User-Generated Content unter dem Aspekt der User Interfaces von In-Game Editoren

Alexander M. Pippan

DIPLOMARBEIT

eingereicht am
Fachhochschul-Masterstudiengang

Digital Arts

in Hagenberg

im Jänner 2012

Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die aus anderen Quellen entnommenen Stellen als solche gekennzeichnet habe.

Hagenberg, am 25. Januar 2012

Alexander M. Pippan

Inhaltsverzeichnis

Kurzfassung

Parallel mit der Entwicklung des Web 2.0 wandelte sich der *User-generated Content* (UGC) im Kontext der Games. Das Erstellen von Inhalten wird dem Spieler zunehmend durch In-Game Editoren als Teil des Gameplays und zusätzliches Feature ermöglicht. Für Spieler wird durch UGC die Spieldauer verlängert. Entwickler profitieren u. a. durch eine verlängerte Produktlebenszeit und neuem kreativen Input durch die Spieler.

Als Teil der Spielerfahrung stellt sich für die Entwickler die Frage wie die In-Game Editoren für Spieler unterhaltsam gestaltet werden können und diese gleichzeitig in ihrer Tätigkeit unterstützen können.

Diese Diplomarbeit definiert durch die Analyse von vier In-Game Editoren der Games *Little Big Planet 2*, *ModNation Racers*, *inFAMOUS 2* und *Create* eine Einteilung in vier Repräsentationen *Spieler*, *Tools*, *UGC*, *Progress/Status* im User Interface von In-Game Editoren. Die Grundlage der Enteilung bilden identifizierte *Mechanics* und *Aesthetics*. Die richtige Umsetzung der Repräsentationen kann dazu beitragen den Spieler bei der Erstellung von Inhalten zu unterstützen und gleichzeitig zu unterhalten.

Im Diplomprojekt *Little Long Finger* wird die getroffene Einteilung bei der Konzeption zur Anwendung gebracht. Die *Repräsentation des Spielers* und die mit dieser verbundene *Repräsentation der Tools* dienen als stärkstes Mittel zur Steigerung des Unterhaltungswertes, sind aber zugleich abhängig vom Genre des Games. Im Gegensatz dazu, ist *Feedback auf Aktionen* vom Genre unabhängig und kann durch Einsatz von zusätzlichen Effekten wie Partikel, Animationen oder akustischen Signalen unterhaltsam untermalt werden. Für User Interface Elemente die zur Darstellung und Vermeidung von Eingabefehlern dienen, soll die unterhaltsame Darstellung minimiert werden und vor allem selbsterklärend und schnell erkennbar sein.

Abstract

Parallel to the transformation of the internet into what is called Web 2.0, *User-generated Content* in video games made a major step in its evolution. Content creation for players is increasingly made possible by in-game editors as part of the gameplay or as an additional feature. User-generated content extends the games duration of play. Developers benefits include an increased shelf time of the product and new creative input by the players.

By being part of the game experience, developers have to concentrate on how to design in-game editors as entertaining as possible while supporting the players in their activities.

This thesis describes a division into four representations *Player*, *Tools*, *UGC*, *Progress/Status* within user interfaces of in-game editors, whose successful implementation can support and entertain the player during the creation of contents. The division is based on the *Mechanics* and *Aesthetics* identified during the examination of the in-game editors of the four games *Little Big Planet 2*, *ModNation Racers*, *inFAMOUS 2* and *Create*.

The thesis project *Little Long Finger* complements the results by applying the described division to the concept of an in-game editor. The player s representation and the connected representation of the tools are the most striking means to increase the entertainment value of an in-game editor. Their implementation is dependent on the genre of the game. In contrast, the feedbacks of the player s actions are independent and in order to entertain the player they can be designed by using additional effects such as particles, animations or acoustic signals. User interface elements, such as elements for input error avoidance should feature a self-explanatory and easily recognizable design and they should minimize the usage of entertaining effects.

Kapitel 1

Einleitung

1.1 Einführung

Durch technologische Entwicklungen konnten von Nutzern[1] generierte Inhalte, so genannter *User-generated Content* (UGC), eine bedeutende Rolle im Web 2.0 einnehmen. Durch den Wunsch der Spieler Veränderungen und Anpassungen an Computer Games durchzuführen, entstanden noch vor dem Web 2.0 nutzergenerierte Inhalte für Games, so genannte *Modi cations* (Mods). Die für die Entwicklung von *Mods* notwendigen Tools und Editoren entstammten zunächst den Moddern und Communities selbst. Durch die erlangte Aufmerksamkeit der Games Industrie wurden diese später u. a. in Form von *Source Development Kits* (SDKs) zur Verfügung gestellt. Durch neue Kommunikationsabläufe und Verteilungsmöglichkeiten z. B. durch die Hilfe des Web 2.0 und durch Vertriebsplattformen wie z. B. *Steam*, entwickelte sich das Modding zu einem wesentlichen Bestandteil der Games Industrie. Die Vorteile für die Industrie ergeben sich primär aus einer verlängerten Produktlebenszeit und neuem kreativen Input für zukünftige Entwicklungen.

Mit der Entwicklung von den Computer Games weg von den Spielhallen hin ins heimische Ambiente entstand parallel mit dem Aufkommen der *Mods* das Genre der *Open-World Games*, einer neuen Form des freien Spielens. Diese Entwicklung führte mitunter auch zu den *Building Games* durch *Will Wright*. Die Kreativität des Spielers wurde durch einfache Spielmechaniken genutzt und Aufgaben der Spieleentwicklung, z. B. das Gestalten der Spielwelt in der *Simcity-Reihe*, dem Spieler überlassen. Ein neuer Grad an Interaktionsmöglichkeiten zwischen Spieler und Spielwelt begründete den Begriff der *Sandbox Games*. Mit *Spore* wurden verstärkt Modeling Aufgaben aus dem Bereich der Spieleentwicklung als Spielelement eingesetzt. Das Design der Kreaturen, Fahrzeuge und Gebäude wurde als integraler Bestandteil

[1] Zur besseren Lesbarkeit wurden sämtliche personenbezogenen Bezeichnungen nur in einer Form gestaltet. Diese sollen als geschlechterneutral verstanden werden und spiegeln keine Präferenz einer Form wider.

des Games und nicht als optionale Dreingabe integriert. Die einfache und motivierende Abbildung des Modelings in einen für Spieler verständlichen In-Game Editor war ein für den Erfolg und die Akzeptanz durch den Spieler kritischer Punkt.

Der aktuellste Meilenstein von *UGC* im Kontext der Video Games ist die Einführung des *Play, Create und Share Genres* tituliert durch das von *Sony* betriebene Marketing im Zusammenhang mit dem ersten Vertreter *Little Big Planet*. *Mods* sind auf Konsolen traditionell durch das geschlossene System fast nicht vorhanden. Durch die Vernetzung und Öffnung der Konsole für das Internet und die Integration von In-Game Editoren konnten die Spieler in *Little Big Planet* einfach und schnell durch andere Spieler erstellte Inhalte spielen.

1.2 Motivation

Die Entwickler haben das Potential und die Vorteile von UGC in und au erhalb von Video Games erkannt. Externe Modding Editoren erhalten zunehmend Eigenschaften von Sandbox Games, In-Game Editoren wie u. a. die Editoren von Games des Play, Create und Share Genres bieten ein sandboxartiges Gameplay. Das Erstellen von UGC in Games beschränkt sich jedoch meist nur auf die Verwendung als ein optionales Feature. Entwickler u. a. von *Mobile Games* springen auf den Trend UGC auf und bestücken die Spiele nachträglich mit Editoren. Selten wie im Falle von *Spore* oder Vetreter der Play, Create und Share Genres ist UGC und die damit verbundenen In-Game Editoren ein integrales Kernfeature des Spieles, da Risiken der Akzeptanz durch die Spieler und damit u. a. auch finanzielle Risiken für die Entwickler gegeben sind. Wie die Geschichte des Moddings zeigt, ist ein Editor der die gestellten Anforderungen in gebrauchstauglichen Art und Weise erfüllt, ein wesentliches Kriterium bei der Erstellung von UGC.

Editoren in Games sollten zusätzlich unterhaltsam sein und den Spa mit dem Spiel fördern. Es liegt im Interesse der Entwickler, dass In-Game Editoren den Spieler unterstützen die Quantität als auch Qualität der erstellten Inhalte zu erhöhen. Im Sinne des Spielers sollen In-Game Editoren dessen kreativen Schöpfungsdrang durch entsprechende Möglichkeiten im Rahmen der vorgegebenen Spielstruktur bedienen.

Diese Arbeit beschäftigt sich daher mit der Frage nach unterstützenden und unterhaltsamen Ma nahmen die durch Entwickler im User Interface von In-Game Editoren gesetzt werden können.

Die Extraktion von Ma nahmen erfolgt durch eine qualitative Analyse von aktuellen In-Game Editoren. Die notwendige Basis zur Ausarbeitung der Analysekriterien wird im Laufe dieses Dokumentes erarbeitet. Die Arbeit wird zum Verständnis der Strukturierung im folgenden Abschnitt grob umrandet.

1.3 Aufbau dieser Arbeit

Die Behandlung der Fragestellung geschieht in der vorliegenden Arbeit durch die Unterteilung in sieben Bereiche die insgesamt acht Kapitel umfassen:

Kapitel 2 3: In Kapitel 2 (Terminologie und Bedeutungen) werden Definition und Bedeutung der in dieser Arbeit gebräuchlichen Begriffe *User-generated Content* und *Sandbox Games* erarbeitet. Kapitel 3 (UGC im Kontext der Games: Entwicklung und Verbreitung) dient zur Darlegung des aktuellen Standes von UGC im Kontext der Games durch die geschichtliche Entwicklung und die Zusammenhänge zwischen *Modi cations*, *Editoren* und *Sandbox Games*. Aus Sicht der Entwickler werden die Vorteile von UGC erläutert.

Kapitel 4: In Kapitel 4 (In-Game Editoren und Usability in Games) wird die theoretische Grundlage zur Analyse der Fallbeispiele geschaffen. Zunächst wird der Forschungsgegenstand unter besonderer Betrachtung von *Sandbox Games* konkretisiert. Anschlie end werden *Usability* und *Usability in Games* beschrieben um zur Ausarbeitung der Fragestellung überzuleiten.

Kapitel 5: In Kapitel 5 (Forschungsfrage) wird eine detaillierte Problembeschreibung vorgenommen und auf Basis dieser die Formulierung der Forschungsfrage durchgeführt. Die für die Analyse notwendigen Orientierungspunkte bzw. Kriterien werden unter Zuhilfenahme von Methoden aus dem Bereich der *Usability Evaluation* im Anschluss an die Fragestellung erarbeitet und als Fragenkatalog zusammengefasst.

Kapitel 6 7: Kapitel 6 (Analyse von In-Game Editoren und Sharing) und Kapitel 7 (Ergebnisse der Analyse) stellen das Ergebnis der qualitativen Analyse auf Basis der zuvor festgelegten Fragestellung und Fragenkatalog dar. Die Ergebnisse werden zunächst in Kapitel 6 dokumentiert und in Kapitel 7 interpretiert und zusammengefasst.

Kapitel 8: Das Kapitel 8 (Diplomprojekt) befasst sich mit der Ausarbeitung und Entwicklung des Diplomprojektes auf Basis der ermittelten Ergebnisse. Die Ergebnisse der Analyse werden mit den Ergebnissen des Projektes verglichen, erweitert und zur Beantwortung der Fragestellung hinzugezogen.

Kapitel 9: Kapitel 9 (Zusammenfassung und Ausblick) bildet den Abschluss dieser Arbeit durch die Zusammenfassung der Ergebnisse und einem Ausblick auf die Zukunft von UGC und Editoren in Video Games.

Kapitel 2

Terminologie und Bedeutungen

In der vorliegenden Arbeit werden in der Erörterung des State of the Art, Fragestellung und Analyse gebräuchliche Begriffe aus der Welt der Video Games verwendet. Für diese besteht meist ein allgemeines jedoch subjektives Verständnis. Eine für diese Arbeit relevante Definition der Begriffe *User-generated Content* und *Sandbox Games* wird daher in den folgenden Abschnitten durchgeführt.

2.1 User-Generated Content

Der Begriff *User-generated Content* (UGC) oder auch *User-created Content* (UCC) bezeichnet neu erstellte Inhalte (Content[1]) durch den Benutzer (User) eines Systems und wurde vor allem geprägt im Zusammenhang mit dem Terminus des *Web 2.0* und der Weiterentwicklung des Internets Anfang des 21. Jahrhunderts [15]. Der Begriff *User* definiert eine Trennung zwischen der Content erstellenden Person, dem *Maker*, und der Person die Content besitzt und konsumiert, dem *User*. *UGC* impliziert zumindest zwei Parteien *Maker* und *User* und im Zusammenhang dieser zwei durch diese Parteien nutzbare Dinge *Tools* und *Content*. *User* werden zu *Maker* und umgekehrt [11].

Wenngleich ein subjektives Verständnis des Begriffes *UGC* existiert, offenbart sich bei genauerer Betrachtung die Schwierigkeit den Begriff auf Grund der Vielschichtigkeit und der vielfältigen Anwendungen zu definieren. In [15], einer Studie von *UGC* im Kontext des Internets im Speziellen des *Web 2.0*, wurden drei Charakteristika (*Creative E ort*, *Publication Requirement*, *Creation Outside of Professional Routines and Practises*) spezifiziert, die als Grundlage dienen *UGC* als solchen zu identifizieren und von *Nicht-UGC* zu unterscheiden. Wird *UGC* im Kontext der Video Games betrachtet lassen sich diese Charakteristika ebenfalls erkennen. Im folgenden Abschnitt

[1] Der Begriff *Content* wird allgemein in Broadcast- und Unterhaltunsindustrie zur Beschreibung von Informationen verwendet, die einem Publikum präsentiert werden können [11].

werden diese nach der Definition von [15] erörtert um auf Basis derer die Abgrenzung und Einteilung von *UGC* im Kontext der Games zu treffen.

2.1.1 Charakteristika von UGC

Creative E ort

Der *Creative E ort* von UGC verweist auf einen kreativen Aufwand der bei der Erstellung von neuem Content oder Adaption von bestehendem Content vom User investiert werden muss. Der User muss eigenen kreativen Input liefern und eine Wertsteigerung herbeiführen. D. h. die Aufnahme einer z. B. TV-Sendung Online verfügbar zu machen, erfüllt die Charakteristik des *Creative E orts* nicht, da hier dem bestehendem Content keine Wertsteigerung zugefügt wurde. Ein eigenes neues Video, eigene Fotos oder der Ausdruck eigener Gedanken über Blogs erfüllen die Charakteristik, weil ein kreativer Aufwand erbracht wurde. Würde die Aufnahme der TV-Sendung in anderer Form durch den User arrangiert werden, wäre die Charakteristik wiederum erfüllt. Der minimale kreative Aufwand der betrieben werden muss lässt sich schwer definieren und hängt vom jeweiligen Kontext ab [15].

Im Kontext der Games muss der Spieler durch das Spielsystem den Freiraum besitzen einen kreativen Aufwand zu erbringen. Dies kann z. B. wie in *Simcity 4*[2] (2003, Maxis) oder *Minecraft*[3] (2009, Moyang) bereits durch die Möglichkeit erfüllt sein, Strukturen nach eigenen Vorstellungen, im Rahmen des im Spielsystem Möglichen, zu platzieren. In Video Games existieren unter dem Begriff *Crafting* (siehe Abschn. 2.1.2) Mechanismen die es dem User ermöglichen neuen Content, meist in Form von Items, zu erzeugen. Der *Creative E ort* ist hierfür jedoch nicht erfüllt, da es sich um keine echte *Creation* handelt [11].

Publication Requirement

In [15] wird ein *Publication Requirement* d. h. die Verbreitung des erstellten Contents in einem Kontext z. B. Veröffentlichung in einem Netzwerk, einem Blog oder auf einer Webseite als wichtiges Merkmal von UGC mit dem Bezug zu Web 2.0 definiert. Dadurch lassen sich Services des Internets wie *E-Mail*, *Instant-Messages*, u. Ä. effektiv ausschlie en [15].

Mit Bezug auf die Video Games ist diese Charakteristik nicht in allen Fällen erfüllt. Im Zuge dieser Arbeit und der Definition von UGC in Games wird diese Charakteristik als optional angesehen. UGC kann hier auch innerhalb des geschlossenen Systems des Games Anwendung finden. UGC wird in Games wie z. B. *Simcity 4* und *Minecraft* als Teil des Gameplays durch einfache *Game Mechanics* zur Platzierung von Spielobjekten eingesetzt. Die

[2] http://simcity.ea.com
[3] http://www.minecraft.com

Spieler schaffen sich während dem Spielen eine selbst kreierte Welt die im weiteren Spielverlauf ausschlaggebend für zukünftige Entscheidungen ist. Sie publizieren und konsumieren fortlaufend die selbst erstellten Inhalte.

Creation Outside of Professional Routines and Practises

UGC sollte frei von kommerzieller oder institutioneller Verbindungen entstehen. Die Entstehung soll durch nicht-professionelle User von statten gehen und ein wirtschaftlicher Gewinn nicht als die treibende Kraft im Hintergrund stehen [15].

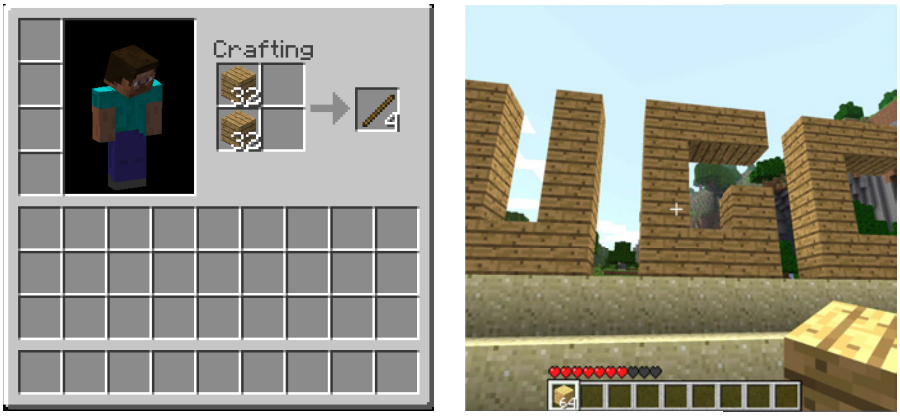
In [15] wird weiters von einem Trend zur Monetarisierung von UGC gesprochen der in gewisser Weise Parallelen zum UGC im Kontext der Games aufweist. So existieren für Games Business-Modelle zur Monetarisierung von UGC durch den User wie z. B. in *Team Fortress 2*[4] (2007, Valve Corporation) seit dem so genannten *Mann-Conomy Update* [69], bei dem durch einen Contest [67] ausgewählter UGC in Form von Gegenständen wie Waffen oder Bekleidung in das Spiel integriert wurden [69].

