis der Untersuchung verschlechtert. Die abgeschwächte Variante des nebenläufigen lauten Denkens stellt daher in Verbindung mit dem Ziel einen möglichst natürlichen Spielablauf zu gewährleisten bei den Spielen *Anno 1701* und *Drakensang* einen guten Kompromiss dar.

6.1.5 Retrospektives lautes Denken

Die Methode des retrospektiven lauten Denkens wurde bei den Tests mit *Mirror's Edge* verwendet. Im Anschluss an die Spielphase wurde die Aufnahme des Tests zusammen mit dem Probanden gesichtet. Des Weiteren wurde die Testperson aufgefordert seine Gedanken zu Auffälligkeiten zu äußern.

Durch das retrospektive laute Denken wurden nur wenige Probleme gefunden, die nicht bereits beobachtet wurden. Dadurch war die Besprechung der Auffälligkeiten mit der Besprechung im Rahmen eines Nachinterviews vergleichbar. Dies lag vermutlich an den gegebenen Umständen durch *Mirror's Edge*. Die meiste Zeit befanden sich die Probanden in dem Tutorial des Spiels. Daher waren die Verständnis- und Steuerungsprobleme, die den Großteil der Probleme des Spiels ausmachten, relativ offensichtlich und kaum erklärungsbedürftig.

Auf Basis der durchgeführten Tests fällt eine Beurteilung der Methode des retrospektiven lauten Denkens sehr schwer. Im Rahmen der Tests erwies sie sich als nicht ergiebig, ein Verzicht hätte durch das Nachinterview aller Vorrausicht nach zu dem gleichem Ergebnis geführt. Gleichzeitig wurde aber deutlich, dass die Testdauer für retrospektives lautes Denken zu kurz angesetzt war. Während der Tests kam es daher nur zu sehr wenig erklärungsbedürftigen Problemen.

6.1.6 Multiplayer ohne aufgefordertes lautes Denken

Bei den Tests mit Boom Blox wurde auf eine Aufforderung zum lauten Denken verzichtet. Da es sich um Tests mit jeweils zwei Probanden handelte, wurden aber trotzdem einige Prob-

leme laut ausgesprochen. Dies geschah im Dialog zwischen den beiden Testpersonen, die sich bei Problemen oft gegenseitig geholfen haben. Dies war durchgängig bei allen Tests zu beobachten, unabhängig davon ob ein kooperativer oder konkurrierender Spielmodus gewählt wurde. Dieser Dialog stellte also ein natürliches Spielverhalten im Multiplayer dar. Eine Aufforderung die Gedanken zu bestimmten Elementen des User Interfaces zu verbalisieren, hätte den Dialog zwischen den Probanden, und daher auch den natürlichen Spielablauf, gestört. Gleichzeitig wurden alle auffälligen Probleme und Unklarheiten von den Probanden selbst angesprochen. Zusätzlich boten die Dialoge zwischen den Probanden Ansatzpunkte für Vertiefungen im Rahmen des Nachinterviews. Tatsächlich fand also ein lautes Denken statt, ohne das dazu aufgefordert wurde.

Constructive Interaction, eine Variante des lauten Denkens, ist eine vergleichbare Methode für klassische Informationssysteme (vgl. 2.2.3). Dabei wird das Testobjekt von zwei Probanden zusammen benutzt, um das laute Denken natürlicher erscheinen zu lassen. Der Unterschied zu dem durchgeführten Test liegt in der Nutzungssituation: Bei Desktopanwendungen und Webseiten stellt die Benutzung mit zwei Probanden gleichzeitig normalerweise keine natürliche Benutzungssituation dar. Bei Multiplayer-Casual-Spielen wie *Boom Blox* hingegen schon. Für dieses bestimmte Spiele-Genre wird also eine natürliche Benutzungssituation trotz einer Variante des lauten Denkens erzielt.

7 Evaluierung von Heuristiken

Anhand der Ergebnisse der durchgeführten Usability-Tests wurden Heuristiken hinsichtlich ihrer Eignung für die Evaluierung von Computerspielen untersucht.

Das Vorgehen hierfür ist in drei Phasen unterteilt:

1. Auswahl der zu evaluierenden Heuristiken
2. Analyse der Auswertungsergebnisse
3. Bewertung der ausgewählten Heuristiken

In der ersten Phase wurden veröffentlichte Heuristiken gesammelt und eine Auswahl getroffen, welche näher betrachtet werden sollen.

In der zweiten Phase wurden die Ergebnisse der Usability-Tests betrachtet und allgemeine Problemkategorien identifiziert. Diese bildeten dann die Grundlage für die Formulierung eines Satzes von Heuristiken, der die gefundenen Probleme abdeckt.

In der dritten Phase fand dann die Evaluierung der ausgewählten Heuristiken statt. Hierfür wurde die Überdeckung der Problemkategorien untersucht und ein Vergleich mit den Auswertungsergebnissen und selbst formulierten Heuristiken vorgenommen. Anhand dieser Analyse wurden die Heuristiken bewertet.

7.1 Auswahl der zu evaluierenden Heuristiken

Ein häufig betrachteter Satz von Heuristiken sind die zehn Heuristiken von Nielsen [Nielsen93]. Diese beziehen sich zwar ursprünglich auf Desktop-Anwendungen, sind aber allgemein gehalten und können daher grundsätzlich auf Spiele übertragen werden. Die Bedeutung der Nielsen‘schen Heuristiken zeigt sich dadurch, dass diese in zahlreichen Veröffentlichungen zu spiele-spezifischen Heuristiken als Referenz angegeben sind. Auf eine Darstellung der Heuristiken von Nielsen kann an dieser Stelle jedoch verzichtet werden. Eine ausführliche Besprechung findet sich in [Federoff02]. Demnach sind die Heuristiken von Nielsen hilfreich um das User Interface eines Spieles zu evaluieren, relevante Probleme des Gameplays werden aber nicht abgedeckt. Gameplay-Probleme sind aber für die Zufriedenheit des Benutzers von großer Bedeutung [Vgl. Federoff02]. Diese These deckt sich mit der Definition der Game Usability, die dieser Arbeit zu Grunde gelegt wird (vgl. 2.3). Um Gameplay-Aspekte zu berücksichtigen werden daher im Folgenden nur Sätze von Heuristiken betrachtet, die einen Bezug zu Computerspielen besitzen. Von diesen gibt es bereits eine Vielzahl, wie die Auswahl von betrachteten Veröffentlichungen in Tabelle 9 zeigt.