Ein weiterer Graubereich dieser Charakteristik ergibt sich durch den Begriff User. Dieser kann im Fall von UGC auch einen professionellen User beschreiben, also eine professionell tätige Person die in ihrer Freizeit Content erstellt. Auch besteht wie im Fall von *Team Fortress 2* die Möglichkeit, dass User auf Grund ihrer erstellten Inhalte zum professionellen User avancieren [15]. Beispiele dieser erfolgten Wandlung finden sich in der Modding-Szene (siehe Abschnitt 3.1) wie z. B. *Counter-Strike*[5] (1999, Minh Le).

2.1.2 Crafting und Creation

Um ein besseres Verständnis der Charakteristik des Creative Efforts und der Identifikation von UGC in Video Games zu erlangen, wird folgend als Beispiel der Unterschied von *Crafting* und *Creation* herangezogen. *Crafting* findet meist Anwendung im Genre der *Massive Multiplayer Online Role-Playing Games* (MMORPGs) aber auch z. B. im bereits genannten *Team Fortress 2*, einem Vertreter des *First-Person Shooter* (FPS) Genres. In [11] wird es im Kontext der MMORPGs als Prozess der Weiterbildung des Charakters, dem so genannten *Leveling* bezeichnet. Dies geschieht durch die repetitive Generierung von Spielobjekten. Die Spielobjekte die mittels *Crafting* durch den Spieler gewählt und erzeugt werden, sind durch die Entwickler des Games vorgegeben. Crafting-Systeme basieren meist auf dem Sammeln von Rohmaterialien und deren Kombination zu neuen, aber vordefinierten Gegenständen. Es wird kein neuer Content im Sinne des Creative Efforts durch den User erstellt, da es keinen kreativen Aufwand und keine Wertsteigerung gibt. Dieser Sachverhalt existiert nicht au erhalb der virtuellen Welt, in der

[4] http://www.tf2.com
[5] http://www.counter-strike.net



(a) (b)

Abbildung 2.1: Verdeutlichung der Charakteristik des *Creative Efforts* am Beispiel *Crafting* (a) und *Creation* (b) in *Minecraft Beta 1.8.1.* In (a) werden durch den Entwickler festgelegte Regeln neue Gegenstände erzeugt z. B. Stäbe aus dem Rohmaterial Holz. In (b) wurde mit Holz durch den User ein neues Gebilde erschaffen. Die Charakteristik ist im letzteren Fall erfüllt.

höherwertige Gegenstände durch Kombination von niederwertigen Materialen und einem kreativen Aufwand erzeugt werden. Als Beispiel wird in [11] eine aus Metall und Glas hergestellte Uhr angeführt, deren Wert sich u. a. durch den investierten Aufwand und Zeit definieren lässt.

Um in der virtuellen Welt, also z. B. in Video Games von *echter Creation* sprechen zu können, bedarf es der Möglichkeit des Users die Charakteristik des Creative Efforts auch umzusetzen [11]. Als illustratives Beispiel (siehe Abb. 2.1) sei hier Minecraft angeführt, dass zum Abbau von Rohmaterialien, zum *Crafting* und zur Platzierung *Game Mechanics* bereitstellt. Ein durch Platzieren von Rohmaterialien oder Gegenständen neu erstelltes Gebilde (Content), kann als UGC bezeichnet werden da die Charakteristik des Creative Efforts durch den Akt des freien Platzierens erfüllt werden kann.

Parameter-based Creation

Parameter-based Creation wird in [59] als einfache Form der Erstellung von UGC in Games bezeichnet. Es handelt sich hierbei z. B. um die Anpassung des eigenen Charakters durch vorgegebene Parameter [59]. Ähnlich dem Abschnitt *Crafting und Creation* (s. oben) stellt sich hier die Frage nach der Charakteristik des Creative Efforts. Die Parameter sind vorgegeben und damit verbunden auch die möglichen Kombinationsformen und Ergebnisse. Theoretisch könnte auch bei *Minecraft* jeder Platzierungspunkt beliebig kombiniert werden. Einstellungen könnten jedoch z. B. einer Person nachempfunden werden, besitzen daher auch eine Steigerung des subjektiv emp-

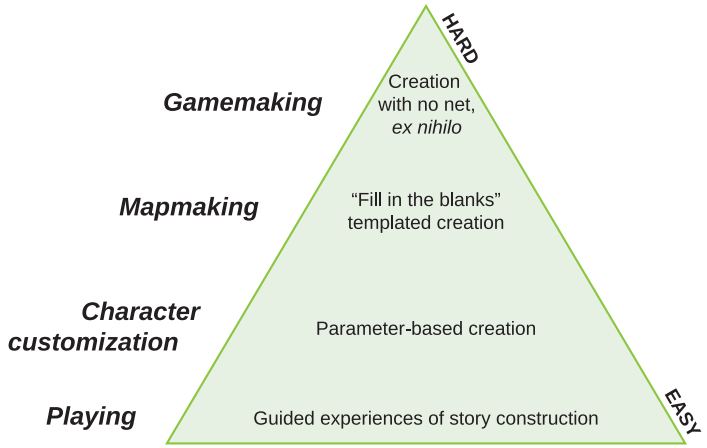


Abbildung 2.2: Stufen der *Content Creation* bzw. *User Engagement* nach Koster [59]. Jeder Spieler erstellt Inhalte auf einer bestimmten Stufe, wobei die Anzahl der Spieler mit steigendem Schwierigkeitsgrad abnimmt. Nach Bildquelle: [59].

fundenen Wertes und ein Einbringen eines kreativen Aufwandes. Es kann von echter *Creation* gesprochen werden.

2.1.3 Stufen der Content Creation

Koster hat in [59] basierend auf der *Content Creation Pyramide* von *Will Wright*, eine aus Sicht der Spieler relevante Umstrukturierung vorgenommen [59, 21]. Die Pyramide nach Koster (siehe Abb. 2.2) setzt die Formen von UGC in Bezug zum Schwierigkeitsgrad. Dabei sinkt mit zunehmender Schwierigkeit die Anzahl erstellender Spieler. Koster stellte sich nicht die Frage ob Spieler Inhalte erstellen sondern ob die präferierte Ebene den jeweiligen Spielern zur Verfügung steht [59]. Die Pyramide wird in Abschn. 5.1 zur Formulierung der Problemstellung und Zielsetzung herangezogen und dient im folgenden Abschnitt als Ergänzung zur Einteilung von UGC im Kontext der Games.

2.1.4 UGC im Kontext der Video Games

UGC im Kontext von Video Games existiert in einem sehr breiten Spektrum. In den vorangeganenen Abschnitten wurde auf Grund der starken Prägung des Begriffes UGC im Kontext des Web 2.0 dessen Charakteristika erörtert. Diese sollen dazu dienen UGC prinzipiell zu erkennen. Die wichtigste Charakteristik stellt dabei der *Creative E ort* dar, da wie bereits angesprochen ein *Publication Requirement* und *Non-Professional Routines* für Games als nicht erforderlich zu betrachten sind. Für UGC im Kontext der Games wurde für

den weiteren Verlauf dieser Arbeit eine Einteilung[6] nach den Charakteristika des UGC, sowie Zweck und Ort der Verwendung der Editoren getroffen. Diese teilt sich in *Integrierte In-Game Editoren*, *Optionale In-Game Editoren*, *Externe Editoren* und *Ludic Artifacts*:

Integrierte In-Game Editoren: Der erstellte UGC ist Teil des Gameplays. Die Editoren oder einzelne Mechanics zum Editieren müssen vom Spieler benutzt werden um im Game voranzuschreiten. Beispiele sind *Minecraft*, *The Sims*[7] (2000, Maxis) oder *Spore*[8] (2008, Maxis).

Optionale In-Game Editoren: Das Erstellen von UGC und das Bedienen des Editors ist optional. Diese Arbeit beschäftigt sich primär mit dieser Form von In-Game Editoren und UGC. Eine detailliertere Beschreibung wird daher in Abschn. 2.1.5 vorgenommen.

Externe Editoren: UGC wird durch die Verwendung von externen Editoren au erhalb des Games erstellt. Die Erstellung von Inhalten, so genannten *Mods* (siehe Abschn. 3.1), setzt vor allem ein umfangreiches Know-How durch den User vorraus [2].

Ludic Artifacts: Ludic Artifacts sind UGC im Kontext der Games definieren aber keinen zusätzlichen Content *für* Games. UGC wird au erhalb der Spielgrenzen in Form von z. B. Gra ties, Kostümen oder Videos erzeugt [66].

2.1.5 Fokus: Play, Create und Share Genre

Der Fokus dieser Arbeit liegt auf UGC durch *Optionale Editoren*. Der produzierte UGC lässt sich am stärksten mit der Definition von UGC im Kontext des Web 2.0 vergleichen. Im weiteren Verlauf dieser Arbeit wird diese Kategorie als *Editoren des Play, Create und Share Genres* bezeichnet.

Der Ausdruck des *Play, Create and Share* wurde durch *Sony*[9] und *Media Molecule*[10] im Zusammenhang mit deren Game *Little Big Planet*[11] (2008, Media Molecule) geprägt und geformt. Das Verwenden des Editors ist für den *Play-Part* nicht erforderlich kann aber als Core-Feature des Games betrachtet werden. Dem User bleibt die Wahl ob er sich kreativ betätigen möchte. Es wird versucht die Grenze zwischen *Play* und *Create* zu verwischen [50, 61, 65].

Ein weiteres wichtiges Feature ist das *Sharing*. *Sharing* ermöglicht es den selbst erstellten Content direkt über das Game online zu verteilen oder neuen Content runterzuladen und zu spielen. Games die diesem Genre angehören

[6] Diese Einteilung versteht sich als Vorschlag und dient zur besseren Differenzierung von In-Game Editoren innerhalb dieser Arbeit. Sie erhebt keinen Anspruch auf Vollständigkeit.

[7] http://thesims.ea.com

[8] http://www.spore.com

[9] http://www.sony.com

[10] http://www.mediamolecule.com

[11] http://www.littlebigplanet.com

sind u. a. *ModNation Racers*[12] (2010, United Front Games) und *Create*[13] (2010, EA Bright Light).

2.2 Sandbox Games

Sandbox Games bezeichnen ein Genre der Video Games das sich besonders durch die dem Spieler vermittelte Freiheit innerhalb der Spielwelt auszeichnet. Sie werden auch als *Open-World Games* bzw. *Exploration Games* bezeichnet, wenngleich sich diese Bezeichnung im aktuellen Verständnis von *Sandbox Games* auf ein spezifisches Merkmal der Spielwelt bzw. des Gameplay-Designs bezieht. *Sandbox Games* realisieren spezifische Merkmale und Design-Strukturen des abstraken Konzeptes einer *Sandbox* [44, 54]. Zum Verständnis und Identifikation dieses Genres werden diese im Abschnitt 2.2.1 zusammengefasst.

So genannte *Building Games* wie die *Simcity*-Reihe[14], also Games mit UGC nach der Einteilung (siehe Abschnitt 2.1.4) *Integrierter Editoren*, als auch UGC im *Play, Create and Share Genre* sind unter anderem Subgenres des *Sandbox Game* Genres. Bei ihnen ist die Grenze zwischen *Create* und *Play* ie end [44], woraus sich die Relevanz von *Sandbox Games* für diese Arbeit ergibt. In Abschn. 3.2 wird daher auch der geschichtliche Verlauf von *Sandbox Games* in Verbindung mit UGC beschrieben und in Abschn. 4.1 der Zusammenhang zwischen *Sandbox Games* und dem Forschungsgegenstand hergestellt.

2.2.1 Merkmale und Design-Strukturen

Sandbox Games besitzen eine *Non-Lineare* und *Open-Ended* Spielstruktur in einer frei erkundbaren Spielwelt. D. h. es obliegt dem Spieler wie sich den Zielen genähert wird, wie diese beendet werden und welche Hilfsmittel benutzt werden. Dies führt zu einem verstärktem Gefühl der Zufriedenheit und dem Teilsein mit der Spielwelt. Spieler sind frei in der Möglichkeit sich eigene Spielziele zu kreieren und die Bewertung ihres Erfolges durchzuführen. Die Reihenfolge von Inhalten die dem Spieler präsentiert werden, sind nicht durch den Entwickler des Games festgelegt. Es gibt keinen richtigen Weg das Game zu spielen [38, 44, 47, 71].

Zur Identifikation von Sandbox Games als solche seien die folgenden Merkmale bzw. Design-Strukturen gegeben:

Open-World-Design: Ein Merkmal von Sandbox Games ist die Auslegung der Spielwelt weg von einer klassischen linear präsentierten Levelstruktur hin zu einer durch den Spieler frei begehbaren und erkund-

[12] http://www.modnation.com
[13] http://create.ea.com
[14] http://simcity.ea.com

baren Spielwelt. *Movement* und *Exploration* sind die grundlegenden Gameplay-Konzepte in *Open-World Games*. Der Spieler ist auf sich alleine gestellt eine gro e Welt zu erkunden. Wann neue Regionen besucht werden obliegt dem Spieler [44, 54].

Multi-Threaded oder Non-Linear Story: Wenn vorhanden, gestaltet sich die Präsentation einer Story in Sandbox Games durch die Freiheit des Spielers im Vergleich zu anderen Games meist in Form einer *Multi-Threaded* oder *Non-Linear* Story, d. h. durch eine Erzählung die je nach Aktion und Reihenfolge des Spielers verschiedene Handlungsstränge aufweisen kann oder die Reihenfolge der Erzählung für den Plot nicht von Relevanz ist. Die Reihenfolge von erzählerischen Elementen wird durch den Spieler vorgegeben, nicht durch den Entwickler. Es wird in dieser Form von *Responsive Narratives* gesprochen [38, 44].

Automated Responsiveness to Player Behaviour: Der Begriff Sandbox Game wurde zu Beginn des Jahrtausends ma geblich geprägt durch diese neue Entwicklung im Game Design. *Sandbox Design* verspricht dem Spieler eine hohe Interaktionsmöglichkeit mit dem Spiel durch u. a. *Non-Player Characters* (NPCs), als auch eine aktive Unterstützung beim Experimentieren mit der Spielwelt [44, 48].

In *GTA III*[15] (2001, Rockstar North) wurde erstmals eine Spielwelt geschaffen die in umfangreicher Form auf das Verhalten des Spielers reagierte und so eine neue Ebene an Interaktion erreichte. In *The Sims* simulieren glaubhafte Charaktere den notwendigen Raum zur Interaktivität [44, 48].

User-Generated Content: Game Design selbst wird als das absolute Sandbox Game gesehen. Spielobjektive werden festgelegt, Inhalte erstellt, eine Spielwelt geschaffen in die der Spieler eintaucht. In Games findet dieses Konzept u. a. Anwendung im Genre der *Building Games*. Dem Spieler wird z. B. in *Utopia* (1981, Intellivision) und *Simcity* (1989, Maxis) freies Bauen innerhalb der durch Regeln festgelegten Grenzen ermöglicht [44].

Aktuelle Spiele wie z. B. *Spore* gehen einen Schritt weiter und präsentieren spezielle Bereiche des Game Designs als Core Mechanic bzw. Core Feature. Die Modellierung von Charakteren, Fahrzeugen und Gebäuden wird für Einsteiger zugänglich In-Game aufbereitet [44].

2.2.2 Metaplay

Unabhängig davon ob vom Entwickler konzipiert oder nicht, können Games ein *sandboxartiges Gameplay* oder *Metaplay* bereitstellen. *Metaplay* beschreibt ein Spielen des Games in einer Form die nicht durch die Entwickler vorrausgesehen war. Der Spieler beschäftigt sich z. B. mit der Auslotung von

[15] http://www.rockstargames.com/grandtheftauto3

Fehlern, den Limits des Games und der Verfolgung von eigenen aufgestellten Zielen [44]. Zum Beispiel ob die Missionen in *Command and Conquer: Generals*[16] (2003, Electronic Arts) nur mit Hilfe eines Einheitentyps erfolgreich beendet werden können.

Der Spieler denkt mehr an das dahinterliegende System und wie dieses optimiert oder dessen Regeln gebrochen werden können. Das Sandbox Game Genre ist demnach im Sinne des *Play* ein wesentlich breiteres Genre als im Sinne von durch den Entwickler verwendeten Design-Strukturen. Viele Games können als solche gespielt werden, es ist nur abhängig vom Einfallsreichtum und Kreativität des Spielers [44].

2.3 Zusammenfassung

User-Generated Content (UGC) im Web 2.0 definiert sich durch die drei Charakteristika *Creative E ort*, *Publication Requirement* und *Creation Outside of Professional Routines and Practises*. Zur Identifikation von UGC in Games dient vor allem der *Creative E ort* da ein Nutzen für Games auch ohne *Publication Requirement* gegeben ist. Für diese Arbeit wurde eine Einteilung von UGC in Games getroffen in *Integrierte In-Game Editoren*, *Optionale In-Game Editoren*, *Externe In-Game Editoren* und *Ludic Artifacts*. Der Fokus dieser Arbeit liegt auf den optionalen Editoren des *Play, Create und Share Genres*, deren Eigenschaften mit denen des UGC im Web 2.0 vergleichbar sind. Die freie Spielgestaltung und hohe Interaktionsmöglichkeit sind Grundeigenschaften von *Sandbox Games*. Die Verbindung zwischen *Sandbox Games* und Editoren wird im weiteren Verlauf dieser Arbeit näher behandelt.

[16] http://www.commandandconquer.com

Kapitel 3

UGC im Kontext der Games: Entwicklung und Verbreitung

Dieses Kapitel stellt den State of the Art von UGC im Kontext von Video Games dar. Die Trennung des Kapitels erfolgt in eine für diese Arbeit relevante Differenzierung in UGC au erhalb und UGC innerhalb von Games. Neben einem allgemeinen geschichtlichen Überblick und dem aktuellen Stand liegt der Fokus insbesondere auf den Vorteilen von UGC für die Entwickler und wie die Entwicklung der *Mods* das *Play, Create und Share Genre* ankündigte.

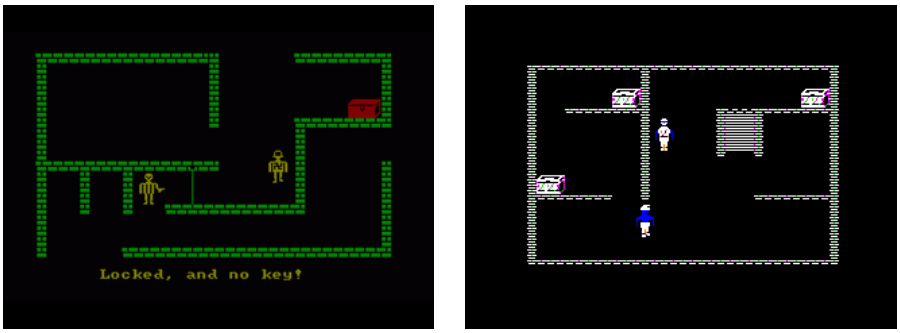
3.1 UGC außerhalb von Games: Mods

Im Allgemeinen bezeichnet eine *Modi cation* kurz *Mod* (deutsch Modifikation) eine Veränderung oder Erweiterung eines ursprünglichen Computer Games durch den User[1] (siehe Abschn. 2.1) [2]. Das Ausführen des Aktes der Erstellung von *Mods* wird als *Modding* bezeichnet, die ausführende Person als *Modder* [22]. Im Kontext der *Mods* von Computer Games können die Grenzen zwischen User und Entwickler, dem Ersteller in erster Instanz, verwischen. Durch das *Modding* wird ein User gleichzeitig zum Entwickler [9]. *Modi cations* von Computer Games gehören zu der ursprünglichsten und etabliertesten Form von UGC im Kontext von Video Games [2]. Nach der in Abschn. 2.1.4 durchgeführten Einteilung wird dieser UGC der Gruppe der *Externen Editoren* zugeteilt.

3.1.1 Überblick und Einteilung

Die *Modder-Kultur* entstand noch vor dem Aufkommen des Internets und entsprang der durch *Steven Levy* in [12] benannten *Hackerethik* [60, 22]. *Adventure* (1976, Donald Woods) war eine erste öffentlich bekannte Modifikation basierend auf einem 1972 entwickelten text-basierten Computer Game

[1] Ein Spieler, Käufer eines Computer Games bzw. ein Konsument [9]



(a) (b)

Abbildung 3.1: Ein Vergleich zwischen originalem Game (a) *Castle Wolfenstein* (1981, Muse Software) und der ersten unter der heutigen Vorstellung gesehenen *Mod* (b) *Castle Smurfenstein* (1983). Bildquellen v.l.n.r.: [26] und [27].

von *Will Crowther*. *Adventure* war durch die Bewegungsfreiheit des Spielers eine Vorankündigkung von Sandbox Games (siehe Abschn. 3.2.3) [44, 60].

Als erster Vertreter der heutigen Vorstellung unter dem Begriff Mod wird *Castle Smurfenstein*[2] (1983) (siehe Abb. 3.1) gesehen, eine Modifikation des originalen Games *Castle Wolfenstein* (1981, Muse Software) (siehe Abb. 3.1). In der Mod wurden die Charaktere und Texte mit Schlümpfen und ähnlichen Inhalten ausgetauscht [9, 70]. Das Aufkommen der Mods war kein Produkt der Industrie, sondern das Ergebnis der Motivation von Spielern das Game zu verbessern oder ihren eigenen Bedürfnissen anzupassen [70]. Die weitere Entwicklung der Mods ist eng verknüpft mit der Entwicklung der PCs, der aufkommenden 3D-Grafik und dem breiten Zugang zum Internet [22].

In dieser Entwicklung kristallisierten sich verschiedene Ausprägungsformen und Stufen von Mods heraus. Aufbauend darauf wurden diese in Abb. 3.3 zugeordnet und dargestellt. Die Ausprägungsformen unterscheiden sich in Grö e und Komplexität [14]. Die Aufteilung gibt eine Übersicht der Teilbereiche eines Games die durch Spieler modifiziert werden können. Bezugnehmend auf die *Stufen der Content Creation* nach Koster (siehe Abschn. 2.1.3), beginnt das Modding ab der Stufe der *Parameter-based Creation*. Neben *Total Conversions*, der umfangreichsten Veränderung am Original (siehe Abb. 3.2), sind Mods in Form von Anpassungen an *Skins*, *Models* und *Maps* am weitesten verbreitet [2].

Die Modder wurden ein integraler Bestandteil der Games Industrie. Durch eine aktive Mod-Gemeinde verwischen die Grenzen zwischen Spieler und Entwickler [70]. Sie beein ussen die Industrie und die Entwicklung der Games auf technologischem Level als auch auf dem Level des zukünftigen Game-Designs [14]. Eine aktive Mod-Gemeinde erhöht den Wert eines Produktes

[2] http://www.evl.uic.edu/aej/smurf.html

(a) (b)

Abbildung 3.2: Vergleich des ursprünglichen Games *Unreal Tournament 2004* (2004, Digital Extremes, Epic Games) (a) und der *Total Conversion Red Orchestra: Ostfront 41–45* (2006, Tripwire Interactive) (b). Bildquellen v.l.n.r.: [28] und [29].

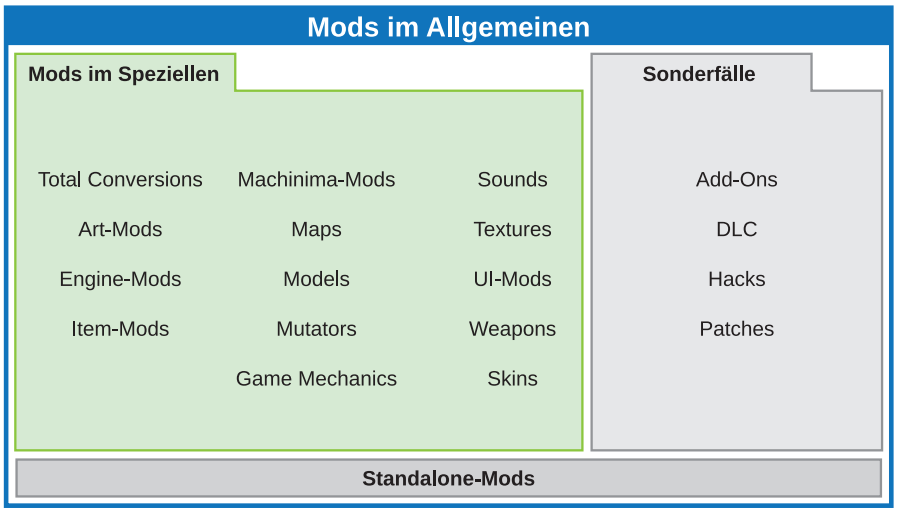
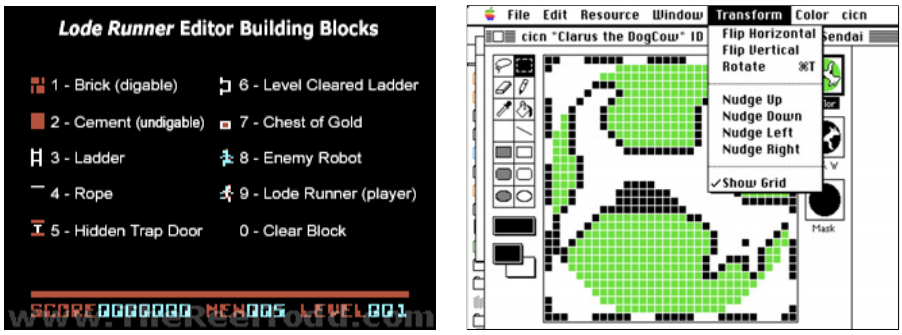


Abbildung 3.3: Übersicht und Einteilung verschiedener Ausprägungen von Mods nach [2, 9, 14, 18, 22, 70] unter Berücksichtigung der historischen begrifflichen Auslagerung.

und verlängert den Lebenszyklus durch Erweiterung der Spieldauer über die des originalen Games hinaus. Sie liefert neue innovative und kreative Ideen und Konzepte, deren Umsetzung im Rahmen einer kommerziellen Entwicklung ein zu hohes Risiko bedeuten würde. U. a. erkannte *Will Wright* frühzeitig den Nutzen und sah das Modding als wichtiges Feature für *The Sims* und folgende Games an [9, 70].



(a) (b)

Abbildung 3.4: Zwei Editoren (a) der Hilfebildschirm des beigelegten Leveleditors von *Lode Runner (C64)* und (b) *ResEdit* von *Apple Computer* mit der Ansicht des *Bit Editors*. Beide wurden schon in der Frühzeit der Mods zum Erstellen und Bearbeiten neuer Inhalte für Games verwendet. Bildquellen v.l.n.r.: [30] und [1, S. 28].

3.1.2 Editoren und SDKs

Ein *Editor* ist ein Programm zur Erstellung und Modifikation von Daten [25]. Bei der Entwicklung von Mods kommen je nach Ausprägungsform verschiedene *Editoren* zum Einsatz. So werden für das Zeichnen und Bearbeiten von Texturen *Editoren* wie z. B. *Adobe Photoshop*[3] benötigt, für das Erstellen von neuen Modellen 3D-Grafikprogramme wie z. B. *Autodesk Maya*[4]. Häufig werden für das Modding entsprechende *Editoren* durch den Entwickler als Teil der *Source Development Kits* zur Verfügung gestellt. So genannte *SDKS* beinhalten die Quellcodes, Editoren, Grafiken und sonstige Mediendateien des Games [18]. Bei Nichtvorhandensein solcher finden die technisch versierten Benutzer z. B. durch *Hacks* Wege um die Modifikation des Games zu ermöglichen [60]. Dies geschah beispielsweise Mitte der 80er Jahre am *Apple II* mit dem frei verfügbaren Tool *ResEdit* (siehe Abb. 3.4) [60]. Der mitgelieferte Editor von *Lode Runner* (1983, Broderbund, C-64) (siehe Abb. 3.4) ermöglichte erstmals die Erstellung von neuen Levels durch ein o zielles Werkzeug [22].

Anfang der 90er Jahre stand durch die Entwicklung von Level-Tools durch die Modding-Gemeinde nun jedem mit Interesse und etwas technischem Hintergrundwissen die Möglichkeit offen einen Level für ein umfangreiches Game zu erstellen. Der entscheidende Faktor Internet trug zur schnellen Verbreitung über Foren bei. Spieler wurden zu Spielemachern [60].

Entwickler wie z. B. *id Software*, *Valve Corporation* oder *Epic Games* begannen aktiv die Erstellung von Mods zu forcieren, indem zum einen versucht

[3] http://www.adobe.com/de/products/photoshop.html
[4] http://usa.autodesk.com/maya

wurde die technische Einstiegsbarriere durch zur Verfügung gestellte *Editoren* und *SDKs* zu verkleinern und zum anderen den Prozess der Distribution von Mods zu erleichtern [70]. Beispielsweise stand für die Entwicklung der Online-Vertriebsplatform *Steam*[5] ein umfangreicher Support zur Entwicklung und Distribution von Mods stets als wichtiges Feature im Vordergrund.

Durch die geschichtliche Entwicklung von Relevanz aber auch durch Technologie, Verbreitung und Anzahl der in Entwicklung befindlichen Mods (siehe Tab. 3.1) sind aktuell das *Unreal Development Kit*[6] [7] (UDK, Epic Games), *Source SDK*[8] (Valve Corporation) und *CryENGINE 3 SDK* (Crytek GmbH.) besonders hervorzuheben. Das *UDK* mit dem *UnrealEd* und insbesondere das *CryENGINE 3 SDK*[9] [10] mit dessen *CryENGINE 3 Sandbox* bilden aktuell den State of the Art (siehe Abb. 3.5) der Mod-Editoren in Umfang, Work ow und User Interface. Ein Beispiel eines Editors der *CryENGINE 3 Sandbox* ist der so genannte *Flow Graph Editor* (siehe Abb. 3.6) der es neben seiner primären Aufgabe des codefreien Level-Scriptings z. B. auch ermöglicht neue Spiele in zumindest prototypischer Form zu erstellen, ohne eine Zeile Quellcode zu verfassen [49, 51, 57, 68].

Ein herausragender und wesentlicher Unterschied zum *Valve Hammer* ist das durch *Crytek* titulierte *What You See Is What You Play* (WYSIWYP) System. Der User kann innerhalb des *UnrealEd* oder der *CryENGINE 3 Sandbox* alle Änderungen in Echtzeit begutachten und vor allem sofort und ohne Verzögerung sich als Spieler in die Spielwelt versetzen und eben noch erstellte Inhalte testen und spielen [49, 51, 68].

Die vorgestellten Editoren können in die Gruppe *Externer Editoren* eingeteilt werden. Im Gegensatz zu In-Game Editoren wird vom User ein hohes Ma an Motivation und technischem Know-How vorrausgesetzt. Die *Editoren* sind spezialisierte oder durch profesionelle Anwenderschaft genutzte *Tools* mit einem umfangreichen Repertoire an Optionen und Möglichkeiten (siehe Abb. 3.7) [2]. Für die *In-Game Editoren* ist ein möglichst simples Interface und einfache Bedienung bei hohem kreativen Freiraum gefordert [52, 65]. Die Verwendung von In-Game Editoren wird als Teil des folgenden Abschnittes behandelt.

3.2 UGC innerhalb von Games: Sandbox Games

Der letzte Abschnitt dieses Kapitels befasste sich primär mit UGC au erhalb von Games in der Form von Mods. Es wurde ein kurzer Überblick über die Entwicklung der Mods und externen Editoren gegeben. Die Bedeutung und

[5] http://www.steampowered.com
[6] http://www.udk.com
[7] http://www.unrealengine.com
[8] http://developer.valvesoftware.com/wiki
[9] http://www.mycryengine.com
[10] http://www.crydev.net

Tabelle 3.1: Au istung bereits veröffentlichter und Anzahl in Entwicklung be ndlicher Mods unter Verwendung von *CryENGINE 3 SDK*, *Source SDK* und *Unreal Development Kit* nach http://www.moddb.com stand 16.08.2011. Zur Inkludierung des historischen Verlaufs wurden auch Mods die über ältere SDK-Versionen erstellt wurden berücksichtigt.

Aktuelles SDK	*Markante Games*	*Beendet*	*Gesamt*
CryENGINE 3 SDK	Far Cry, Crysis, Crysis 2, Crysis Warhead, Crysis Wars	76	272
Source SDK	Half-Life (HL), HL2, HL2:Episode 2, Left 4 Dead (L4D), L4D 2	634	2309
UDK	Unreal Tournament (UT), UT 2003, UT 2004, UT3	347	587

Vorteile von einer aktiven Modding-Community wurden erörtert.

Der Fokus dieser Arbeit liegt auf Editoren die innerhalb des Games Anwendung finden. UGC innerhalb von Games ist weit verbreitet und heute u. a. ein wichtiger Bestandteil von Sandbox Games. In diesem Abschnitt wird daher auf die geschichtliche Entwicklung der Sandbox Games als auch auf UGC innerhalb von Games eingegangen.

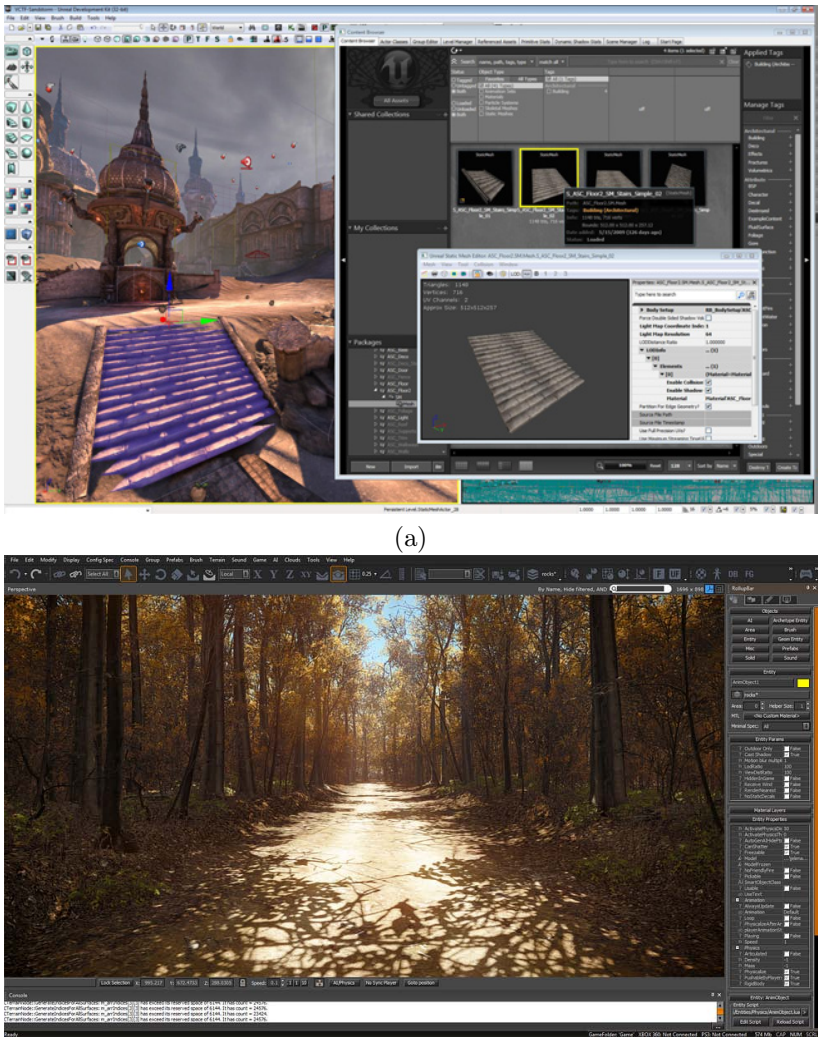
3.2.1 Open-World Games

Vor der Prägung des Begriffes Sandbox Games um die Jahrtausendwende entstand mit Beginn der 80er eine neue Art und Genre von Computer Games die *Open-World Games*. Verantwortlich dafür war eine Veränderung der Spielgewohnheit weg von Spielhallen hin zum Wohnzimmer-Ambiente. Für die Entwickler und Designer wurde es möglich den Spielern mehr Zeit für die Erkundung der Spielwelt zu geben [44]. Eine erste Ankündigung und Vorhut des Genres bildete *Adventure* (1978, Atari 2600) von *Warren Robinett* in welchem der Spieler die Spielwelt frei erkunden konnte [71].

Die Erkundung in *Open-World Games* geschieht durch Bewegung, wie z. B. in *Flight* (1982, subLOGIC) dessen Konzept auf der Bewegung des Flugzeuges allein basiert und keine erzählerischen Elemente beinhaltet. Das so genannte *Movement* wird auch in aktuellen Titeln wie *Assassins Creed*[11] (2007, Ubisoft Montreal) oder *Mirror s Edge*[12] (2008, DICE) umgesetzt [44].

Elite (1983) entwickelt von *Ian Bell* und *David Braben* stellte durch die weitreichende Spielwelt und die Freiheit der Bewegung und Entscheidung des Spielers einen Meilenstein in der Entwicklung der *Open-World Games* dar. Das Universum, die Spielwelt, von *Elite* wurde prozedural beim Starten

[11] http://assassinscreed.ubi.com
[12] http://www.ea.com/mirrors-edge



(a) (b)

Abbildung 3.5: Aktuelle mit SDKs mitgelieferte Entwicklungsumgebungen (a) *UnrealEd* und (b) *CryENGINE 3 Sandbox*. Bildquellen v.o.n.u.: [31] und [32].

erzeugt. Dies umfasste Planeten, Namen, politische Systeme, Ökonomien, Handelsrouten und Raumstationen. Der Umfang der Spielwelt wurde auf eine neue Ebene gesetzt. Das Universum und die Objekte in diesem konnten durch den Spieler frei bereist werden und die Wahl des Spielzieles selbst getroffen werden. Dem Spieler stand es frei Handel oder Piraterie zu betreiben oder sich als Kopfgeldjäger zu versuchen [44, 47, 71].