Heuristiksatz / Autor / Quelle	Beschreibung
Melissa Federoff [Federoff02]	40 Heuristiken; 3 Kategorien: Game Interface, Game Mechanics, Gameplay; Entstanden durch Literaturrecherche und Arbeit mit einem Spiele-Entwickler.

David Pinelle, Nelson Wong [Pinelle081]	10 Heuristiken; Keine Kategorisierung; Formuliert auf Basis einer Analyse von Spiele-Reviews einer Spiele-Website
Heuristics for Usability in Games, Noah Shaffer [Shaffer07]	29 Heuristiken; 3 Kategorien: General, Graphical User Interface, Gameplay; Entwickelt während einer Tätigkeit bei einem Spiele-Entwickler
Heuristics for Evaluating Playability [Desurvire04]	43 Heuristiken; 4 Kategorien: Gameplay, Game Story, Mechanics, Usability; Entstanden durch Evaluierung veröffentlichter Heuristiken, z. B. [Federoff02]
Playability Heuristics for Mobile Games [Korhonen06]	29 Heuristiken; 3 Kategorien: Game Usability, Mobility, Gameplay Enstanden durch Analyse der Nutzungssituationen mobiler Geräte, von Nielsens Heuristiken und von Game Design Richtlinien
Sauli Laitinen [Laitinen08]	25 Heuristiken; 2 Kategorien: Usability, Gameplay Basieren auf den Heuristiken von Nielsen [Nielsen93] und Korhonen [Korhonen06]

Tabelle 9: Sätze von Heuristiken für Computerspiele

Die veröffentlichten Sätze von Heuristiken unterscheiden sich inhaltlich und in der Anzahl der Heuristiken. Zwischen den Veröffentlichungen gibt es aber auch Gemeinsamkeiten und Beziehungen, wodurch diese in drei Kategorien eingeteilt werden können:

1. Game Design-orientierte Heuristiken
2. Erlebnisorientierte Heuristiken
3. Problemorientierte Heuristiken

Die Heuristiken aus [Federoff02] und [Desurvire04] orientieren sich stark am Game Design und bilden daher die Kategorie der *Game Design-orientierten* Heuristiken. In [Federoff02] finden sich z. B. Heuristiken wie „Create a great storyline" und „The game should have an unexpected outcome". Dies sind zweifellos Merkmale eines Spiels die für Spaß sorgen können, zielen aber sehr stark auf Elemente des Game Designs, und weniger auf Game Usability. In [Desurvire04] wurde die Veröffentlichung von Federoff aufgegriffen, um eine neue, verbesserte Liste von Heuristiken zu erzeugen. Neben dem Großteil der Richtlinien aus [Federoff02] finden sich in dieser Liste auch eindeutige Usability-Heuristiken wie „Provide immediate feedback for user actions" und „The interface should be as non-intrusive to the Player as possible".

Die von Korhonen und Koivisto in [Korhonen06] beschriebenen Heuristiken sind im Rahmen eines iterativen Designprozesses für ein Spiel für mobile Geräte entstanden. Daher enthält dieser Satz neben allgemeinen Heuristiken für Usability und Gameplay auch spezifische Heuristiken für mobile Geräte. Laitinen veröffentlichte in [Laitinen08] einen Satz von Heuristiken,

der die Heuristiken aus [Korhonen06] verallgemeinert und mit Gedanken aus [Nielsen93] ergänzt. Diese Heuristiken wurden in zwei Kategorien aufgeteilt: Usability und Gameplay. Usability umfasst hier Heuristiken die sich auf das User Interface und die Bedienung beziehen, wie z. B. „Game controls are convenient and flexible“ und „Use easy-to-understand terminology“, während die Gameplay-Heuristiken Bezug auf das eigentliche Spielerlebnis nehmen, wie z. B. „The first-time experience is encouraging“ , „The player is in control“ und „The game supports different playing styles“. Auf Grund der Berücksichtigung der Gameplay-Aspekte bilden die Heuristiken aus [Korhonen06] und [Laitinen08] die Kategorie der *erlebnisorientierten* Heuristiken.

Die in [Pinelle081] beschrieben Heuristiken wurden ähnlich wie die unter 7.1 formulierten Heuristiken ermittelt. Es wurden Problemkategorien identifiziert, die in Heuristiken überführt wurden. Die Probleme wurden aber nicht aus Usability-Untersuchungen ermittelt, sondern aus Spiele-Reviews einer Spiele-Website. Da die Heuristiken auf Basis von vorhandenen Problemen formuliert wurden, bilden sie die Kategorie der *problemorientierten* Heuristiken.

Auf Grund der Unterschiede in der Entstehung und im Inhalt zwischen den Kategorien ist anzunehmen, dass die Heuristiken sich auch in ihrer Eignung zum Aufdecken von Problemen unterscheiden. Daher wurden alle Kategorien berücksichtigt und jeweils ein Vertreter ausgewählt. Da innerhalb der Game Design-orientierten und erlebnisorientierten Heuristiken bereits eine Art von Evaluierung stattfand, fiel hier die Wahl auf die jeweils aktuellste der betrachteten Veröffentlichungen. Diese sind [Desurvire04] und [Laitinen08]. Vervollständigt wird die Auswahl durch [Pinelle081].

Kategorie	**Vertreter** (fett: Auswahl für weitere Evaluierung)
Game Design-orientiert	[Federoff02], **[Desurvire04]**
Erlebnisorientiert	[Korhonen06], [Shaffer07], **[Laitinen08]**
Problemorientiert	**[Pinelle081]**, (7.2.2)

Tabelle 10: Kategorien und Auswahl von Heuristiksätzen

7.2 Analyse der Auswertungsergebnisse

Die Auswertungsergebnisse wurden analysiert, um diese in eine für die Evaluierung der ausgewählten Heuristiken geeignete Form zu überführen. Hierfür wurden Problemkategorien identifiziert und eigene Heuristiken formuliert.