Die 80er Jahre brachten weitere Vertreter der *Open-World Games* hervor. So führte z. B. *Mercenary* (1985, Novagen Software) ein *Elite*, mit dem

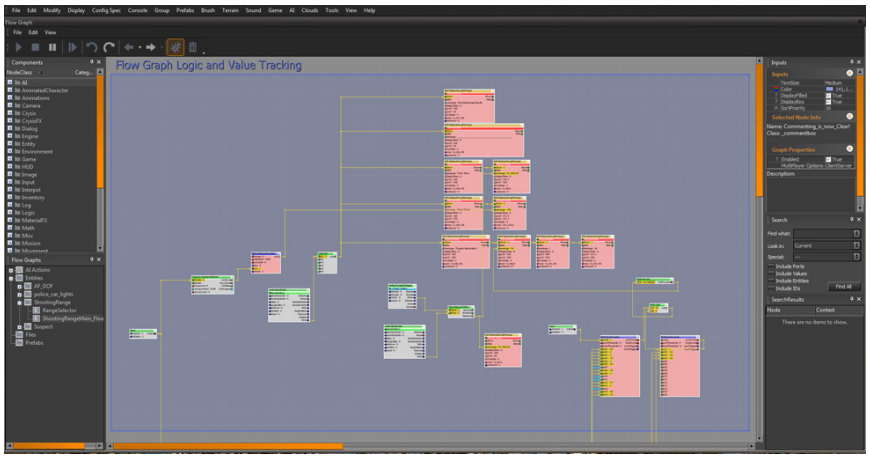
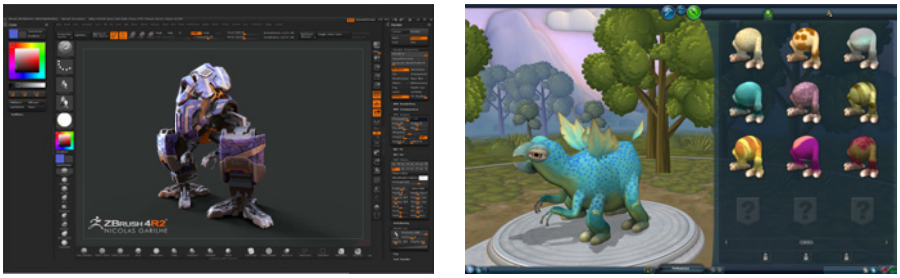


Abbildung 3.6: Der *Flow Graph Editor* innerhalb der *CryENGINE 3 Sandbox* ermöglicht das Erstellen von Abläufen und Verknüpfungen zwischen Objekten ohne eine Zeile Quellcode zu verfassen. Bildquelle: [33].



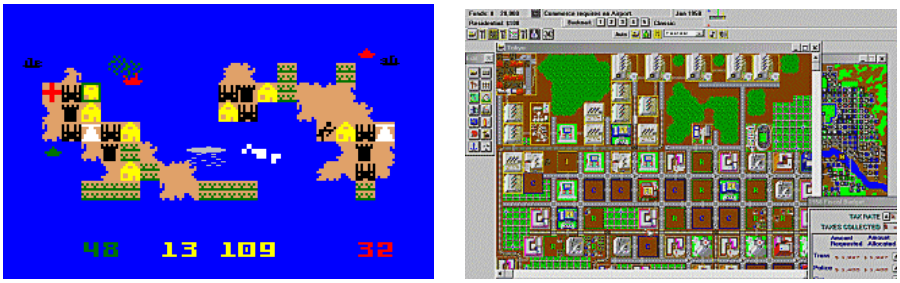
(a) (b)

Abbildung 3.7: Vergleich Modeling von Figuren in *ZBrush 4R2* (a) und dem *Spore Labor Basisversion* (b). Bildquelle links: [34].

Schauplatz auf der Erde, neben der Spielerfreiheit auch eine nicht-lineare erzählerische Struktur ein, eine Vorankündigung von Spielen wie der *GTA*-Serie [71].

3.2.2 Anfänge von UGC in Games

Im Sinne des freien Spielens wie es in Open-World Games existiert, entstand Anfang der 80er Jahre mit dem Pionier *Utopia* (1981, Don Daglow) das Genre der *Building Games* (siehe Abb. 3.8), deren Innovation das Miteinbeziehen des Spielers in die Gestaltung des Games darstellt. Inbesondere *Will Wright* erkannte das Potential hinter dem Vergnügen Spielwelten zu gestalten und schuf die *Simcity*-Reihe. Andere Entwickler folgten diesem Trend und kombinierten weitere Elemente z. B. wirtschaftliche Elemente mit dem Konzept



(a) (b)

Abbildung 3.8: UGC in Games durch die aktive Einbeziehung des Spielers in die Gestaltung der Spielwelt wie z. B. in den *Building Games Utopia* (a) und *Simcity* (b). Bildquellen v.l.n.r.: [35] und [36].

der *Building Games*. Die Anfänge Spieler am Design der Games teilhaben zu lassen kündigten Games wie *Little Big Planet*[13] (2008, Media Molecule) (LBP) oder *Spore* an [44].

3.2.3 Sandbox Games

In [44] werden neben Multiplayer und dem Schritt von 2D zu 3D in Computer Games, *Sandbox Games* als dritte bedeutende Innovation in der Games Industrie bezeichnet. Der Begriff, Genre und Konzept der *Sandbox Games* entstammt aus den Open-World Games wurde jedoch erst im Zusammenhang mit Titeln wie *GTA 3* oder auch der *Sims*-Serie durch die Entwickler als solche benannt. Wie die Open-World Games bieten diese eine offene Spielwelt, die der Spieler frei erkunden kann. Der Spieler kann sich die Ziele selbst wählen. Die wesentliche Innovation liegt jedoch im Grad der Interaktion zwischen Spieler und Spielwelt [44, 48]. Brian Baglow, CEO von *Indoctrimat PR*, ehemaliger Mitarbeiter von *DMA Design* dazu in [48]:

> Up until that point, with very few exceptions, the game world was passive, or at least offered only minimal interaction with the player such as falling blocks, rising spikes or, if you were very lucky, swinging ropes, ...

Das Konzept der freien Bewegung und Entscheidung und nicht-linearer Geschichte innerhalb der Spielwelt wurde durch einen gro en Schritt im Design Detail und der Fülle an Interaktionsmöglichkeiten erweitert. Der Spieler wurde zum verstärkten Experimentieren mit der Spielwelt motiviert. Die Spielwelt wurde z. B. in *GTA 3* ein aktiver Teil des gesamten Gameplays, die auf den Fortschritt des Spielers und dessen Aktionen reagiert. Glaubhafte und selbst-motivierte Charaktere übernehmen einen wesentlichen Teil dieser

[13] http://www.littlebigplanet.com

verstärkten Interaktion [44, 48]. So entwickelt sich z. B. in *The Sims* die Story aus der Interaktion mit den Charakteren, eine Erfahrung die bei jedem Spieldurchgang unterschiedlich ausfällt [38].

Für *Sandbox Games* existiert kein richtiger Weg diese zu spielen, das Game gibt nur wenig Richtung vor [43]. Wie in [44] argumentiert wird reicht es jedoch nicht aus den Spieler in eine Sandkiste mit Sand zu setzen. Das Spiel muss auch gewisse Regeln, Vorgaben und Hilfestellungen, d. h. ein Framework besitzen um die Sandkiste für den Spieler interessant zu gestalten.

Ein aktuelles Beispiel eines *Sandbox Games* ist *Minecraft Beta 1.8.1*. Es bietet eine prozedural generierte Spielwelt bestehend aus Blöcken mit verschiedenen Eigenschaften wie z. B. Erde, Holz, Stein oder Wasser. Der Spieler kann sich innerhalb der Spielwelt frei bewegen, die Spielwelt durch Bergbau oder Setzen von gesammelten Blöcken modifizieren. Neue Blöcke können durch ein Crafting-System erzeugt werden. Das Spielziel z. B. Erkunden, Sammeln oder Bauen stellt sich der Spieler selbst. Hilfestellung vom Spiel ausgehend existiert nicht, ebenso gibt es keine Vorgaben bezüglich dem Spielende. Das Spielsystem legt lediglich die notwendigen Regeln, z. B. einen Tag- und Nachtwechsel oder Monster in der Nacht, fest. Die Erfahrung des Spielers wird dadurch interessanter gestaltet.

3.2.4 Play, Create und Share Genre

Das Modding und die resultierenden Mods (siehe Abschn. 3.1.1) entwickelten sich durch die User und zuletzt durch die reagierenden Entwickler zu einem wichtigen Bestandteil der Games Industrie. Das Modding als auch die Bedienung von externen Editoren wie z. B. der *CryENGINE 3 Sandbox* verfügt in sich selbst Parallelen mit dem Spielen von Sandbox Games, z. B. das Festlegen eigener Ziele. Die Grenze zwischen Modding und Spielen wird zunehmend ie end. Die Verbindung zwischen Entwickler und User entwickelte sich weiter, so in [44]:

> Ten years ago, one would be wise to remark that the future of gaming is modding. But over the course of the past decade, *modding* itself has become increasingly part of *playing* the game, and the line between playing and modding is now and forever blurred. From the simplest scenario editors of the late 1990s through *Neverwinter Nights* modding tools, to Crytek s *Sandbox*, game production has increasingly focused upon in enabling and encouraging player design, and today s games often present certain forms of design as a core ingredient of the gameplay.

Was die Anfänge von UGC in Games (siehe Abschn. 3.2.2) durch die Building Games bereits ankündigte wurde mit der Veröffentlichung von *Spore* auf eine neue Ebene gesetzt. Durch die umfangreiche Verwendung von prozeduralen Algorithmen konnte *Spore* durch seine Editoren (Creature, Building, Vehic-

le) eine zuvor nicht dagewesene kreative Einbindung des Spielers erwirken. Jedem Spieler wurde es durch die auch für Anfänger geeigneten integrierten Modeling Editoren ermöglicht sich im Rahmen seiner kreativen Möglichkeiten in das Spiel einzubringen. Es basiert im Wesentlichen auf der Idee von *Will Wright* Teile des modernen Game Designs und dessen Spass bei der Durchführung als Teil von gutem Gameplay umzusetzen [43, 44]. Bei der Auswahl des gestalterischen Looks von *Spore* wurde dabei im Speziellen auf die Möglichkeiten der User eingegangen [46].

Der Begründer und erster Vertreter des in dieser Arbeit titulierten *Play, Create und Share Genres* ist *Little Big Planet*. Wie in Abschn. 2.1.5 bereits erwähnt entspringt der Name des Genres einer umfangreichen Marketing-Kampagne des *PlayStation 3*[14] Herstellers *Sony*. Wie aus dem Namen hervor geht bilden das Spielen (*Play*), das Erstellen (*Create*) als auch das Verteilen (*Share*) die drei Säulen des Genres. Das Genre kann durch die Verkaufserfolge und den Nachfolger *Little Big Planet 2* (2011, Media Molecule), als auch *ModNation Racers*[15] (2010, United Front Games) und *Joe Danger*[16] (2010, Hello Games) die den ähnlichen Kernpunkten folgen als etabliert angesehen werden.

Media Molecule versuchte die Freude am Erstellen von neuen Inhalten in ein Game zu integrieren [58]. Die Funktionalität des *Play, Create und Share* Konzeptes auf Konsolen ist u. a. stark abhängig von der Qualität der Editoren die die Entwickler den Spielern zur Verfügung stellen [52]. Die Besonderheit des Genres als auch im Speziellen die Innovation in *LBP* liegt in der Meisterung eines für Konsolen einfach zu bedienenden Editoren-Systems (siehe Abb. 3.9) und der schnellen Möglichkeit erstellte Levels über das Internet zu verteilen und runterzuladen. Zur unterhaltsamen und einfachen Bedienung des Editors wurde der Charakter als Brücke zwischen User und Editor integriert. Die Spieler haben im Editor die grafische als auch textuelle Freiheit ihre Kreationen zu erstellen. Die Filterung von Inhalten geschieht durch die Community via Rating-Mechanismen und Flagging-Mechanismen [45].

Der erwähnte Nachfolger *LBP 2* ermöglicht nicht nur das Erstellen von neuen Levels, sondern die Erstellung von neuen Games, wie z. B. Racing Games, Multiplayer Shooter oder RPGs [61]. Mit *Create*[17] (2010, EA Bright Light) schaffte *Electronic Arts* einen 3D-Puzzler mit der Möglichkeit des Spielers sich durch Editoren unterstützte kreative Puzzle Lösungen zu erarbeiten [40]. Mit dem *Mission Architect* für *City of Heroes*[18] (2004, Cryptic Studios) kann der Spieler das Game durch eigene Missionen mit eigenen Charakteren und Story aufwerten. Ein Vorteil für MMORPGs, deren Pro-

[14] http://www.playstation.com
[15] http://www.modnation.com
[16] http://www.hellogames.org
[17] http://www.ea.com/create
[18] http://na.cityofheroes.com

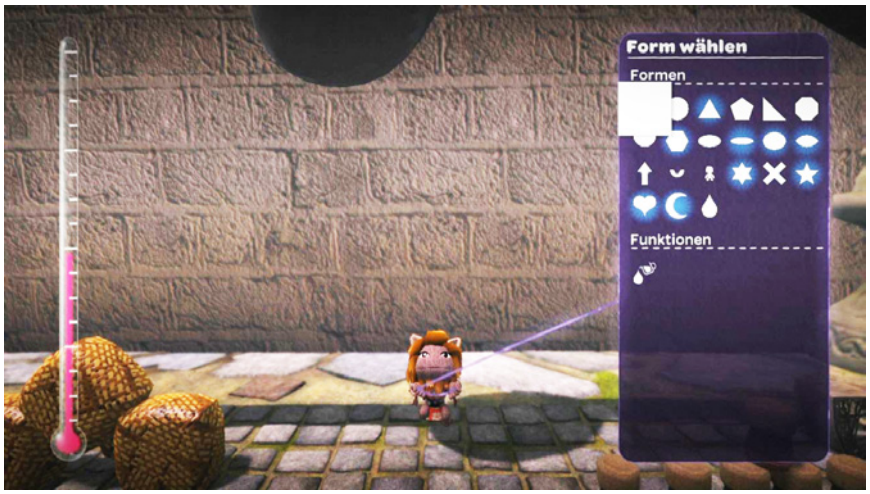


Abbildung 3.9: Im Editor von *Little Big Planet* auf der *Sony PlayStation 3* dient die Darstellung des Spielers als Schnittstelle zwischen Spieler und Editor. Bildquelle: [37].

duktionsaufwand primär in der Bereitstellung von genügend Inhalten liegt. Diese macht sich u. a. auch *inFAMOUS 2*[19] (2011, Sucker Punch Productions) mit einem *Optionalen Editor* zu Nutze. Die durch Entwickler erstellten Story-Missionen werden durch die User-Missionen ergänzt [53, 64, 74].

3.3 Zusammenfassung

Mit Beginn der Computer Games existierte der Wunsch der Spieler diese zu modifizieren und zu erweitern. Anfangs durch die technische Hürde nur den Hackern vorbehalten, erkannte die Games Industrie das Potential und ihre Vorteile durch *Mods* und stellte den Spielern o zielle Entwicklungsumgebungen z. B. *CryEngine 3 SDK*, *Source SDK* und *Unreal Development Kit* zur Verfügung. Durch die Entwicklung des Internets und der besseren Verbreitungsmöglichkeiten gewann UGC zunehmend an Bedeutung. Mit dem Aufkommen des freien Spielens in *Sandbox Games* fand das Erstellen von Inhalten verstärkt den Weg in die Games als Teil der Spielerfahrung. Aktuelle Lösungen des *Play, Create und Share Genres* ermöglichen die Erstellung von z. B. Levels mit Hilfe umfangreicher In-Game Editoren und unterscheiden sich von anderen In-Game Editoren durch die unterhaltsame Umsetzung und die einfache und schnelle Verteilung und Beschaffung von UGC.

[19] http://infamousthegame.com

Kapitel 4

In-Game Editoren und Usability in Games

In den vorangeganenen Kapiteln wurde die theoretische Basis zur Festlegung des Forschungsgegenstandes gelegt. Dies geschah durch Definition der wichtigen Begriffe, historischen Bedeutsamkeit und dem aktuellen Stand im Kontext von UGC. Dieses Kapitel dient der genauen Abgrenzung des Forschungsgegenstandes als Vorarbeit der Analyse in Kap. 6. Mit Ausblick auf Kap. 5 *Forschungsfrage* wird dieses Kapitel in Abschn. 4.2 durch den aktuellen Stand von Usability in Games mit Bezug zum Forschungsgegenstand ergänzt.

4.1 Forschungsgegenstand: In-Game Editoren

Bevor in Kap. 5 auf die Problem- und Fragestellung eingegangen werden kann, bedarf es der genauen Festlegung des Forschungsgegenstandes der In-Game Editoren. Wie in *Editoren und SDKs* (siehe Abschn. 3.1.2) definiert sind Editoren im Allgemeinen Programme zur Erstellung und Modifikation von Daten. Aus dem Ort des Aufrufes von In-Game Editoren geht hervor, dass diese Teil eines Programmes, dem Game, sind. Der Spieler muss zur Benutzung des Editors das Game nicht verlassen.

Die Bedeutung des Editors für die Erfahrung des Spielers mit dem Game als auch die Bedeutung aus Sicht der Entwickler kann je nach Funktionsumfang und Integration in dem Game variieren. Ein im Umfang simpler Charakter-Editor (siehe Abschn. 2.1.2 *Parameter-based Creation*) zur Auswahl von Merkmalen der Figur und Kleidung kann in dessen Gewichtung für die Erfahrung des Spielers geringer eingestuft werden als die In-Game Editoren von z. B. *Spore*. Die Ergebnisse des In-Game Editors von Spore wirken sich auf das weitere Game aus. Ein anderes Beispiel sind die umfangreichen In-Game Editoren des Play, Create und Share Genres, die einen gro en Funktionsteil des beworbenen Produktes ausmachen und deren vorgesehene

Nutzung ein höheres Ma an investierter Zeit erfordert.

Die beiden Anwendungen von UGC im Kontext der Games durch In-Game Editoren offenbaren sich bei Betrachtung des Gesamtkonzeptes von *Spore* und *Little Big Planet*. In *Spore* sind die In-Game Editoren mit dem Erfolg des Spielers direkt verknüpft, d. h. der Spieler muss den Editor benutzen um im Game voranzuschreiten. Die Ergebnisse des Editors, die nicht ausschlie lich optischer Natur sind z. B. das Anbringen von Flügeln, können sich auf den weiteren Verlauf auswirken. In *LBP* ist die Bedienung des Editors durch den Spieler optional, da neben dem Editor, dem *Create*-Part, ein eigener *Play*-Part existiert. Dem Spieler bleibt es frei sich als Produzent von Inhalten zu betätigen oder diese nur zu konsumieren.