7.2.1 Problemkategorien

Zur Bestimmung der Problemkategorien wurden Gemeinsamkeiten zwischen den identifizierten Problemen der Spiele gesammelt und gruppiert. Dadurch wurden zehn Problemkategorien identifiziert:

1. **Unverständliche Aufgaben und Spielziele**: Bei *Anno 1701* war mehreren Probanden unklar, wie genau die nächsten Entwicklungsstufen der Siedlungen erreicht werden können. Bei *Drakensang* und *Mirror’s Edge* waren sich mehrere Probanden am An-

fang des Spiels nicht sicher, was sie genau zu tun haben. Und sogar bei *Boom Blox* fragte ein Proband, was das Ziel der ganzen kleinen Spiele eigentlich sei.

2. **Kontrollverlust**: Bei den getesteten PC-Spielen gibt es Stellen im Spielablauf, an denen dem Spieler das Gefühl genommen wird, das Spiel zu kontrollieren. Beispiele sind die eingeschränkten Aktionsmöglichkeiten während des Tutorials bei *Anno 1701*, die Blockierung des restlichen User Interfaces bei Erscheinen einer Tutorial-Beschreibung bei *Drakensang* oder Inhalte die wiederholt und nicht übersprungen werden können bei *Mirror's Edge*.
3. **Steuerung**: Probleme mit der Steuerung wurden bei allen untersuchten Spielen festgestellt. Bei *Boom Blox* war den Probanden mehrfach unklar wie die Bewegungen genau ausgeführt werden müssen. Bei *Mirror's Edge* gab es Probleme mit der Tastaturbelegung. Bei Spielern ohne Genre-spezifische Vorkenntnisse kam es zusätzlich zu Problemen mit der Bewegung des Avatars. Bei *Anno 1701* funktionierte das Anlegen von Wegen anders als von vielen Probanden erwartet. Beim Bauen von Gebäuden wurden mehrfach auch unterstützende Funktionen vermisst. Bei *Drakensang* bereitete die Steuerung während eines Kampfes bei allen Probanden Probleme.
4. **Unkomfortable Kameraführung:** Bei *Drakensang* und *Boom Blox* war es oftmals nötig, die Kamera selbst neu zu positionieren. Bei *Boom Blox* ist die Bedienung der Kamera intuitiv über die Bewegung der Wii Remote möglich, wodurch lediglich die ungeübten Probanden eine Eingewöhnungsphase benötigten. Kritischer stellte sich das Kameraproblem bei *Drakensang* dar. Ein Proband dachte z. B. auf Grund des Kamerawinkels, dass seine Bewegungsfreiheit in der Spielwelt eingeschränkt sei. Ein anderer Proband vermisste die Einstellungsmöglichkeit einer „intelligenten" Kamera. Unzufrieden mit der Kamerabedienung bei *Drakensang* waren alle Probanden.
5. **Inkonsistente und unverständliche Visualisierungen**: Bei allen untersuchten Spielen fanden sich Probleme, die auf eine unverständliche Darstellung zurückzuführen sind. Mehrfach waren verwendete Symbole oder Icons unverständlich. Zwei Probanden brauchten bei Boom Blox sehr lange für die Auswahl des passenden Spielmodus, da die Symbole im Menü nicht verstanden wurden. Bei *Anno 1701* traten vergleichbare Probleme mit den Menüs innerhalb des Spiels auf. Bei *Drakensang* wurde die Minimap[5] von zwei Probanden nicht verstanden, teilweise auch auf Grund unverständlicher Symbole. Bei *Mirror's Edge* ist die Momentum[6]-Anzeige einem Probanden zu klein gewesen, und wurde daher nicht als Spielinformation wahrgenommen.
6. **Inkonsistentes Systemverhalten**: Bei *Mirror's Edge* ist bei einem Zwischenschritt im Tutorial keine Runner Vision [7]vorhanden, obwohl bei allen anderen Schritten eine entsprechende Markierung vorgenommen wurde. Bei *Anno 1701* fehlten an einigen Stellen Tooltips auf Steuerelementen, obwohl bei vergleichbaren Elementen Tooltips

[5]Minimap: Eine permanent sichtbare Umgebungskarte, auf der das umliegende Gelände und Symbole für Verbündete, Gegner und Zielpunkte dargestellt werden.
[6]Momentum: Ein Spielelement, das präzisere Bewegungen und Spezialbewegungen ermöglicht.
[7]Runner Vision: Eine Hilfe zur Wegfindung, die relevante Objekte rot eingefärbt.

eingesetzt wurden. Bei Drakensang war zudem das Verhalten von Gruppenmitgliedern nicht nachvollziehbar: Ob ein Gruppenmitglied automatisch folgt oder nicht, war mehreren Probanden nicht ersichtlich.

7. **Unzureichende Anleitung und Hilfe**: Bei *Anno 1701* wurde mehrfach versucht die Hilfe des Spiels zu benutzen, z. B. bei nicht verstandenen Spielmechanismen. In allen Fällen konnten aber keine hilfreichen Informationen gefunden werden. Bei Boom Blox versuchten zwei Probanden erfolglos Hilfen zur Steuerung während des Spiels zu bekommen. Eine situationsabhängige Hilfe existiert bei *Boom Blox* aber nicht.
8. **Unverständliche Beschreibungen und Anweisungen:** Bei *Mirror's Edge* sind einige Beschreibungen im Tutorial unverständlich, da unerklärte Begriffe und unklare Tastenbezeichnungen verwendet werden. Bei *Anno 1701* waren mehrere gesprochenen Anweisungen im Tutorial für einen Probanden unverständlich. Bei *Boom Blox* sind die Beschreibungen, die jedem Spiel vorausgehen, von den Probanden mehrfach nicht verstanden worden.
9. **Unverständliche, unübersichtliche Menüs und Steuerelemente:** Bei *Anno 1701* und *Boom Blox* gab es bereits vor dem eigentlichen Spiel Probleme, da die Menüs nicht für alle Probanden verständlich waren. Zusätzlich erwiesen sich einige Steuerelementen und Menüs innerhalb des eigentlichen Spiels bei als problematisch. Bei Anno *1701* war mehreren Probanden nicht verständlich wie Handel funktioniert oder ein Schiff beladen wird, obwohl das entsprechende Menü gefunden wurde.
10. **Vorausgesetzte Genre-Standards**: Grundsätzlich sind Genre-Standards sinnvoll und gerade geübten Spielern wird der Einstieg dadurch erleitert. Bei *Mirror's Edge* und *Drakensang* wurden die Genre-Standards aber teilweise nicht erläutert. Da bei *Mirror's Edge* die für Spiele mit Ego-Perspektive übliche WASD-Steuerung verwendet wird, wurde also die fundamentale Steuerung nicht erläutert. Dies führte bei allen Probanden ohne spezifische Vorkenntnisse zu einem schwerfälligen, teilweise frustrierenden, Einstieg in das Spiel. Bei *Drakensang* wurde das Quest-System nicht näher erläutert. Ein Proband dachte er könnte nur ein Quest zur gleichen Zeit annehmen, wodurch sein Spielfortschritt unnötig verzögert wurde.

Der Schwerpunkt der identifizierten Problemkategorien liegt auf der Steuerung, dem User Interface und der Hilfe. Gameplay-Aspekte werden durch Kategorie 1, 2 und 6 lediglich angedeutet. Es ist aber anzunehmen, dass bei einer längeren Testdauer mehr Probleme auftreten die im Gameplay begründet sind. Anzeichen hierfür lieferten die Tests mit den Probanden die eine hohe Spiele-Affinität und Vorkenntnisse des jeweiligen Genres aufwiesen. Wenn Gameplay-Probleme identifiziert wurden, dann mit dieser Probandengruppe. Der Grund hierfür ist, dass bei Spielern mit spezifischen Vorkenntnissen weniger grundlegende Probleme auftraten und sich die wahrgenommenen Probleme in den Gameplay-Bereich verlagerten.

Nr	Beschreibung	Beispielprobleme
1	Unverständliche Aufgaben und Spielziele	Unklare Aufgabenbeschreibung (Drakensang), Unklare Bedingungen für die Entwicklung der Siedlung (Anno 1701)
2	Kontrollverlust	Überspringen von sich wiederholendem Content nicht möglich (Mirror's Edge), Blockierung oder Einschränkung der Nutzungsmöglichkeiten des User Interfaces bis eine bestimmte Aktion ausgeführt wird (Drakensang, Anno 1701)
3	Steuerung	Unerwartete Tastaturbelegung (Drakensang), Steuerung zum Werfen unklar (Boom Blox), Spieler ist nicht in der Lage Fähigkeiten des Avatars anzuwenden (Drakensang)
4	Unkomfortable Kameraführung	Kameraanpassung erforderlich (Drakensang)
5	Inkonsistente und unverständliche Visualisierungen	Unverständliche Symbole (Anno 1701, Drakensang), Unterschiedlicher Grafikstil für Icons (Anno 1701), unverständliche Minimap (Drakensang)
6	Inkonsistentes Systemverhalten	Ein Tutorialschritt ohne „Runner Vision" (Mirror's Edge), Fehlende Tooltips (Anno 1701), Keine alternierende Spielerreihenfolge (Boom Blox), Gruppenmitglied folgt? (Drakensang)
7	Unzureichende Anleitung und Hilfe	Grundlagensteuerung ohne Erklärung (Mirror's Edge), Keine Hilfe für komplexe Spielzusammenhänge (Anno 1701)
8	Unverständliche Beschreibungen und Anweisungen	Unverständliche textuelle Beschreibungen (Mirror's Edge, Drakensang, Boom Blox), unverständliche gesprochene Anweisungen (Anno 1701)
9	Unverständliche, unübersichtliche Menüs und Steuerelemente	Unverständliche Ingame-Menüs (Anno 1701), Gesuchte Tastaturbelegung nicht gesehen (Mirror's Edge)
10	Vorausgesetzte Genre-Standards	WASD-Problem (Mirror's Edge), automatische Kampfpause (Drakensang)

Tabelle 11: Problemkategorien

7.2.2 Formulierung von Heuristiken

Auf Basis der unter 7.2.1 identifizierten Problemkategorien, wurden von mir Heuristiken bestimmt. Hierfür wurden die negativ formulierten Problembeschreibungen in positive Formulierungen umgewandelt. Vielfach entspricht dies einer eins-zu-eins-Beziehung, lediglich die Problemkategorien 5 und 9 wurden auf Grund ihrer Ähnlichkeit zu einer Heuristik zusammengefasst. Die erzeugte Liste von Heuristiken zeigt Tabelle 12. Es werden alle zehn Problemkategorien überdeckt. Diese Heuristiken sind also vollständig bezüglich der Testergebnisse, und bieten daher eine Orientierung zur Bewertung der ausgewählten Heuristiken. Gleichzeitig sind die zu erwartenden Gameplay-Probleme, bei einer längeren Testdauer, nicht abgedeckt.

Nr	Heuristik	Problem-kategorie
1	Verständliche Aufgabenstellungen und Spielziele	1
2	Der Spieler wird nicht vom Spielen abgehalten	2

3	Angemessene und intuitive Steuerung	3
4	Angemessene Kameraführung	4
5	Konsistente und verständliche Darstellungen von Symbolen, Steuerelementen und Menüs	5, 9
6	Konsistentes und nachvollziehbares Verhalten des Systems und nicht spielergesteuerter Einheiten	6
7	Angemessene Anleitungen und Hilfe	7
8	Verständliche Sprache und einfache Formulierungen in Anleitungen, Beschreibungen und in Menüs	8
9	Verwende Genre-Standards, aber erkläre sie unerfahrenen Spielern	10

Tabelle 12: Heuristiken

7.3 Bewertung der ausgewählten Heuristiken

Um die ausgewählten Heuristiksätze zu evaluieren, wurden diese analysiert und mit den von mir entwickelten Problemkategorien abgeglichen. Da die Kategorien auf eindeutig identifizierten Problemen beruhen, ist eine vollständige Überdeckung dieser anzustreben. Zusätzlich ist es wichtig, dass auch Gameplay-Aspekte berücksichtigt werden, diese sind in den Problemkategorien nur angedeutet (vgl. 7.2.1). Ein Satz von Heuristiken sollte also

1. alle Problemkategorien abdecken,
2. zusätzliche Gameplayaspekte enthalten.