Die vorliegende Arbeit beschäftigt sich primär mit *In-Game Editoren des Play, Create und Share Genres*. Zur Beantwortung der Fragestellung (siehe Abschn. 5.1) werden allerdings *Integrierte Editoren*, u. a. jener von *Create*, zum Vergleich und Diskussion herangezogen. Der Fokus auf das Play, Create und Share Genre ergibt sich aus den erfüllten Eigenschaften des UGC (siehe Abschn. 2.1.1). Neben dem wichtigen Merkmal des *Creative E orts* wird hier im Gegensatz zu z. B. *Spore* auch das *Sharing* als explizite Funktion im Genre umgesetzt. Die Entwickler machen sich die produzierten Inhalte der Spieler gezielt zu Nutze. Der UGC ist damit vergleichbar mit dem des Web 2.0. Für den Entwickler ergeben sich dabei wie in den vorhergehenden Kapiteln beschrieben z. B. Vorteile wie die Verlängerung der Spieldauer durch neue Inhalte, damit verbundene erhöhte Verkaufszahlen oder wie gängige Praxis im Modding, das Rekrutieren von neuen Mitarbeitern.

Im Zuge der Unterstützung von In-Game Editoren und des UGC nach Web 2.0 entstehen bei der Entwicklung des Games eine Reihe funktioneller Anforderungen die mit dem UGC einhergehen. Diese dienen der Unterstützung des *Creative E orts* und des *Publication Requirements* (siehe Abschn. 2.1.1). Zugleich definieren diese beiden Eigenschaften die Funktionalitäten eines Games die im Fokus der Analyse stehen. Bekannte Funktionalitäten und Anforderungen werden daher in den folgenden zwei Abschnitten als *Creation* und *Sharing* zusammengefasst.

4.1.1 Creation

Wie in Abschn. 2.1.1 beschrieben bezeichnet der Creative Effort von UGC einen kreativen Aufwand der durch den Spieler bei der Erstellung oder Bearbeitung von Inhalten erbracht werden muss. Der *Create*-Part muss dem Spieler die Möglichkeit bereitstellen den Creative Effort von UGC zu erfüllen. Für In-Game Editoren wird dies häufig in Form von Tools, die z. B. *Atomistic Constructions* ermöglichen umgesetzt. Es handelt sich hierbei um Konstruktionen basierend auf einfachen Teilen durch deren Kombination komplexe Gebilde entstehen können [17, S. 14 15]. Zum Beispiel die einfachen Blöcke in *Minecraft* die sich zur Gestaltung von neuen Gebilden mit unterschiedli-

chen Funktionen verbinden lassen. Die Definition von *Creation* soll sich im Allgemeinen auf Grund möglicher ie ender Übergänge zwischen *Play*-Part und *Create*-Part auf jegliche Art von Erstellung oder Manipulation von Inhalten im Game beziehen. Im Speziellen stehen explizit In-Game Editoren, sofern als solche erkennbar, im Fokus.

4.1.2 Sharing

Das Publication Requirement definiert in Abschn. 2.1.1 muss erfüllt sein um von UGC in Form des Web 2.0 zu sprechen. Das Play, Create und Share Genre defininiert sich wie in Abschn. 3.2.4 beschrieben u. a. über die schnelle und simple *Sharing*-Funktionalität. Diese konnte erst durch die Integration des Internets auf der Konsole z. B. durch das *PlayStation Network* (PSN) für *PlayStation 3* erfüllt werden. Das *Sharing* stellt die Schnittstelle zwischen dem *Play*-Part und dem *Create*-Part her. Es kann als Teil des Erstellens und des Spielens angesehen werden und ist dadurch für In-Game Editoren bzw. *Creation* relevant. Neben dem Versenden von erstellten Inhalten ist auch die Integration der Inhaltsbeschaffung bei der Entwicklung und Konzeptionierung durch die Entwickler zu beachten. Die Moderation von UGC bleibt weiters ein Problem bei der Integration [56]. Dies erfordert u. a. ein *Rating*-System um neue Inhalte durch Spieler bewerten zu lassen oder ein *Tagging*-System ähnlich dem Web 2.0 um diese zu kategorisieren [45, 56]. Dies dient zum einen dazu qualitativ hochwertige bzw. gegenteilige Inhalte zu filtern und zum anderen User die viel Zeit in die Kreation investieren entsprechend zu belohnen. Sharing definiert zusammengefasst jegliche Mechanismen zur Verteilung, Erhalt und Bewertung von UGC im Game.

4.1.3 In-Game Editoren als Sandbox Games

Wie in Abschn. 3.2.4 angeschnitten und in [44] bereits artikuliert, ist es möglich Game Design an sich als *das Sandbox Game* zu bezeichnen. Im Falle der In-Game Editoren werden Teile des Game Designs, z. B. die Erstellung von neuen Levels, dem Spieler In-Game zur Verfügung gestellt. Es kann als eine Form von *In-Game Modding* bezeichnet werden, da ähnlich dem Modding diverse Spielinhalte im Rahmen der Möglichkeiten verändert oder erstellt werden. Es zeigen sich Parallelen in den Eigenschaften von Sandbox Games im Vergleich zu denen von In-Game Editoren (siehe Tab. 4.1). Dies eröffnet vermeintlich die Möglichkeit In-Game Editoren als Sandbox bzw. Sandbox Game in sich selbst zu bezeichnen, unabhängig ob der Editor selbst Teil eines Sandbox Games ist. Bei genauerer Betrachtung der Eigenschaften offenbart sich der Umstand, dass sich lediglich die Bedienung des Editors und die zur Verfügung gestellten Funktionen in einer sandboxartigen Form präsentieren. Zumindest die Bezeichnung als Sandbox Game ist auf Grund der Eigenschaften daher nicht zulässig. Es stellt sich die Frage nach den Eigenschaften

Tabelle 4.1: Überblick des Vorkommens der identi zierten Eigenschaften von Sandbox Games (siehe Abschn. 2.2.1) im Vergleich zwischen Sandbox Games und In-Game Editoren. Abgesehen von der Story die jeweils als optional zu betrachten ist, zeigt sich die Gemeinsamkeit durch die freie De nition eines Zieles durch den Spieler (*Open-World* bzw. *Open-Ended*) und die Unterstützung des Experimentierens durch das Reagieren auf die Eingaben und das Verhalten des Spielers (*Automated Responsiveness*). Das Entstehen neuer Verhaltensweisen (*Emergent Behaviour*) aus den simplen Regeln heraus kann mit dem ähnlichen Verhalten von simplen Tools verglichen werden.

Eigenschaften Sandbox Games	*Sandbox (Game)*	*In-Game Editor*
Open-World	x	x
Multi-Threaded oder Non-Linear Story	optional	optional
Automated Responsiveness to Player Behaviour	x	x
Emergent Behaviour	x	x (Tools)

eines Games bzw. unter welchen Umständen ein In-Game Editor zurecht als Sandbox Game bezeichnet werden könnte.

Games und Toys

Wie Breslin in [44] durch Anwendung einer Sandkisten-Metapher feststellte, genügt es nicht dem Spieler eine Sandbox bzw. einen In-Game Editor zur Verfügung zu stellen und diese als kreativ und unterhaltsam zu betrachten. Vielmehr benötigt der Spieler die Sandbox umgebende Rahmenbedingungen um über die Möglichkeiten Bescheid zu wissen und sich gemä den Eigenschaften von Sandbox Games die Herausforderungen selbst stellen zu können. Das dahinter liegende Framework soll die Präsentation von neuen Elementen steuern, wenn der Spieler in der Spielwelt voranschreitet und im Fall von In-Game Editoren neue Tools kennenlernt. In Sandbox Games wird dies sehr oft in Form von *Reward*-Systemen realisiert die z. B. neue Bereiche, Strukturen oder für In-Game Editoren weitere Tools und Items freischalten [44].

Passend dem Vergleich Sandbox und Sandbox Game kann eine durch Schell [20] durchgeführte Differenzierung zwischen *Toys* und *Games* zur weiteren Erklärung herangezogen werden. Das Spielen mit einem Toy soll aus eigenem Antrieb erfolgen. Ein gutes *Toy* wird bezeichnet als *object that is fun to play with* [20, S. 37]. Das Spielen mit einem *Toy* soll *Fun* (siehe Abschn. 4.1.4) erzeugen bevor mit dem Design eines Games rund um das *Toy* begonnen wird [20, S. 90]. Die einzelnen Werkzeuge im In-Game Editor können als Toy bezeichnet werden. Die Sammlung dieser bildet folglich der Analogie die Sandbox jedoch bilden Toys alleine noch kein Game. Games haben nach Schell im Gegensatz zu Toys konkrete Ziele und liefern eine reichere Erfahrung im Sinne des Problemlösens. Viele Games bauen auf *Toys*

auf. Zum Beispiel wird ein laufender und springender Avatar als *Toy* bezeichnet und *Donkey Kong* (1981, Nintendo) als das darauf aufbauende Game. Ebenso wäre Baseball das Game-Pendant zum Ball [20, S. 90]. Die Erfahrung im Sinne des Problemlösens entsteht durch die Eigenschaften von Games wie z. B. *Goals*, *Con icts*, *Rules* oder *Challenges* [20, S. 34 37].

Sandbox zu Sandbox Games

Wie bereits in Abschn. 2.2.1 festgestellt wurde obliegt es meist dem Spieler sich die Ziele (*Goals*) und Herausforderungen (*Challenges*) innerhalb gewisser Grenzen selbst zu definieren. Diese Grenzen können u. a. durch die *Rules* des Games definiert sein. Die *Rules* definieren den Raum, Objekte, Aktionen und deren Konsequenzen und Folgerungen sowie auf einer übergeordneten Ebene ebenfalls die Ziele eines Games [20, S. 144 146]. Unterliegt der Spieler im In-Game Editor z. B. Restriktionen von Raum oder verfügbaren Ressourcen die durch Rules definiert wurden, so könnte dieser auch als Sandbox Game bezeichnet werden.

4.1.4 Fun und Editoren

Im Allgemeinen werden Editoren bzw. Tools benötigt um UGC zu erstellen und zu verteilen. Externer UGC in Form von Modding unterscheidet sich in dieser Hinsicht nicht von UGC innerhalb von Games. Die *Usability* (siehe Abschn. 4.2) ist ein wichtiger Aspekt bei der Entwicklung von interaktiven Systemen bzw. Software [5]. Für In-Game Editoren wurde in Abschn. 4.1.3 festgestellt, dass diese als Sandbox Game in sich selbst bezeichnet werden können. Ein wichtiger Aspekt von Games und markanter Unterschied zwischen In-Game Editoren und externen Editoren wurde in den vorhergehenden Kapiteln gestreift, jedoch nicht weiter behandelt *Fun*. Koster [10, S. 40] beschreibt *Fun* als *A Source of Enjoyment* ausgelöst durch physikalische Stimulation, ästhetische Würdigung oder chemische Manipulation. Schell definiert in [20, S. 37] *Fun* als *Pleasure with Surprises*. *Fun* gilt als fast unabkömmliche Eigenschaft eines jeden Games [20, S. 26 27]. *Sony San Diego* Produzent Vernon Mollette in [52] über die PSP-Version (PlayStation Portable) von *ModNation Racers* bezugnehmend auf *Sandbox Games*, *Fun* und *UGC*:

> I think that creating the correct sandbox environment that enables people to have fun with the user-generated content is the biggest challenge [in creating a user-gen game].

Neben der korrekten Sandbox-Umgebung, also der Umsetzung der Anforderungen und Eigenschaften von Sandbox Games, ist besonders *Fun* mit dem UGC wichtig. Dieser ergibt sich u. a. aus einer gelungenen Umsetzung. Die Aussage bezieht sich auf die Erstellung von Inhalten in den In-Game

Editoren, die einfache und integrierte Art neue Inhalte zu erhalten und das eigentliche Spielen von UGC. *Fun* bzw. der Spa und die Unterhaltung sind ein wichtiges Unterscheidungsmerkmal im Sinne der *Usability* zwischen Anwendungssoftware und Games (siehe Abschn. 4.2).

Bevor die Problemstellung, die Fragestellung und die Zielsetzung in Kap. 5 ausformuliert bzw. erfasst werden können, ist es erforderlich den Unterschied zwischen *Usability in Anwendungssoftware* und *Usability in Games* zu erarbeiten. Dies geschieht im nächsten Abschnitt durch die geschichtliche Aufarbeitung, Aufbereitung der theoretischen Basis und der Methoden zur Bewertung der *Usability*.

4.2 Usability in Games

Dieser Abschnitt beleuchtet die Entwicklung der *Usability in Games*. Die Gliederung erfolgt durch die allgemeine Definition von *Usability*, dem Unterschied zwischen *Tool* und *Toy* im Sinne der Beweggründe des Zugangs und leitet über zu den *Heuristiken* bzw. *Heuristiken für Games*, einem Hilfsmittel zur Evaluierung der *Usability in Games*.

4.2.1 De nition Usability

Die Definition für *Usability* entstammt einem Standard für interaktive Systeme der *International Origanization for Standardization*[1] (ISO) als Teil des Standards ISO 9241 *Ergonomic Requirements for O ce Work*. Darin wird *Usability* in 9241-11:1998[2] definiert als *E ectiveness*, *E ciency* und *Satisfaction* mit denen spezifizierte User ein spezifiziertes Ziel in einer bestimmten Umgebung erreichen [5, S. 275 277].

E ectiveness: Die Genauigkeit und Vollständigkeit mit der ein spezifizierter User ein spezifiziertes Ziel in einer bestimmten Umgebung erreicht [5, S. 277].

E ciency: Die benötigten Mittel im Verhältnis zur Genauigkeit und Vollständigkeit der erreichten Ziele [5, S. 277].

Satisfaction: Der Komfort und die Annehmbarkeit des Arbeitssystems durch seine User und durch andere Personen die durch dessen Benutzung beein usst werden [5, S. 277].

Traditionell standen *E ectiveness* und *E ciency* im Vordergrund der Interaktion. Mit der zunehmenden Anzahl an interaktiven Systemen au erhalb der Arbeitsumgebung wurde die *Satisfaction* zum zunehmend wichtigeren Bestandteil des Begriffes *Usability*. Der User muss das System bedienen wollen und bedient es effektiver wenn er die Bedienung zunehmend als Vergnügen betrachtet [5, S. 156].

[1] http://www.iso.org

[2] http://www.iso.org/iso/catalogue_detail.htm?csnumber=16883

4.2.2 Tools und Toys

In [13] wurde der Versuch unternommen das Vergnügen von Games auf die Interfaces von *Tools* zu übertragen. Es wurde zwischen *Tools* und *Toys* als Nutzungsarten von Computersystemen unterschieden. *Toy*-Systeme z. B. Games werden aus eigener Motivation bedient und besitzen kein extern vorgegebenes Ziel. *Tool*-Systeme hingegen wie z. B. Text-Editoren werden als Hilfsmittel zum Erreichen eines extern vorgegeben Zieles bedient. Entgegen der von Schell [20] in Abschn. 4.1.3 vorgestellten Differenzierung zwischen Games und Toys differenziert Malone in [13] zwischen Tools und Toys auf der übergeordneten Ebene der Software. *Challenge*, *Fantasy* und *Curiosity* bilden dabei die Stützpfeiler der in [13] vorgestellten *Heuristics for Designing Enjoyable User Interfaces*. Die wesentliche Erkenntnis die daraus hervorgeht liegt in der Abgrenzung von *Toys* und *Tools* durch die *Challenge*, die Herausforderung die das System an den User richtet [13, S. 66]:

> In a sense, a good game is intentionally made di cult to play, but a tool should be made as easy as possible to use.

Schell beschreibt in [20, S. 32] *Challenge* in Games in ähnlicher Form. Gute Games haben die richtige Menge an *Challenge*, schlechte Games zu wenig oder zu viel [20, S. 26,31,37,90]. Der optimale Zustand zwischen Unterforderung z. B. Langeweile und Überforderung z. B. Frustration des Spielers wird als Konzept des *Flow*[3] (siehe [20, Kap. 9]) beschrieben. Weitere Gemeinsamkeiten zu Malone [13] finden sich in der Beschreibung des Zuganges zum *Toy*. So besteht die gemeinsame Meinung das die Benutzung des *Toys* aus eigenem Antrieb geschehen muss um es als *Play* zu bezeichnen.

Zusammenfassend gesprochen, soll bei einem *Tool* das Ziel möglichst einfach und schnell erreicht werden. Bei einem *Toy* bzw. Game soll es durch eine sich an die Fähigkeiten des Spielers anpassende *Challenge* erreicht werden. Das Spielen mit einem *Toy* soll unterhalten und Vergnügen bereiten. Der Spieler soll mit ihm aus eigenem Antrieb heraus und spontan spielen wollen. Das Game Design muss demnach entsprechend umgesetzt sein. Schell beschreibt in [20] so genannte *Lenses* zur Evaluierung des Game Designs. Dabei wird das Game durch verschiedene Perspektiven, die *Lenses*, betrachtet. Ein anderes Konzept zur Evaluierung der *Usability in Anwendungssoftware* bzw. der *Usability in Games* wird im folgenden Abschnitt beschrieben.

[3] Ein mentaler Zustand in dem äußere Einflüsse ausgeblendet werden und nur die aktuell durchgeführte Aktivität in das Bewusstsein dringt. Ein Zustand *totaler Fokussierung*, *Pleasure* und *Enjoyment* in dem jegliches Zeitgefühl verloren geht [20, S. 118-119].

4.2.3 Heuristiken und Usability in Games

Heuristiken sind *Guidelines*[4] oder generelle Prinzipien zur Unterstützung einer Design-Entscheidung oder zur Analyse von Entscheidungen die bereits getroffen wurden. *Heuristic Evaluation* ist ein exibler und ökonomischer Ansatz die *Usability*, *Funktionalität* und *Akzeptanz* eines interaktiven Systems zu testen. Sie wurde durch Jakob Nielsen und Rolf Molich entwickelt. Zur Analyse der *Usability* von interaktiven Systemen wurden von Nielsen 1994 *10 Usability Heuristiken* auf Basis von *Usability Prinzipien* und *Guidelines* verfasst (siehe [5, Kap. 7]). Prinzipiell können die *Heuristiken* durch zusätzliche für das Gebiet spezifische *Heuristiken* erweitert werden [5, S. 324 325].

Vergleichbar dem Versuch von Malone in [13] *Heuristiken* zum Design unterhaltsamer Interfaces aufzustellen, wurden durch Federoff [7] 40 *Heuristiken* und *Guidelines* zur Evaluierung von *Fun* in Video Games beschrieben. Es wurde hierbei u. a. belegt, dass sich die *10 Heuristiken von Nielsen* auf Games anwenden lassen, jedoch der essenzielle Bereich des Gameplays auf Grund der tool-orientierten Herkunft[5] der *Heuristiken* nicht zufriedenstellend abgedeckt werden konnte. Dies erklärt sich durch andere Design-Überlegungen die bei Games erforderlich sind und *Usability*-Probleme die auftreten können. *Satisfaction* als Teil der bereits gegebenen Definition von *Usability* erfährt in Games eine höhere Gewichtung als *E ciency* und *E ectiveness*. Wäre die Interaktion sehr e zient, wäre das Spielen durch das Fehlen der *Challenge* nicht unterhaltsam und langweilig. *E ektivität* lässt sich auf Grund eines optimalen Pfades oder auf Grund des fehlenden Endzieles nicht messen [7]. Die *Satisfaction* muss verstärkt werden da das Ziel die Unterhaltung und nicht die Produktivität des Spielers ist. Für Games drückt sich *Satisfaction* als *Fun* (siehe Abschn. 4.1.4), *Immersive Environments* und *Compelling Experiences* aus [7]. Federoff [7] bezeichnet *Immersive Environments* als die Möglichkeit des Spielers gänzlich in die Spielwelt einzutauchen. Game Interfaces müssten hierfür für den Spieler nicht sichtbar sein oder nicht wahrgenommen werden. In [7] wird eine Einteilung der *Heuristiken* in *Game Interface*, *Game Mechanics* und *Game Play* vorgenommen. *Game Interface* wird definiert als das Mittel z. B. Keyboard durch welches der Spieler mit dem Game interagiert. *Game Mechanics* definiert sich als die Physik des Games die sich durch eine Kombination von Programmierung und Animation entwickelt. *Game Play* ist der Prozess durch welchen der Spieler das Ziel des Games erreicht. Zusammengefasst wird die Kategorisierung, also die *Usability für Games*, unter dem Überbegriff *User Experience* [7]. *User Experience*, das Nutzungserleben,

[4] Guidelines finden im Design interaktiver Systeme Anwendung, wo auf Grund der Unvollkommenheit dahinterliegender Theorien keine Formulierung von spezifischen Standards möglich ist. Im Gegensatz zu diesen haben Guidelines einen anregenden Charakter und sind allgemeiner formuliert [5, S. 277].

[5] Heuristiken bezogen sich bei Games primär auf das Interface [7].

ersetzt zunehmend den Begriff *Usability*. Sie betont die ganzheitliche Sicht auf die subjektiv erlebte Produktqualität [8].

4.2.4 HEP und PLAY

Ein weiterer Schritt zur Erstellung von Heuristiken zur Evaluierung von Usability in Games wurde in [3] durch die *Heuristic Evaluation for Playability* (HEP) unternommen. *Usability* bzw. *User Experience* wird hier auch als *Playability* bezeichnet. Die Einteilung erfolgte hierbei in *Game Play*, *Game Mechanics*, *Game Story* und *Game Usability*. In dieser Studie wurden im Gegensatz zu [7, 13] die aufgestellten Heuristiken und deren verbundene Relevanz für die Bewertung mit den Ergebnissen von *User Studies* als Evaluierungsmethode verglichen. Es wurde festgestellt dass sich die *HEP* gut gegen frühe Fehler im Design verwenden lassen, insbesondere für allgemeine Game Prinzipien. Die Gemeinsamkeiten mit den Ergebnissen der *User Studies* brachten jedoch auch hervor, dass auf spezifische Verhaltsweisen, die nur durch Beobachtung des Spielers erkannt werden können, mit den *HEP* nicht evaluiert werden kann [3]. In [19] einem White Paper mit Beispielen bezüglich Heuristiken für Games, wird die schlechte Verwendungsmöglichkeit der Heuristiken von Federoff [7] in der Protoyping-Phase kritisiert. Weiters wird auch die vage Definition und das Fehlen von konkreten Beispielen zum Verständnis der *HEP*-Heuristiken bemängelt. Es wird auf weitere ergänzende Evaluierungsmethoden mit *Usability Professionals* verwiesen, wie *Personae*[6] und *Cognitive Walkthroughs*[7] oder wie bereits erwähnt *User Testing*[8] [19].

In [4] wurde eine erforderliche Generaliserung von *HEP* durchgeführt und in Form von *Principles of Game Playability* (PLAY) veröffentlicht. Es handelt sich dabei um eine verfeinerte Variante der *HEP*-Heuristiken. Sie können bereits früher in der Entwicklungsphase von Spielen angewandt werden. Die Heuristiken wurden für die drei Genres *Real-time Strategy* (RTS), *Action Adventure* und *First-Person Shooter* (FPS) generalisiert. Weiters wurde eine feinere Gruppierung vorgenommen. Die Intention dahinter bestand in der breiteren Anwendungsmöglichkeit der Heuristiken die dadurch besser je nach Anforderungen adaptiert oder angepasst werden können [4].

[6] Eine umfangreiche Beschreibung einer imaginären Person die die Kern-Benutzergruppe repräsentiert. Im Verlauf des Designs kann anhand dieser das Verhalten mit dem System evaluiert werden [5, S. 204].

[7] Es werden die notwendigen Schritte aufgestellt die der User über das Interface benötigt um eine bekannte Aufgabe zu erreichen. Die Schritte werden anschließend auf mögliche Usability-Probleme evaluiert [5, S. 321].

[8] User Testing evaluiert das Design mit den eigentlichen Usern des Systems und nicht durch den Designer oder Usability-Experten selbst. Die Teilnahme von Usern bei der Evaluierung erfolgt meist erst nach Fertigstellung eines Prototypen [5, S. 327]. Für die verschiedenen Methoden zur Evaluierung durch Teilnahme der User sei auf [5, Abschn. 9.4] verwiesen.

4.3 Zusammenfassung

Als Forschungsgegenstand und damit Gegenstand der Analyse in Kap. 6 wurden In-Game Editoren (*Creation*) und das *Sharing* definiert. In-Game Editoren wurden als Form von *Sandbox* bzw. unter gewissen Vorraussetzungen von z. B. Zielen oder einer gestellten Herausforderung als *Sandbox Game* bezeichnet. Das Sharing wurde als die Schnittstelle zwischen *Play* und *Create* bezeichnet. Der wesentliche Unterschied zwischen der *Usability in Anwendungssoftware* und *Usability in Games* liegt in der Gewichtung der *Satisfaction*. *Satisfaction* ist neben *E ectiveness* und *E ciency* ein Bestandteil der Definition von *Usability*. Anwendungssoftware und Games bzw. *Tools* und *Toys* unterscheiden sich durch die Motivation der Aufgabe und *Challenge*. *Fun* wurde als wichtiges Unterscheidungsmerkmal angeführt. Zur Evaluierung der Usability in Games wurden auf Basis der Usability in Anwendungssoftware bereits verschiedene Heuristiken aufgestellt u. a. die *HEP*-Heuristiken und die *PLAY*-Heuristiken. Auf Basis der Erkenntnisse aus diesem Kapitel wird im folgenden Kapitel auf die Forschungsfrage eingegangen.

Kapitel 5

Forschungsfrage

Dieses Kapitel befasst sich mit der Aufstellung und Definition der Forschungsfrage (siehe Abschn. 5.1). Dies erfolgt zunächst durch Umrandung der Problemstellung und Argumentation die zur konkreten Fragestellung überleiten. In Abschn. 5.2 wird basierend auf dieser die Aufstellung der Kriterien zur Beantwortung der Fragestellung durchgeführt.

5.1 Problem- und Fragestellung

Die Entwicklung des externen UGC durch Modding (siehe Abschn. 3.1) hat gezeigt dass sich ein Trend einer engeren Verknüpfung zwischen Editoren und dem Spielen abzeichnet. Durch die möglichst ökonomische Anwendung der externen Modding-Editoren für die Entwicklung der Games bzw. dem Modding, kann hier jedoch nach wie vor von einer durch *E ciency* und *E ectiveness* gewichteten *Usability* ausgegangen werden. Games integrieren auf der anderen Seite den Spieler zunehmend in die Erstellung von neuen Inhalten z. B. durch Levels oder machen die Erstellung der Inhalte zum integralen Bestandteil des Spieles z. B. *Building Games* (siehe Abschn. 3.2.2). Alex Evans, Mitbegründer von *Media Molecule*, prognostiziert in [58] für die Zukunft von Games UGC als ein erwartetes Feature und weniger als Bonus wie es aktuell meist gesehen wird. Insbesondere für geschlossene Platformen wie Konsolen mit Verbindung zum Internet oder mobile Devices bieten sich integrierte In-Game Editoren zur Erstellung und Ausnützung von UGC an. Traditionell war das Modding z. B. auf den Konsolen auf Grund des geschlossenen Systems nicht existent [16].

Es liegt im Interesse der Entwickler eine möglichst breite Masse an Spielern zu mobilisieren UGC zu erstellen. Dies dient zum einen um das ökonomische Risiko von Spielen die gänzlich auf UGC aufbauen zu minimieren und zum anderen die daraus resultierenden Vorteile, wie z. B. eine verlängerte Spieldauer durch die zusätzlichen Inhalte, zu nutzen.

Daniel Volk betrachtet in [23] *The Continuum of Co-Creation* (siehe

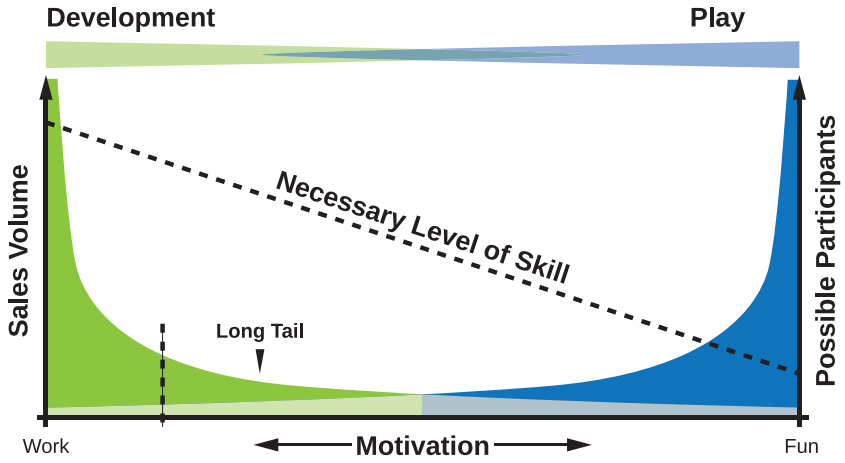


Abbildung 5.1: *The Continuum of Co-Creation* Bei Anwendung der Motivation (Work bis Fun) auf den Graphen des Volumens eines Games ergibt sich ein invertiertes Bild von diesem. Die Anzahl potentieller User steigt mit zunehmenden Fun als Motivation bzw. dem notwendigen Skill zur Bedienung der Tools. Durch Fun motivierte User erreichen in Summe ein gleichwertiges Volumen. Nach Bildquelle: [23, S. 264].

Abb. 5.1), das Verkaufsvolumen eines Games zusätzlich durch die Motivation der Entwickler dahinter. Diese reicht von Arbeit bis zu ausschlie lich *Fun*. Die Steigerung der potentiell aktiven Modder wird dabei in Abhängigkeit von einfach zu bedienenden Tools gesetzt die sich über den damit verbundenen Faktor *Fun* bzw. dem notwendigen *Skill* durch den User definieren. Wie sich zeigt, kann durch die hohe Masse an Usern die einen weichen Übergang zwischen Arbeit und *Fun* bilden in Summe ein gleiches Volumen erreicht werden wie durch die stärker durch Arbeit motivierten User, deren Produktqualität jedoch höher sein mag [23, 24].

Die Vorraussetzungen in der Modding-Community sind jedoch unterschiedlich als z. B. im Play, Create und Share Genre. Ein ähnlicher Zusammenhang zwischen *Fun* und *simplen Tools* lässt sich durch die *Stufen der Content Creation* (siehe Abschn. 2.1.3) als auch aus Interviews der Entwickler von *Little Big Planet* und *ModNation Racers* in [41, 52, 58, 63, 65] ableiten. In diesen wird vor allem darauf eingegangen, dass durch umfangreiche *User Tests* die Steigerung des *Fun* am Erstellen und der Qualität der Inhalte in Abhängigkeit von simplen Tools festgestellt wurde.

Qualität liese sich auch durch erhöhte Quantität erreichen. Nach *Sturgeon s Law* [73] (*Ninety percent of everything is crap*) folgt der Steigerung der Quantität an erstelltem UGC, linear die Steigerung der Quantität an qualitativ hochwertigen Inhalten. Bei rund 70000 [55] durch User erstellte Missionen in *inFAMOUS 2* könne durch die hohe Quantität von etwa 7000

qualitativ hochwertigeren Missionen ausgegangen werden. Dies ist weit mehr als ein Entwicklerteam innerhalb der selben Zeit erschaffen könnte. Für *City of Heroes* wurde innerhalb eines Tages eine Inhaltsmenge von fünf Jahren Entwicklungszeit des Entwicklerteam erreicht [39]. Zu beachten ist, dass die Qualität erstellter Inhalte subjektiv wahrgenommen wird. Nicht jeder Spieler besitzt das gleiche Ma an Kreativität [44]. Für UGC der in das Gameplay integriert ist hat dies keine Auswirkungen für den Entwickler da der Spieler mit dem UGC nur innerhalb seiner Spielwelt interagiert. Im Play, Create und Share Genre wird der UGC auch Teil der Spielwelt anderer Spieler. Die Inhalte mögen den Ansprüchen von anderen UGC Konsumenten nicht entsprechen. Die Filterung muss hier wie bereits in Abschn. 4.1.2 besprochen z. B. durch *Rating*-Mechanismen und *Tagging* geschehen.

Zur Überleitung zur Fragestellung wird aus den bereits erwähnten Interviews in [41, 52, 58, 63, 65], exemplarisch eine Aussage von Dan Sochan, Produzent von *ModNation Racers* in [41] aufgegriffen. Sie stand im Zusammenhang der Herausforderung bei der Entwicklung von Games mit UGC als zentralem Thema:

> The balance of creating editors and tools that are easy to pick up and play, that feel fun, but that also still have enough depth that people can create really impressive-looking [content], ...

Der Aussage folgend liegt die Herausforderung der Entwickler in der richtigen Gewichtung zwischen Editoren und Tools die simpel aufgebaut sind, leicht zu bedienen sind und zugleich die notwendige Kombinierungsmöglichkeit und Tiefe besitzen um die individuelle Kreativität der Spieler zu bedienen. Interessant ist die Verwendung von *Play* und *Fun* im Zusammenhang mit den Editoren und Tools. Dies lässt darauf schlie en dass ähnlich dem Versuch von Malone [13], Elemente aus Games zur Befriedigung der *Satisfaction* auch Anwendung in In-Game Editoren finden. Wie bereits festgestellt wurde kann die notwendige Kombinierungsmöglichkeit und Tiefe der Tools mit dem Merkmal von Sandbox Games (*Emergent Behaviour*) verglichen werden. Ein In-Game Editor kann als Sandbox Game in sich selbst bezeichnet werden. Die Aussage vermisst folglich die Bezeichnung der Umsetzung von Editoren und Tools als *Toy*. Für Sandbox Games typisch ist der eigene Antrieb des Spielers sich eine *Challenge* zu stellen. Der gesamte In-Game Editor kann als die Spielwelt (*Game World*) bezeichnet werden, die auch die Anforderung hat dem Spieler im Sinne des Sandbox-Gameplays neue mögliche Herausforderungen aufzuzeigen. Die *Toys* sind im Sinne der *Usability* nach wie vor Tools sollen den Spieler daher in der selbst gestellten *Challenge* unterstützen diese effektiv und e zient abzuschlie en. Bezugnehmend auf Toys nach Schell [20] (siehe Abschn. 4.1.3) und *Fun* und *Play* der Aussage, sollen die Toys den Spieler ebenfalls unterhalten. Ihre Summe und zusätzliche Regeln bilden das Game und damit einen Anreiz für den Spieler die Tools zu verwenden und Inhalte zu erstellen. Es stellt sich daher die Frage wie Tools und Editoren bzw. *Toys*

und Game miteinander vereint werden bzw. welche Elemente von Games sich im Editor wiederfinden. Daraus ergibt sich die folgende Fragestellung:

> *Welche für den Spieler unterstützenden und unterhaltsamen Ma - nahmen können zur Bedienung von In-Game Editoren ergri en werden?*

Die Frage befasst sich mit der Umsetzung *Fun* erzeugender Elemente z. B. *Toys* innerhalb des User Interfaces (siehe Abschn. 5.2.1) von In-Game Editoren insbesondere des Play, Create und Share Genres. Neben der *Satisfaction* sollen die Elemente auch aus Sicht der *Usability* betrachtet und interpretiert werden d. h. wie diese den Spieler in der Produktion von UGC unterstützen. Die Kriterien zur Beantwortung der Fragestellung werden im folgenden Abschn. 5.2 erarbeitet.

Zur Beantwortung der Fragestellung wurden folgende Nebenfragen formuliert und bereits in den vorangegangenen Kapiteln behandelt:

Kapitel 2: Wie wird UGC im Kontext der Spiele definiert und verwendet? Wie werden Sandbox Games definiert und welche Eigenschaften besitzen diese?

Kapitel 3: State of the Art von UGC im Kontext der Games? Wie stehen Modifications und Editoren in Verbindung? Wie stehen Sandbox Games und Editoren ìn Verbindung?

Kapitel 4: Welche Anforderungen entstehen dem Entwickler bei In-Game Editoren? Was ist Usability für Anwendungen und was für Games?

5.2 Orientierung der Analyse

Die Beantwortung der Fragestellung soll durch eine qualitative Analyse (siehe Kap. 6) von relevanten Games (siehe Tab. 6.1) erfolgen. Als Forschungsgegenstand werden wie in Abschn. 4.1 definiert die In-Game Editoren und ebenso die Sharing-Funktionalität betrachtet. Beide beziehen sich insbesondere auf Vetreter der Umsetzung von UGC als Form des Web 2.0 (Play, Create und Share Genre). Zusätzlich wird auch ein integrierer In-Game Editor analysiert und zum Vergleich herangezogen. Die Begründung darin liegt in der Annahme, dass dieser Editor möglicherweise stärker über Toys konzipiert wurde. Die Umsetzung möge daher verstärkt auf Unterhaltung und spielerisches Betätigen ausgelegt sein als Vetreter des Play, Create und Share Genres, deren Benutzung optional ist.

Games sind ein interaktives Medium. Die Interaktion zwischen User und Game erfolgt über das User Interface des Games. Die Analyse soll daher aus Identifikation und Interpretation von Ma nahmen im User Interface bestehen. Gemeint sind hierbei Ma nahmen die zur Erzeugung von *Fun* bzw. Erfüllung der *Satisfaction* beitragen könnten und möglicherweise gleichzeitig den User in seiner Tätigkeit der Erstellung von UGC unterstützen. Es geht

hierbei um Ma nahmen die üblicherweise in Anwendersoftware nicht zu finden sind und speziell im Medium der Games gebräuchlich sind um den Reiz den Editor zu *spielen* zu erhöhen.

Zunächst werden zur Unterstützung der Analyse die Verwendungsarten von User Interfaces in Games im folgenden Abschn. 5.2.1 kurz zusammengefasst. Abschn. 5.2.2 befasst sich mit den Bestandteilen eines Games die Ein uss auf das User Interface haben.

Darauffolgend werden in Abschn. 5.2.3 auf Basis der *PLAY*-Heuristiken für die Analyse notwendige Orientierungspunkte unter den Aspekten der *Usability in Games*, *User Interfaces in Games* und *Toys* ausgearbeitet und in Fragestellungen formuliert.