Im Folgenden wird die Bewertung der ausgewählten Sätze von Heuristiken dargestellt.

7.3.1 Game Design-orientierte Heuristiken

Die Heuristiken von Desurvire sind im Vergleich zu den anderen betrachteten Heuristiksätzen sehr zahlreich (Tabelle 13), decken aber trotzdem nur sieben der zehn Problemkategorien teilweise ab.

Nr	Heuristik
Gameplay	
GP1	Player's fatigue is minimized by varying activities and pacing during game play.
GP2	Provide consistency between the game elements and the overarching setting and story to suspend disbelief.
GP3	Provide clear goals, present overriding goal early as well as short-term goals throughout play.
GP4	There is an interesting and absorbing tutorial that mimics game play.
GP5	The game is enjoyable to replay.
GP6	Game play should be balanced with multiple ways to win.
GP7	Player is taught skills early that you expect the players to use later, or right before the new skill is needed.
GP8	Players discover the story as part of game play.
GP9	Even if the game cannot be modeless, it should be perceived as modeless.
GP10	The game is fun for the Player first, the designer second and the computer third. That is, if

	the non-expert player's experience isn't put first, excellent game mechanics and graphics programming triumphs are meaningless.
GP11	Player should not experience being penalized repetitively for the same failure.
GP12	Player's should perceive a sense of control and impact onto the game world. The game world reacts to the player and remembers their passage through it. Changes the player makes in the game world are persistent and noticeable if they back-track to where they've been before.
GP13	The first player action is painfully obvious and should result in immediate positive feedback.
GP14	The game should give rewards that immerse the player more deeply in the game by increasing their capabilities (power-up), and expanding their ability to customize.
GP15	Pace the game to apply pressure but not frustrate the player. Vary the difficulty level so that the player has greater challenge as they develop mastery. Easy to learn, hard to master.
GP16	Challenges are positive game experiences, rather than a negative experience (results in their wanting to play more, rather than quitting).
Game Story	
GS1	Player understands the story line as a single consistent vision.
GS2	Player is interested in the story line. The story experience relates to their real life and grabs their interest.
GS3	The Player spends time thinking about possible story outcomes.
GS4	The Player feels as though the world is going on whether their character is there or not.
GS5	The Player has a sense of control over their character and is able to use tactics and strategies.
GS6	Player experiences fairness of outcomes.
GS7	The game transports the player into a level of personal involvement emotionally (e.g., scare, threat, thrill, reward, punishment) and viscerally (e.g., sounds of environment).
GS8	Player is interested in the characters because (1) they are like me; (2) they are interesting to me, (3) the characters develop as action occurs.
Mechanics	
M1	Game should react in a consistent, challenging, and exciting way to the player's actions (e.g., appropriate music with the action).
M2	Make effects of the Artificial Intelligence (AI) clearly visible to the player by ensuring they are consistent with the player's reasonable expectations of the AI actor.
M3	A player should always be able to identify their score/status and goal in the game.
M4	Mechanics/controller actions have consistently mapped and learnable responses.
M5	Shorten the learning curve by following the trends set by the gaming industry to meet user's expectations.
M6	Controls should be intuitive, and mapped in a natural way; they should be customizable and default to industry standard settings.
M7	Player should be given controls that are basic enough to learn quickly yet expandable for advanced options.
Usability	
U1	Provide immediate feedback for user actions.
U2	The Player can easily turn the game off and on, and be able to save games in different states.
U3	The Player experiences the user interface as consistent (in control, color, typography, and dialog design) but the gameplay is varied.
U4	The Player should experience the menu as a part of the game.
U5	Upon initially turning the game on the Player has enough information to get started to play.
U6	Players should be given context sensitive help while playing so that they do not get stuck or have to rely on a manual.

U7	Sounds from the game provide meaningful feedback or stir a particular emotion.
U8	Players do not need to use a manual to play game.
U9	The interface should be as non-intrusive to the Player as possible.
U10	Make the menu layers well-organized and minimalist to the extent the menu options are intuitive.
U11	Get the player involved quickly and easily with tutorials and/or progressive or adjustable difficulty levels.
U12	Art should be recognizable to player, and speak to its function.

Tabelle 13: Game Design-orientierte Heuristiken [Desurvire04]

Generell fällt es schwer die Heuristiken auf die identifizierten Problemkategorien zu übertragen. Gründe hierfür sind die Anzahl der Heuristiken, wodurch sich einige Problemkategorien auf mehrere Heuristiken verteilen. Zusätzlich sind die Heuristiken teilweise zu allgemeine und teilweise zu spezifisch formuliert. „ The game is enjoyable to replay “ ist z. B. eine sehr allgemeine Heuristik, der vermutlich niemand wiedersprechen wird, die gleichzeitig aber schwer zu überprüfen ist. „There is an interesting and absorbing tutorial that mimics game play” ist wiederum sehr spezifisch. Diese Heuristik bestätigte sich in den Tests mit Mirror’s Edge, gleichzeitig ist sie für andere Spiele nicht zutreffend, da z. B. Spiele wie Tetris kein Tutorial benötigen.

Der Bereich des User Interfaces und die Verständlichkeit von Beschreibungen oder Visualisierungen wird im Gegensatz zu den anderen Heuristiksätzen oft nur angedeutet. „Make the menu layers well-organized and minimalist to the extent the menu options are intuitive” bietet z. B. ein Gestaltungshinweis, welche Probleme bei Menüs im Spiel auftreten, im Rahmen der Tests waren es hauptsächlich Verständnisprobleme, lässt sich nur erahnen.

Die Gameplay-Probleme die in den identifizierten Problemkategorien enthalten sind, wie ein unklares Spielziel oder unklare Handlungsmöglichkeiten, werden nicht nur vollständig abgedeckt, sondern es lassen sich noch einige weitere Ansätze in diesem Bereich finden.