5.2.1 User Interfaces in Games

Die Interaktion ist der Dialog zwischen Mensch und Computer. Die Art des gewählten Interfaces beein usst die Qualität und Nutzung des Dialoges [5, Kap. 3.5]. Für Games ist folgend der Meinung von Moore in [62] das User Interface das Game und das Game das User Interface. Eine korrektere Bezeichnung wäre alles vom Game präsentierte ist das User Interface. In Games existiert das User Interface durch eine Vielzahl verschiedener Formen z. B. *Audio*, *Animation*, *Head-up-Display* (HUD) [42]. Auch kann die *Game World* z. B. der Level selbst als Interface fungieren [62] und somit den Spieler stärker in das Spielgeschehen einbinden. Es existieren verschiedene Ebenen der Verschmelzung zwischen Interface und Game World (siehe Abb. 5.2) die in [6] aufgestellt und in [42] zusammengefasst wurden:

Diegetic: User Interface dass innerhalb der *Game World* existiert. Es kann durch Charaktere innerhalb der *Game World* gehört und gesehen werden [42].

Non-diegetic: User Interface dass au erhalb der *Game World* existiert und nur für reale Spieler hörbar und sichtbar ist [42].

Spatial: User Interface Elemente die im 3D-Raum der *Game World* angezeigt werden unabhängig ob sie als *Diegetic* oder *Non-diegetic* einzuordnen sind. Als Beispiel wurden hier die Charakter-Outlines von *Left 4 Dead*[1] als *Non-diegetic Spatial* User Interface angeführt [42].

Meta: Repräsentationen die in der *Game World* existieren können, aber nicht notwendigerweise als *Spatial* User Interface für den Spieler visualisiert wurden. Dies sind *Meta Representations* wie z. B. Blutspritzer auf der Kamera [42].

Die Analyse beschäftigt sich mit den virtuellen User Interfaces von Games und lässt dabei die physischen Interfaces wie Keyboard und Controller au er Acht. Das User Interface in Games kann durch die Bestandteile *Aesthetics*, *Mechanics*, *Story* und *Technology* geprägt sein (siehe Abschn. 5.2.2).

[1] http://www.l4d.com

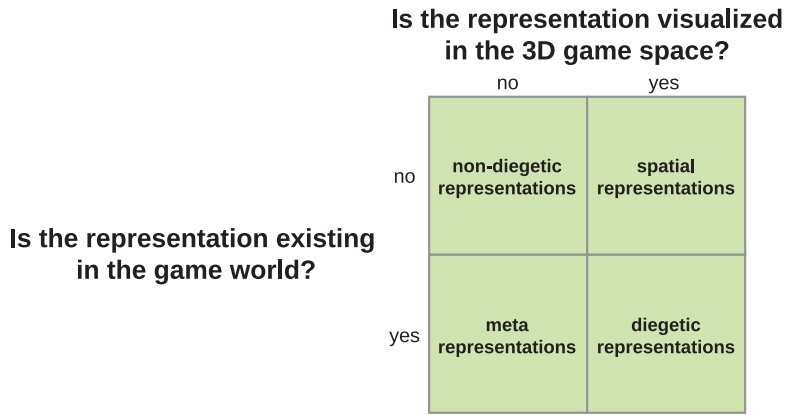


Abbildung 5.2: Repräsentationsformen von User Interfaces in Games. Nach Bildquelle: [42]

5.2.2 Mechanics, Story und Aesthetics

Wie Schell in [20] feststellte bestehen Games u. a. aus *Toys* und der Zugabe von aufgestellten Regeln. Die Präsentation für den Spieler erfolgt über das User Interface des Games. Die durch Schell [20, S. 41 46] definierten vier Bestandteile eines Games sind *Aesthetics*, *Mechanics*, *Story* und *Technology*, wobei alle vier als gleichwertig und wichtig zu betrachten sind. *Aesthetics* sind für den Spieler die sichtbarsten Bestandteile und daher aus Sicht des User Interfaces und der Analyse besonders relevant. Die dahinterliegende *Technology* soll in der Analyse nicht betrachtet werden.

Mechanics: Beschreiben die Abläufe und Regeln des Games und sind damit ein entscheidener Differenzierungspunkt zu z. B. Filmen oder Büchern. *Mechanics* müssen durch die *Technology* unterstützt werden, durch *Aesthetics* vorgehoben werden und durch die *Story* muss ihnen Sinn verliehen werden [20, S. 41].

Story: Die *Story* kann vordefiniert sein oder sich wie z. B. in Sandbox Games in verschiedene Stränge aufteilen oder sich durch das Spielen entwickeln. Es gilt ähnlich den *Mechanics*, dass die *Story* durch die anderen Bestandteile *Aesthetics*, *Mechanics* und *Story* unterstützt werden muss [20, S. 41 42].

Aesthetics: *Aesthetics* haben den grö ten Ein uss auf die Erfahrung des Spielers, da diese für den Spieler am sichtbarsten sind. Sie definieren den *Look*, *Sounds* und *Feeling* des Games. Es liegt hier an den anderen Bereichen *Mechanics* und *Technology* die durch *Aesthetics* etablierte *Story* optimal zu unterstützen [20, S. 42].

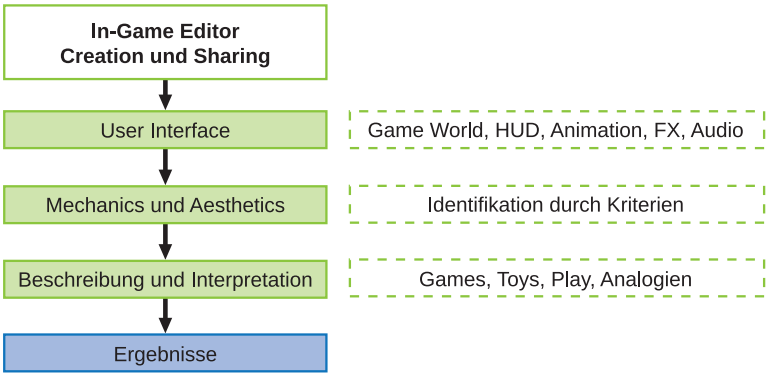


Abbildung 5.3: Veranschaulichung des Analysevorganges von In-Game Editoren und Sharing-Funktionalität. Unter Betrachtung des User Interfaces werden anhand festgelegter Kriterien *Mechanics* und *Aesthetics* identi ziert die zur Usability beitragen und im Sinne der Satisfaction besonders markant für Spiele sind. Die identi zierten Kriterien werden beschrieben und dienen als Basis zur Bildung des Ergebnisses.

5.2.3 Analysevorgang und Kriterien

In-Game Editoren werden in der Analyse als Editoren mit spielanregenden Ma nahmen betrachtet. Wie festgestellt wurde können sie als Sandbox Games angesehen werden. Ziel ist die Identifikation jener Ma nahmen im User Interface die zugleich den Spieler in der Erstellung und Sharing von UGC unterstützen. Es sollen die dahinterliegenden Design-Konzepte von *Aesthetics*, *Mechanics* und *Story* extrahiert werden und anhand von Vergleichen mit anderen Games, Toys, dem *Play*-Part oder Analogien zur echten Welt beschrieben und interpretiert werden. Der Bestandteil der Story dürfte zu vernachlässigen sein, da die Story gemä Sandbox-Gameplay sich erst durch die Interaktion und die selbst gestellten *Challenges* entwickelt. Der Analysevorgang ist in Abb. 5.3 zur Verdeutlichung abgebildet.

Zur Identifikation sind Analysekriterien notwendig die zur Orientierung dienen sollen. Es werden hierfür Kriterien benötigt die Zugleich die Eigenheiten der Satisfaction in Games behandeln, als auch Orientierung im Sinne der klassischen Usability liefern Kriterien der *User Experience*. Die *PLAY*-Heuristiken (siehe Abschn. 4.2.4), eine Sammlung von Heuristiken zur Evaluierung der Usability in Games, werden als Ausgangspunkt zur Aufstellung der Analysekriterien herangezogen. Zunächst wurden die relevanten *PLAY*-Heuristiken betreffend In-Game Editoren bzw. Sandbox Games gefiltert und anschlie end eine Zuordnung und Generalisierung ausgeführt. Durch die sich teilweise überlappenden Heuristiken bzw. die breite Interpretationsmöglichkeit dieser konnte es zu Mehrfachzuordnungen kommen. Anhand der mar-

kantesten Zuordnungen wurde eine Kategorisierung in *Allgemeine Kriterien*, *User Support und Terminology*, *Feedback*, *Progress und Status*, *Perception of Control* und *Customization* getroffen. Es wurde dabei darauf geachtet die Benennung aus Sicht der Tools durchzuführen. Die Zuordnungen beinhalten gemä den *PLAY*-Heuristiken auch Heuristiken betreffend Gameplay. Diese wurden im nächsten Schritt relevant.

Anhand der zugeordneten Heuristiken wurde eine Beschreibung der Kategorien in einer Soll-Form formuliert. Auf Basis dieser wurden letztendlich Fragestellungen unter den Gesichtspunkten von *Creation* (In-Game Editoren) und *Sharing* (Sharing-Funktionalität) formuliert. Die Fragestellungen beziehen sich auf die Umsetzung und Anwendung des User Interfaces im Analyseobjekt. Dieser Fragenkatalog bildet in seiner Summe die Analysekriterien bzw. die notwendige Orientierung zur Durchführung der Analyse. Auf Grund der Generalisierung und Eigenheiten von In-Game Editoren müssen und können nicht alle Fragen bzw. Kritieren zu jedem Analyseobjekt beantwortet werden. Die Kriterien sollen zur Identifikation der Ma nahmen dienen, die in der Analyse letztendlich beschrieben und interpretiert werden.

Allgemeine Kriterien

Dieser Punkt stellt den anfänglichen und einführenden Orientierungspunkt dar. Er befasst sich mit allgemeinen Formulierungen um zum einen das Analyseobjekt als Ganzes zu erfassen und zum anderen etwaige sonstige Auffälligkeiten festzustellen.

- Wie ist der allgemeine subjektive Eindruck des Screen-Layouts in Bezug auf Effectiveness und E ciency?
- Existieren besonders hervorstechende oder humoristische Umsetzungen die in den weiteren Kriterien nicht abgedeckt wurden?

User Support und Terminology

Die Fragestellungen befassen sich mit der Hilfestellung durch das User Interface. Der Spieler soll neue Fähigkeiten rechtzeitig lernen, spätestens bevor diese benötigt werden. In jeder Situation sollen dem Kontext entsprechende Hilfestellungen angeboten werden. Alle Spieler, unabhängig ob Anfänger oder Fortgeschrittener, sollen durch Tutorials schnell mit dem Game vertraut gemacht werden. Dem Spieler sollen die Spielziele bekannt sein und Auswirkungen bei deren Erreichen bekannt sein. Neue Spielziele sollen während dem Spielen früh genug eingeführt werden.

- Welche Formen der Hilfestellung erhält der Spieler und wie werden diese präsentiert?
- Fungiert der Play-Part als Tutorial und Einführung der Spielziele?
- Wie äu ert sich das Erlernen von neuen Fähigkeiten?

- Wie werden Spielziele repräsentiert?

Feedback

Das Game soll in jeder möglichen Form auf Aktionen des Spielers reagieren. Feedback soll unmittelbar nach der Aktion in spannender Weise gegeben werden. Das Game soll seine audiovisuellen Möglichkeiten in geeigneter Art und Weise nutzen.

- Wie äu ert sich Feedback ausgelöst durch Aktionen des Spielers im User Interface?

Progress und Status

Die Fragestellungen befassen sich mit der Darstellung des Spielfortschrittes und des Status im User Interface. Das User Interface soll den Spieler im Vermeiden von Fehlern unterstützen. Der Forschritt des Spielers soll nicht durch wiederholte Bestrafung von Fehlern unterbrochen werden. Erreichte Ziele, Status und Besitz sollen bestehen bleiben. Nach dem Einschalten des Games soll der Spieler genug Informationen besitzen um mit dem Spielen beginnen zu können. Der Spielstand soll in verschiedenen Zuständen des Games gespeichert werden können, sodass ein Unterbrechen des Spielens nicht zum Verlust des Fortschrittes führt. Das Game soll leicht zu lernen und schwer zu meistern sein. Der Schwierigkeitsgrad soll sich den Fähigkeiten den Spielers anpassen. Herausforderungen sollen positive Erfahrungen sein. Das Game soll Belohnungen verteilen um den Spieler in das Game eintauchen zu lassen z. B. durch neue Fähigkeiten, Fertigkeiten oder Gegenstände. Die Präsentation des aktuellen Status soll sich nahtlos in das Spielgeschehen integrieren.

- Wie werden Fehler durch den Spieler verhindert?
- Wie können Fehler rückgängig gemacht werden?
- Wird der Spieler bei Fehlern bestraft?
- Wie wird der aktuelle Status repräsentiert?
- Wie ist das Speichern und Laden umgesetzt?
- Verändert sich das User Interface mit zunehmenden Fähigkeiten?
- Stellt das Game Herausforderungen an den Spieler?
- Wie wird der Spieler bei erreichten Herausforderungen belohnt, wie äu ert sich dies im Interface?
- Haben erzielte Herausforderungen des Play-Parts Ein uss auf die Möglichkeiten im Create-Part?

Perception of Control

Der Spieler soll das Gefühl der Kontrolle besitzen und wissen welchen Ein uss seine Aktionen auf die Game World haben. Die Game World reagiert auf den Spieler und merkt sich den beschrittenen Weg. Veränderungen an der Spielwelt durch den Spieler sollen erkennbar und nachvollziehbar sein. Es soll zwischen Spielwelt und Spieler eine emotionale Verbindung herrschen. Die Steuerung des Games soll gleich bleibend sein und Standardkonventionen folgen.

- Wie ist die Repräsentation des Spielers innerhalb der Game World umgesetzt?
- Spiegelt sich das Verhalten des Spielers durch seinen Avatar im User Interface wider bzw. kann die Game World bein usst werden ohne explizit Aktionen auszuführen?
- Gibt es Unterscheidungen in Konventionen und Steuerung zwischen Play- und Create-Part?

Customization

Das Spiel soll verschiedene Spielstile und Wege das Ziel zu erreichen unterstützen. Die ersten zehn Minuten sollen extrem einfach sein und positives Feedback für alle Spielertypen geben. Das Interface soll einfach zu lernen sein, jedoch genug Anpassungsmöglichkeiten besitzen um den Anforderungen fortgeschrittener Spieler zu entsprechen. Es sollen weitere Anpassungsmöglichkeiten oder z. B. Gegenstände durch Belohnungen für das Erreichen von Zielen geben. Der Schwierigkeitsgrad soll adjustierbar sein.

- Wie wird in den ersten zehn Minuten das positive Feedback durch das User Interface unterstützt? Gibt es Veränderungen nach einer gewissen Spielzeit?
- Hat der durch den Spieler gewählte Spielstil Ein uss auf das Interface?
- Ist der Schwierigkeitsgrad justierbar? Wie?

5.3 Zusammenfassung

Der Wunsch aus Sicht der Entwickler liegt in einer Erhöhung der Quantität als auch Qualität von UGC. *Fun* beim Erstellen ist ein wesentliches Kriterium dafür. Die Fragestellung beschäftigt sich mit der Identifikation und Interpretation von Ma nahmen innerhalb des Forschungsgegenstandes die zugleich unterstützend als auch unterhaltsam sind und als typisch für Games zu betrachten sind. Die für die Analyse notwendigen Kriterien bzw. Orientierungspunkte wurden in Form eines Fragenkataloges erarbeitet. Als Basis für diesen wurden die *PLAY*-Heuristiken verwendet und durch Zuordnung und Generalisierung dieser die jeweiligen Kriterien entwickelt.

Kapitel 6

Analyse von In-Game Editoren und Sharing

Die folgenden Abschnitte dieses Kapitels beschäftigen sich mit der Analyse von ausgewählten Games. Die Analyse orientiert sich Anhand der in Abschn. 5.2.3 festgelegten Kriterien bzw. ausgearbeiteten Fragestellungen. Die Zuordnung der Ergebnisse erfolgt nach der in Abschn. 4.1 getroffenen Einteilung in *Creation* und *Sharing*. Die Auswahl an Games (siehe Tab. 6.1) bezieht sich primär auf Vertreter des *Play, Create und Share Genres* auf der PS3-Platform. Zur Ergänzung, Vergleich und Erhalt übergreifender Ergebnisse wurde zusätzlich ein Vertreter von *Integrierten In-Game Editoren* ausgewählt. Die Benutzung ist im Gegensatz zu den anderen Editoren nicht optional. Ziel dieses Kapitels und der Analyse ist die Identifikation von den Spieler unterhaltenden und unterstützenden Ma nahmen bei der *Creation* und dem *Sharing*, Ma nahmen die sich typischerweise von Anwendersoftware und externen Modding-Tools unterscheiden. Die Interpretation und Zusammenfassung der identifizierten Ma nahmen aus übergreifender Sicht erfolgt anschlie end in Kap. 7.

Tabelle 6.1: Überblick der gewählten Games für die Analyse. Editor (a) bezieht sich auf Editoren des *Play, Create und Share Genres*. Editor (b) bezieht sich auf *Integrierte In-Game Editoren*.

Titel	*Editor*	*Jahr*	*Entwickler*	*Platform*
Little Big Planet 2	a	2011	Media Molecule	PS3
ModNation Racers	a	2010	United Front Games	PS3
inFAMOUS 2	a	2011	Sucker Punch Productions	PS3
Create	b	2010	EA Bright Falls	PS3

6.1 Little Big Planet 2

Little Big Planet 2 (LBP 2) ist ein von *Media Molecule* entwickelter Puzzle-Platformer und wurde im Jänner 2011 für die *PlayStation 3* veröffentlicht. *LBP 2* ist der Nachfolger von Little Big Planet (LBP), dem ersten Titel der aktiv unter dem Marketingslogan *Play, Create and Share* publiziert wurde. *LBP 2* besteht im Wesentlichen aus einem Story-Modus dem (Play) und einer Platform zum Tauschen und Spielen von Levels (Sharing) anderer User. Den Create-Part bildet ein durch die Vielzahl an Tools mächtiger Editor zur Erstellung neuer Levels und Spielformen für das Game. Der Editor ist nach der Einteilung dieser Arbeit einer des *Play, Create und Share Genres* bzw. *Optionaler In-Game Editor*. Das Verhalten als Mehrspieler insbesondere im Editor war auf Grund der speziellen Multiplayer-Eigenschaften von Sandbox Games beschrieben in [44] nicht Teil der Analyse.

Der typische Spielablauf im Story-Modus von *LBP 2* wird durch einen Avatar (s. unten Abschn. 6.1.1) bestritten und gemä dem Platformer-Genre in einer Seitenansicht dargestellt. Der Spieler kann sich zusätzlich in der Tiefenachse auf vier Ebenen bewegen. *LBP 2* beschreibt sich selbst als kleine Welt der gebündelten Vorstellungskraft. Die Erstellung von Inhalten und Dekoration ist zu jeder Zeit Teil der Spielerfahrung. Der ästhetische Ansatz ähnelt dem einer Bastelstube. Es wurde versucht den Spieler das Gefühl zu vermitteln Teil der kreativen Welt zu sein. Dies zeigt sich u. a. durch die durchgängige Verwendung von Analogien im gesamten Game. Ein einfaches Beispiel hierzu ist das Ersetzen von Begriffen im Editor z. B. Gruppieren durch Kleben.