Die Herkunft der Heuristiken aus dem Bereich des Game Designs macht sich stark bemerkbar. Einige grundlegende Usability-Probleme lassen sich in den Heuristiken nicht wiederfinden. Mehrere Problemkategorien sind zudem nur durch viel Transferaufwand abgedeckt. Verstärkt wird der Aufwand noch, da ähnliche Probleme auf mehrere Heuristiken verteilt sind und viele Heuristiken mehr Hinweise für den Entwurf des Spiels als für die Evaluierung der Usability bieten.

7.3.2 Erlebnisorientierte Heuristiken

Die Heuristiken von Laitinen, die ausgewählten erlebnisorientierten Heuristiken, decken acht der zehn Problemkategorien vollständig und eine Problemkategorie teilweise ab.

Nr	Heuristik
Usability	
U1	Consistency
U2	Provide feedback

U3	Use easy-to-understand terminology
U4	Minimize player's memory load
U5	Avoid errors
U6	Provide help
U7	Simple and clear menus
U8	Device user interface and game user interface are used for their own purposes
U9	Screen layout is efficient and visually pleasing
U10	Audiovisual representation supports the game
U11	Game controls are convenient and flexible
Gameplay	
GP1	The game provides clear goals or supports player-created goals
GP2	The player sees the progress in the game and can compare the results
GP3	The player is rewarded and rewards are meaningful
GP4	The player is in control
GP5	Challenge, strategy and pace are balanced
GP6	The first-time experience is encouraging
GP7	The game story supports the gameplay and is meaningful
GP8	There are no repetitive or boring tasks
GP9	The game supports different playing styles
GP10	The game does not stagnate
GP11	The game is consistent
GP12	The game uses orthogonal unit differentiation
GP13	The player does not lose any hard-won possessions
GP14	The players can express themselves

Tabelle 14: Erlebnisorientierte Heuristiken [Laitinen08]

Die Probleme durch die vorausgesetzten Genre-Standards lassen sich nicht vollständig wiederfinden. „The first-time experience is encouraging" entspricht aber in gewisser Weise dem Problem bei Mirror's Edge, dass die Standard-Tastaturbelegung nicht erklärt wurde. Diese entsprach zwar dem Genre-Standard, entmutigte aber zwei Probanden direkt beim Spieleinstieg, die den Standard nicht kannten. Das Gegenteil der Heuristik ist also der Fall.

Keine Erwähnung finden Probleme auf Grund einer ungünstigen Kameraposition oder Perspektive auf die Spielwelt. Diese Problemkategorie trat bei den Spielen Drakensang und Boom Blox auf, also 3D-Spiele die eine manuelle Positionierung der Kamera ermöglichen. Die Grundlage für die Heuristiken von Laitinen sind die „Playability Heuristics for Mobile Games" aus [Korhonen06]. Spiele für mobile Geräte bieten diese Möglichkeiten auf Grund ihrer im Vergleich zu PC-Spielen geringen Komplexität üblicherweise nicht. Die Herkunft aus dem mobilen Bereich wird auch an der Heuristik „Device user interface and game user interface are used for their own purposes" deutlich. Bei den getesteten Spielen gab es diesbezüglich keine Probleme, da die Heuristik von allen Spielen eingehalten wurde.

Trotz der erkennbaren Herkunft aus dem mobilen Bereich sind die Heuristiken von Laitinen sehr umfassend, bis auf die angesprochene Kameraproblematik finden sich alle Problemkategorien wieder. Die Bereiche Steuerung, Hilfe, verständliche Darstellung, Kontrolle des Spielers und klare Spielziele lassen sich wiederfinden. Die angedeuteten Gameplay-Aspekte wer-

den also auch abgedeckt. Zusätzlich werden einige Gameplay-Heuristiken genannt, die Ansatzpunkte für weitere Problemkategorien bieten.

7.3.3 Problemorientierte Heuristiken

Der Satz von problemorientierten Heuristiken aus [Pinelle081] (Tabelle 15) deckt von den unter 7.2.1 identifizierten zehn Problemkategorien sieben vollständig und zwei ansatzweise ab.

Nr	Heuristik
1	Provide consistent responses to the user's actions
2	Allow users to customize video and audio settings, difficulty and game speed.
3	Provide predictable and reasonable behavior for computer controlled units.
4	Provide unobstructed views that are appropriate for the user's current actions.
5	Allow users to skip non-playable and frequently repeated content.
6	Provide intuitive and customizable input mappings.
7	Provide controls that are easy to manage, and that have an appropriate level of sensitivity and responsiveness.
8	Provide users with information on game status.
9	Provide instructions, training, and help.
10	Provide visual representations that are easy to interpret and that minimize the need for micromanagement.

Tabelle 15: Problemorientierte Heuristiken [Pinelle081]

„Provide users with information on game status" geht zwar in eine ähnliche Richtung wie „Unverständliche Aufgaben und Spielziele", ein nicht verstandenes Meta-Ziel, ein Problem das bei Anno 1701 auftrat, wird hierdurch aber nicht abgedeckt. „Inkonsistentes Systemverhalten" wird ebenfalls nur teilweise abgedeckt. „Provide predictable and reasonable behavior for computer controlled units" und „Provide consistent responses to the user's action" sind wichtige Unterpunkt dieser Problemkategorie, auf offensichtliche Inkonsistenzen im Verhalten des User Interfaces, wie erwartete aber nicht vorhandene Tooltips, geht aber keine Heuristik ein.

Die Probleme mit vorausgesetzten Genre-Standards die aber nicht erklärt werden, lassen sich in keiner Heuristik wiederfinden. Auf Grund der in [Pinelle081] verwendeten Datenbasis ist das wenig überraschend: Spiele-Reviews werden üblicherweise nicht von ungeübten Spielern geschrieben.

Zwischen den Heuristiken von Pinelle und den unter 7.2.2 formulierten Heuristiken lassen sich einige Gemeinsamkeiten feststellen. Eine angemessene Steuerung, Kamera, verständliche Visualisierungen und ein Bedarf an Anleitung und Hilfe lassen sich in beiden Heuristiksätzen wiederfinden. Gleichzeitig sind die Heuristiken von Pinelle aber weniger umfassend, und schließen Gameplay-Aspekte nahezu komplett aus. Die Kritik an den Heuristiken unter 7.2.2 lässt sich daher auf die von Pinelle übertragen: Zu erwartende Usability-Probleme die im Gameplay begründet sind, werden nicht beachtet.

7.4 Ergebnis der Evaluierung

Die Game Design-orientierten Heuristiken sind für eine heuristische Evaluierung der Game Usability am wenigsten geeignet, da sich Usability-Probleme in ihnen am schwersten wiederfinden lassen. Auch die große Anzahl an Heuristiken ist nachteilig, da dadurch die Quintessenz der einzelnen Heuristiken vernachlässigt wurde.

Die betrachteten problemorientierten Heuristiken sind sehr allgemein gehalten, aber bieten trotzdem verständliche Hinweise auf mögliche Probleme, die sich auch in den durchgeführten Tests wiederfanden. Es werden aber nicht alle entdeckten Probleme abgedeckt. Zusätzlich sind Gameplay-Aspekte fast völlig unbeachtet. Daher sind die unter 7.2.2 formulierten Heuristiken umfassender. Wenn eine der beiden Heuristik-Listen als Grundlage für eine Evaluierung dienen soll, sollte in jedem Fall vorher eine Erweiterung vorgenommen werden.

Die erlebnisorientierten Heuristiken von Laitinen erzielen mit acht von zehn eindeutig abgedeckten Problemkategorien den höchsten Überdeckungsgrad. Auch die Auflistung verständlicher Gameplay-Heuristiken ist ein positiver Aspekt dieser Liste, auch wenn nur wenige bestätigt wurden. Der Grund hierfür ist die geringe Testdauer, die schon bei den Problemkategorien und eigenen Heuristiken zu Einschränkungen führte (siehe 7.2).

Zusammenfassend lässt sich feststellen, dass keine der betrachteten Listen von Heuristiken für die Untersuchung von Game Usability perfekt geeignet ist. Eine vollständige Überdeckung der Problemkategorien kann kein betrachteter Satz von Heuristiken sicherstellen. Die Liste von Laitinen macht jedoch insgesamt den reifsten Eindruck, da der höchste Überdeckungsgrad erzielt wird und zusätzlich noch verständliche Gameplay-Heuristiken genannt werden. Um einen Satz von Heuristiken zu erstellen, der die unter 7.3 genannten Anforderungen vollständig erfüllt, bietet sich eine Erweiterung der Heuristiken von Laitinen an. Dieser Schritt wird im folgenden Kapitel dargestellt.

8 Erstellen eines erweiterten Satzes von Heuristiken

Die unter 7.2.2 selbstformulierten Heuristiken wurden auf Basis der Problemkategorien erstellt und decken diese daher 100%ig ab. Der Heuristiksatz von Laitinen enthält vierzehn Gameplay-Heuristiken, von denen einzelne im Rahmen der Tests bereits bestätigt wurden. Eine Kombination dieser beiden Heuristiksätze erfüllt daher die an einen Satz von Heuristiken gestellten Anforderungen (vgl. 7.3).

Zur Konstruktion des neuen Heuristiksatzes wurde die Liste von Laitinen als Basis genommen, da diese mehr Heuristiken enthält als die selbstformulierten Heuristiken. Anschließend wurden die fehlenden Heuristiken bzw. Problemkategorien ergänzt. Insgesamt wurden drei Heuristiken hinzugefügt und eine modifiziert. Die hinzugefügten Heuristiken sind U12 („Appropriate perspectives“), U13 („Explain genre standards and conventions“), und U14 („Allow the player to skip non-playable content and tutorials“). Modifiziert wurde noch U6. Die ursprüngliche Formulierung „Provide help” wurde zu „Provide training and help” geändert. Hiermit soll verdeutlicht werden, dass Hilfe mehr sein kann als eine Dokumentation von Spielelementen. Tutorial- oder Trainingsphasen sind nicht grundsätzlich nötig, können aber sehr hilfreich sein. Während der Tests mit Mirror’s Edge wurde z. B. von allen Spielern das Tutorial gespielt, um nicht unvorbereitet in das Spiel zu starten. Auch von den Spielern mit spezifischen Vorkenntnissen. Tabelle 16 zeigt die neue Liste von Heuristiken.

Nr	**Heuristik (***kursiv: hinzugefügt/modifiziert***)**
Usability	
U1	Consistency
U2	Provide feedback
U3	Use easy-to-understand terminology
U4	Minimize player’s memory load
U5	Avoid errors
U6	*Provide training and help*
U7	Simple and clear menus
U8	Device user interface and game user interface are used for their own purposes
U9	Screen layout is efficient and visually pleasing
U10	Audiovisual representation supports the game
U11	Game controls are convenient and flexible
U12	*Appropriate perspectives*
U13	*Explain genre standards and conventions*
U14	*Allow the player to skip non-playable content and tutorials*
Gameplay	
GP1	The game provides clear goals or supports player-created goals
GP2	The player sees the progress in the game and can compare the results
GP3	The player is rewarded and rewards are meaningful
GP4	The player is in control
GP5	Challenge, strategy and pace are balanced

GP6	The first-time experience is encouraging
GP7	The game story supports the gameplay and is meaningful
GP8	There are no repetitive or boring tasks
GP9	The game supports different playing styles
GP10	The game does not stagnate
GP11	The game is consistent
GP12	The game uses orthogonal unit differentiation
GP13	The player does not lose any hard-won possessions
GP14	The players can express themselves

Tabelle 16: Erweiterte Liste von Heuristiken

9 Zusammenfassung und Ausblick

9.1 Zusammenfassung

Im Rahmen dieser Arbeit wurden Usability-Tests mit Computerspielen durchgeführt um Usability-Probleme von Spielen festzustellen, eine Einschätzung über die Eignung der verwendeten Testmethoden zu geben und eine Evaluierung von veröffentlichten Game Usability Heuristiken vorzunehmen.

Bei der Durchführung der Tests und der Analyse der erhaltenen Daten zeigte sich, dass die klassischen Usability-Testmethoden auf Computerspiele übertragen werden können. Durch das Zusammenspiel von Beobachtung, Videoaufzeichnung, lauten Denken und Nachinterview konnte eine beträchtliche Menge von Usability-Problemen identifiziert werden. Die identifizierten Usability-Probleme lagen in den Bereichen von Computerspielen, die unter 2.3 der Game Usability zugeschrieben wurden, also im User Interface und im Gameplay.

Anhand der identifizierten Probleme wurden allgemeine Problemkategorien und Heuristiken formuliert. Diese dienten neben der Erfahrung aus den durchgeführten Tests als Grundlage für die Evaluierung von veröffentlichten Sätzen von Heuristiken der Game Usability. Hierbei zeigte sich, dass keine der untersuchten Heuristiksätze alle geforderten Anforderungen erfüllen konnte, die Liste von Laitinen aber bereits eine hohe Qualität aufweist.

Die Evaluierung der Heuristiken zeigte aber, dass die Heuristiken von Laitinen sich mit den selbstformulierten Heuristiken ergänzen, da die jeweiligen Stärken die Schwächen der anderen Liste auffangen. Eine Kombination dieser beiden Listen stellt daher eine logische Konsequenz dar, die abschließend vorgenommen wurde. Hierfür wurde ein Heuristiksatz erstellt, der die Heuristiken von Laitinen um die fehlenden Aspekte aus den Problemkategorien erweitert.

9.2 Ausblick

In Kapitel 8 wurde eine erweiterte Liste von Heuristiken formuliert. Diese Liste bietet mehrere Ansatzpunkte für weitere Untersuchungen, wie

- die Erweiterung und Überarbeitung der Liste durch weiterführende Untersuchungen,
- die Erprobung der Praxistauglichkeit anhand von heuristischen Evaluationen auf Basis dieser Liste,
- die Spezifizierung dieser Liste für spezielle Genres anhand einer weitergehenden Analyse der entsprechenden Spiele-Genres und
- die Anpassung der Heuristiken für spezielle Zielgruppen wie Rentner oder Kinder.

Die Untersuchung der Testmethoden zeigte, dass die betrachteten Usability-Methoden auf Spiele übertragbar sind. Mögliche Anknüpfpunkte an dieser Stelle sind

- die Betrachtung von speziellen Zielgruppen wie Rentner und Kinder,
- die Erprobung der Möglichkeit des Eye-Tracking[8] in Verbindung mit Computerspielen,
- und weiterführenden Untersuchungsmethoden im Allgemeinen, wie z. B. einem Neural Impulse Actuator[9].

[8]Eye-Tracking bezeichnet die Aufzeichnung der Blickbewegungen einer Person

[9]Ein Neural Impulse Actuator misst über ein Stirnband Impulse der Stirn- und Augenmuskulatur und wandelt diese in Eingabesignale um

Literaturverzeichnis

[Adams06]	Adams, Ernest; Rollings, Andrew: *Fundamentals of Game Design*. Pearson, 2006
[Collins97]	Collins, Jeanne: *Conducting In-house Play Testing*, 7.7.1997 http://www.gamasutra.com/view/feature/3211/conducting_inhouse_play_testing.php, Abruf: 5.7.09
[Desurvire04]	Desurvire, Heather; Caplan, Martin; Toth, Jozsef A.: *Using Heuristics to Evaluate the Playability of Games*, 2004
[Federoff02]	Federoff, Melissa A.: *Heuristics and Usability Guidelines for the Creation and Evaluation of Fun in Video Games*. http://melissafederoff.com/heuristics_usability_games.pdf, 2002
[Hoonhout08]	Hoonhout, Henriette: *Let the Game Tester Do the Talking: Think Aloud and Interviewing to Learn About the Game Experience.* In: Isbister, Katherine; Schaffner, Noah: *Game Usability*, Morgan Kaufmann, 2008, S. 65-77
[Korhonen06]	Hannu Korhonen, Elina M.I. Koivisto: *Playability Heuristics for Mobile Games*, 2006
[Laitinen08]	Laitinen, Sauli: *Usability and Playability Expert Evaluation*. In: Isbister, Katherine; Schaffner, Noah: *Game Usability*, Morgan Kaufmann, 2008, S. 91-111
[Nielsen93]	Jakob Nielsen:*Usability Engineering*, Morgan Kaufmann, San Francisco, 1993
[Pinelle081]	David Pinelle, Nelson Wong: *Heuristic Evaluation for Games: Usability Principles for Video Game Design*, 2008
[Pinelle082]	David Pinelle, Nelson Wong: *Using Genres to Customize Usability Evaluations of Video Games*, 2008
[Rollings06]	Rollings, Andrew; Morris, Dave: *Game architecture and design*, New Riders, 2006
[Shaffer07]	Schaffer, Noah: *Heuristics for Usability in Games*, 2007 http://www.playerfriendly.com/files/heuristics.pdf, Abruf: 5.8.09
[Shaffer08]	Shaffer, Noah: *Heuristic Evaluation of Games.* In: Isbister, Katherine; Schaffner, Noah: *Game Usability*, Morgan Kaufmann, 2008, S. 79-89
[ISO9241]	ISO DIN EN 9241: *Ergonomie der Mensch-System Interaktion – Teil 11: Anforderungen an die Gebrauchstauglichkeit – Leitsätze*. 01 1999. – Abruf: 4.7.09
Webseiten der getesteten Spiele	
Anno 1701	http://anno.de.ubi.com/history1701.php
Mirror's Edge	http://mirrorsedge.com/
Drakensang	http://drakensang.de/
Boom Blox	http://www.ea.com/wii-games/boom-blox
Sonstiges	
Nintendo Wii	http://www.nintendo.de/NOE/de_DE/systems/ueber_wii_1069.html
Wii Remote	http://de.wikipedia.org/wiki/Wii-Fernbedienung

Glossar

Heuristik	Eine Heuristik (*griech*. *heuriskein*, „(auf-)finden“, „entdecken“) stellt eine Faustregel zum Auffinden von Usability-Problemen dar
Game Design	Game Design ist die Konzeption der Spielwelt, der Regeln und der Charaktere eines Computerspiels.

Abbildungsverzeichnis