6.1.1 Creation

Der Fokus in diesem Abschnitt liegt auf dem In-Game Level-Editor, der folgend nur als Editor bezeichnet wird. Wie bereits beschrieben sind einige Tools und Funktionen aus dem Editor auch in anderen Bereichen des Games verfügbar. Vielmehr wird versucht durch die ständige jedoch beschränkte Möglichkeit der Dekoration und Anpassung den Spieler auf den Editor vorzubereiten und dem Spieler verschiedene Stufen der Content Creation zur Verfügung zu stellen. Es macht daher zumindest Sinn den Ein uss der anderen Spielteile im weiteren Text kurz zu umranden.

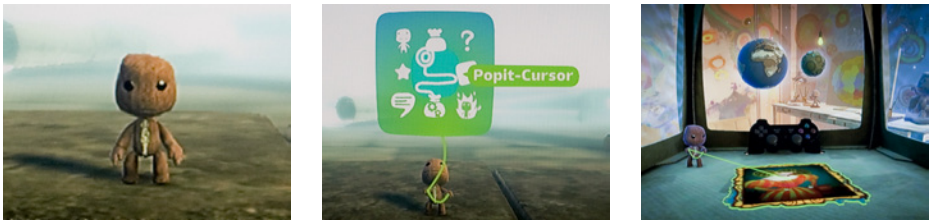
Play und Create sind sehr stark miteinander verknüpft. Erste Missionen des Story-Modus müssen absolviert werden um Zugriff auf den Editor zu erhalten. Der Spieler lernt dadurch den grundlegenden Ablauf, die Ziele und Steuerung des Games. Die Anleitung erfolgt durch den Charakter Da Vinci (siehe Abb. 6.1). Dieser ist als Teil der Game World (Diegetic) umgesetzt. Zusätzlich erhält der Spieler visuelle Hilfe in Form von Sprechblasen (Non-diegetic). Der Spieler wird im Verlauf der ersten Levels mit seinem Avatar (siehe Abb. 6.2) dem Sackboy oder Sackgirl und dessen Steuerung vertraut

Abbildung 6.1: Hilfselemente die den Spieler anfänglich auf den Editor vorbereiten. Die verwendete Ebene der Repräsentation ist unterschiedlich. v.l.n.r.: Da Vinci, Trainings und Tooltips.

gemacht. Dieser kann u. a. Laufen, Springen, Gegenstände verschieben oder sich an Gegenständen festhalten und zwischen den vier Tiefenebenen wechseln. In allen Bereichen des Games wird versucht den Spieler durch den Avatar in das Spielgeschehen reinzuziehen. Über das Popit Menü (siehe Abb. 6.2) können erste Tools wie Sticker zur Dekoration der Game World oder die Anpassung des Charakters bedient werden. Im Laufe des Spiels gesammelte Sticker können u. a. das Erscheinungsbild des eigenen Levels prägen. Gewisserma en durch den Level ein Rückschluss auf den Fortschritt eines Spielers geschlossen werden.

Das zuvor erwähnte Popit Menü ist über eine Art Lasso mit dem Avatar verbunden. Offensichtlich wurde versucht eine Verbindung zwischen dem Avatar (Diegetic) und dem für umfangreichere Screen-Layouts notwendigen 2D-Menü (Non-diegetic) herzustellen. Das 2D-Menü wäre dadurch per Definition eine Meta Representation. Die Entwickler versuchten durchgängig jedwedes Menü in der Game World abzubilden. Der Avatar übernimmt eine tragende Rolle jedwedes Interface-Element in die Game World zu integrieren. Er scheint als zentrales Element der Schnittstelle zwischen Spieler und Game bzw. Spieler und Game World zu fungieren. Die Konzeption des Avatars als zentrales Toy scheint offensichtlich. Noch im Hauptmenü, einer dekorierbaren Kartonbox, wird dem Spieler dessen Ein uss auf die Game World offenbart. Zum Beispiel kann durch rechts und links laufen die Kartonbox in Schwingung versetzt werden. Der Spieler wird quasi noch vor dem Betreten des Editors dazu ermutigt zu experimentieren. Das eigentliche Aufrufen der Menüs aus dem Hauptmenü erfolgt durch Aktivierung des in der Kartonbox platzierten Controllers (siehe Abb. 6.2). Der Avatar stellt sich vor diesen und mimt die Aktionen des Spielers auf dessen Controller. Es kann davon ausgegangen werden dass der Avatar somit auch jedes 2D-Element im Interface wahrnimmt.

Eigene lokale Levels werden durch einen Mond mit Einschlagskratern bzw. Zugängen zur Kreativwelt dargestellt (siehe Abb. 6.3). Die Krater am Mond bilden die Slots für die Levels. Der Mond selbst ist von seiner Mate-



(a) (b) (c)

Abbildung 6.2: Der Avatar (a) scheint als zentrales Element der Schnittstelle zwischen Spieler und Game bzw. Editor zu fungieren. Das Popit Menü (b) dient zur Auswahl der verschiedenen Tools. Über ein Lasso wird versucht das au erhalb der Game World existierende 2D-Menü durch die Verbindung mit dem Avatar in die Game World zu holen. Bereits im kartonboxartigen Hauptmenü (c) werden die Möglichkeiten zur Dekoration als auch die Bindung zum Avatar verdeutlicht.

rialität her dem Game angepasst und besteht aus vernähtem Stoff. Er kann durch Vergabe eines Materials und durch das Setzen von Stickern dekoriert werden. Der Avatar überblickt und steuert die Auswahl durch den Controller im Vordergrund. Klassiche 2D-Menüs sind sichtbar durch den Avatar in der Game World, gleichzeitig aber nicht Teil dieser. Jeder Menüpunkt könnte insofern wieder eine Meta Representation sein. Wird ein Level gewählt wird der Spieler in den runden Krater gesaugt. Eine leere Game World wird angezeigt und der Avatar betritt diese durch das Startportal. Es liegt nun am Spieler den leeren Raum der Game World bzw. der Kreativwelt zu füllen. Die Steuerung des Avatars erfolgt zunächst analog dem Play-Part. Der Spieler ist dadurch sofort in der Lage im Editor zu navigieren.

Die notwendigen Tools zum Erstellen von Inhalten im Editor müssen durch Absolvierung eines Basis-Trainings freigeschalten werden. Ein Monitor aus Karton (siehe Abb. 6.1) zeigt dem Spieler ein Trainingsvideo. Der Monitor ist direkt in die Game World integriert und kann als solcher über den Avatar durch Sprung auf den Button aktiviert werden. Unter humoristischer Anleitung einer Stimme aus dem Off können die Schritte des Trainings direkt in der Game World über den Avatar reproduziert und ausprobiert werden. Auch hier wurde es möglichst vermieden den Spieler durch eine wenig subtile Hilfe aus dem Spiel zu rei en. Die Stimme aus dem Off scheint au erhalb der sichtbaren Spielwelt zu existieren. Sie hat einen mentorartigen Charakter und spricht stets den Avatar in Du-Form an. Der Spieler fühlt sich über seinen Avatar angesprochen. Es wird klar, dass Aktionen über diesen auszuführen sind. Die Tools sind nach Beendigung des letzten Basis-Trainings verfügbar.

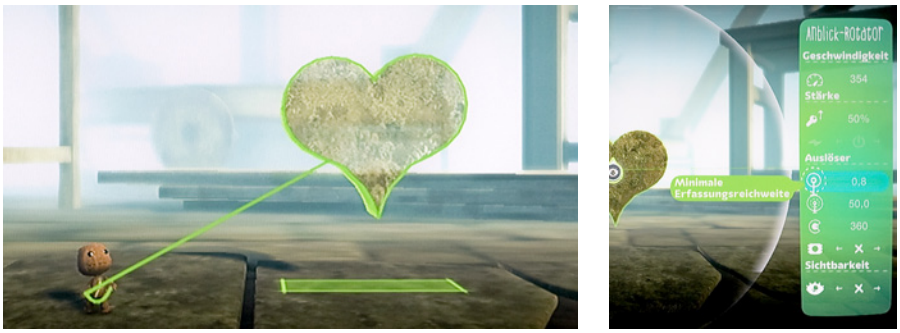
Der Spieler kann mit dem gewonnenen Wissen das Popit Menü seines Avatars benutzen um die verschiedenen Tools und Funktionen (Goodies Bag,

Abbildung 6.3: Lokale Speicherslots für Levels werden in Form eines dreidimensionalen Mondes und Kratern (Levels) dargestellt. Im Vordergrund ist der Avatar zu sehen der den Controller lenkt. Die Menüs und Tooltips sind daher innerhalb der Game World sichtbar, auch wenn sie nicht innerhalb dieser existieren. Sie können als Meta Representations eingeordnet werden.

Tools Bag, Stickers und Popit Cursor) zu benutzen. Zum Beispiel ist der Popit Cursor eine Art Ersatz für einen Mauscursor der u. a. zur Selektion von erstellten Objekten, Verschiebung und Gruppierung verwendet werden kann. Jedes Tool an sich dient in seiner Form selbst als Mauscursor.

Die Navigation innerhalb der Game World des Editors erfolgt wie beschrieben analog dem Play-Part durch den Avatar. Die Steuerung der verschiedenen Tools erfolgt ähnlich einem Mauscursor der über den Joystick des Controllers benutzt wird. Jedwedes im Popit Menü gewählte Tool wird durch eine Art Lasso mit dem Avatar verbunden. Der Avatar hält das Lasso in der Hand (siehe Abb. 6.4). Steuerungseingaben am Controller wirken sich auf die Bewegung der Hände des Avatars als auch auf die Bewegung des Tools aus. Es sieht hierbei so aus als würde der Avatar das Tool zügeln oder steuern. Aktionen und Einstellungen an Objekten werden direkt in der Game World über den Avatar bzw. das ausgewählte Tool vorgenommen. Die Darstellung von Einstellungen die insbesondere ein komplexeres Layout erfordern, werden wie das Popit Menü über ein 2D-Fenster repräsentiert und ebenfalls über ein Lasso mit dem Avatar verbunden (siehe Abb. 6.4). Analog dem Popit Menü kann das Menü durchaus als Meta Representation angesehen werden. Eine mögliche Lösung sich so wenig wie möglich der Game World zu entfernen.

Die Verknüpfung von Elementen wie z. B. einem Druckknopf und einem



(a) (b)

Abbildung 6.4: Der Avatar (a) platziert bzw. malt mit einem Lasso ein Objekt mit gewähltem Material und Pinselform. Die Eingaben des Spielers beim Controller spiegeln sich in der Lassobewegung am Avatar wider. Es scheint als würde der Avatar das Tool zügeln und lenken. Das Menü für erweiterte Einstellungen ist analog dem Popit Menü (b) durch das Lasso mit der Game World und dem Avatar verbunden und damit per De nition als Meta Representation einzuordnen. 2D-Menüs nden in *LBP 2* meist Anwendung wenn die Fülle an notwendigen Informationen nicht als direkter Teil der Game World darstellbar ist.

anderen Objekt z. B. Feuer wird durch ein Knotensystem mit Ein- und Ausgängen umgesetzt. Die Ein- und Ausgänge werden direkt am Objekt in der Game World dargestellt (siehe Abb. 6.5). Die Verbindung erfolgt durch ein Kabel, welches entsprechendes elektrisches Feedback bei dessen Platzierung von sich gibt. Der Zustand des Signals wird direkt über das Kabel gezeigt. Komplexere logische Verknüpfungen können durch Schalttafeln und logische Gatter realisiert werden (siehe Abb. 6.5). Diese Schalttafeln werden vom Game passend als Mikrochips bezeichnet. Offensichtlich bedienen sicht die Entwickler hier wieder der realen Bezeichnungen. Interessant ist die Abstraktion der Umsetzung und der Vergleich der visuellen Umsetzung und Usability mit modernen Modding-Editoren wie z. B. dem Flow Graph Editor (siehe Abb. 3.6).

Ein gro er Vorteil durch die Verwendung eines Avatars ergibt sich beim Testen der Kreationen und Verknüpfungen. Diese können direkt im Editor getestet bzw. gespielt werden. Der Editor stellt verschiedene Modi zur Verfügung. Im Interface sind diese durch das Popit Menü, ein Play oder Pause Icon im HUD oder durch das physikalische Verhalten der Objekte in der Game World dargestellt. Die Aktivierung und Auswahl der Modi ersetzt am Controller die Emotionssteuerung des Avatars. Diesem kann durch das Steuerkreuz im Play-Part ein Gesichtsausdruck verliehen werden. Eine Funktion die wohl auf Grund der notwendigen Undo- und Redo-Funktionalität für den Editor entfernt wurde.

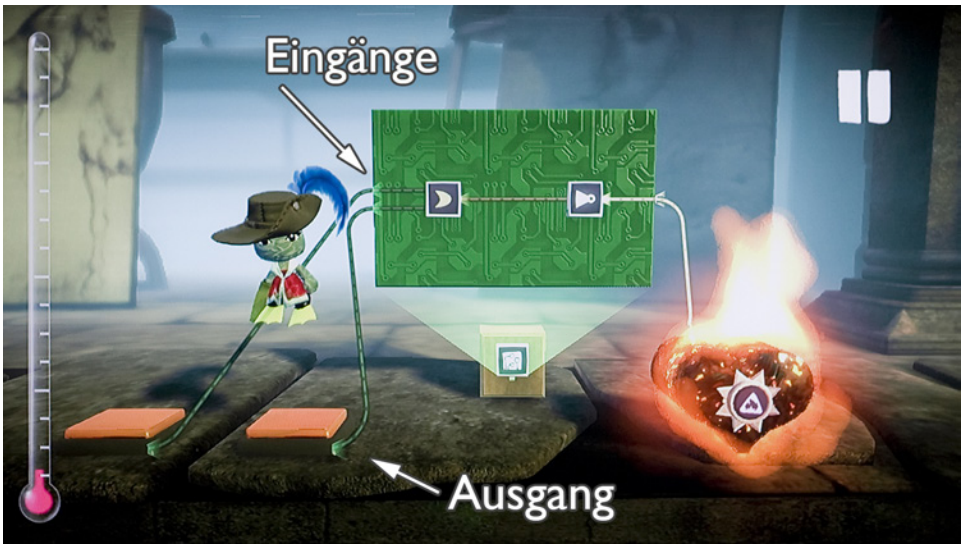


Abbildung 6.5: Verbindungen und Visualisierung der Ein- und Ausgänge sind Teil der Game World. Signalzustände sind durch Einfärbung der Kabel sichtbar. Komplexere Schaltungen werden als Mikrochips realisiert.

Kreativ-Modus: Der Avatar folgt keinen physikalischen Regeln und kann frei im Raum umherschweben (siehe Abb. 6.6). Das Schweben wird durch ein blubberndes Geräusch untermalt. Der Avatar befindet sich in einer Ebene vor der eigentlichen Spielebene kann daher mit keinen Objekten innerhalb der Game World direkt interagieren. Die Interaktion erfolgt ausschlie lich über das gewählte Tool. Der Höhenunterschied macht sich durch rauschenden Wind bemerkbar. Der Kreativ-Modus lässt sich durch das Verhalten und Interaktionsmöglichkeiten des Avatars erkennen jedoch nicht am Verhalten der physikalischen Objekte in der Game World. Neben der freien Bewegung kann im Kreativ-Modus der sichtbare Bereich und Blickwinkel angepasst werden. Der Avatar kann sich schneller bewegen. Der Kreativ-Modus erhöht die E ciency. Dies macht sich auch insofern bemerkbar, dass der Avatar nun dem Tool-Cursor folgt. Das Tool steuert das Toy. Der Spieler soll sich ausschlie lich auf die Umsetzung seiner Vorstellung konzentrieren. Inbesondere mit ansteigender Grö e des Levels bringt der Modus Vorteile.

Popit Menü bzw. Pause-Modus: Ist das Popit Menü geöffnet oder das Spiel im Pause-Modus sind Teile des physikalischen Verhaltens ausgesetzt. Ein sich bewegendes Objekt führt keine Bewegung aus. Objekte kollidieren jedoch beim Verschieben mit dem Popit Cursor. Der Avatar kann bei geöffnetem Popit Menü nicht bewegt werden, au er der Kreativ-Modus ist aktiv. Durch den Avatar können z. B. Sprungabstände zwischen Objekten getestet werden (siehe Abb. 6.6). Er dient

in diesem Modus als Navigation und Tool zum Testen. Der Avatar kann im Pause Modus z. B. durch Berührung eines brennenden Objektes verbrennen (siehe Abb. 6.6), poppt jedoch im Kreativ-Modus wieder an selbiger Stelle auf. Ein Sterben und Verlust des Fortschrittes ist nicht möglich.

Play-Modus: Im Play-Modus entspricht das Verhalten des Avatars als auch aller Objekte in der Game World dem finalen Zustand des Play-Parts. Bewegungen werden ausgeführt, Buttons können durch Springen bedient werden und logische Verknüpfungen werden durch farbige Markierung der Signalströme bei deren Kabeln visualisiert (siehe Abb. 6.6). Der Play-Modus wird im HUD durch ein Play-Zeichen symbolisiert. Er entspricht nicht in voller Form dem Play-Part. Dieser muss über das Hauptmenü aktiviert werden und äu ert sich durch die nun verfügbare Punkteanzeige als auch dem Verschwinden von z. B. Kabeln, logischen Verknüpfungen und Schalttafeln.

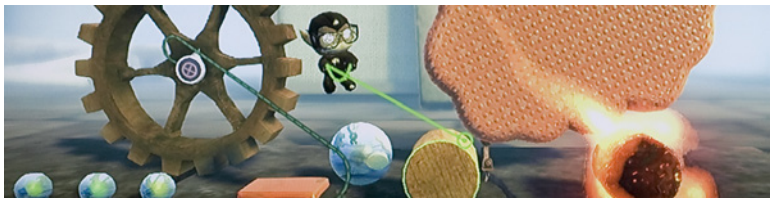
Bleibt der Avatar beim Erstellen zwischen Objekten stecken kann der Spieler analog dem Play-Part den Avatar über das Popit Menü zum Platzen bringen. Im Play-Part kostet diese Aktion ein Leben. Im Editor wird der Avatar im Kreativ-Modus wiederbelebt. Die verschiedenen Modi dienen auch als Undo- und Redo-Funktionalität. Jeweils Play- und Pause-Modus und deren Zustand der Game World werden gespeichert. Der Spieler kann dadurch z. B. im Play-Modus die Auswirkungen der Schwerkraft auf die Objekte betrachten muss jedoch den letzten Zustand mühsam mit dem Avatar wiederherstellen. Ein Knopfdruck am Controller genügt zum Wechsel des Zustandes. Der Wechsel wird durch eine optische Verzerrung analog einer VHS-Kassette beim Spulen signalisiert (siehe Abb. 6.7). Das Spulen wird zusätzlich im HUD angezeigt.

Das HUD kann deaktiviert werden. Es zeigt neben Tooltips und dem aktuellen Spielmodus auch den Zustand der Befüllung des Levels in Form eines Thermometers. Das HUD ist nicht Teil der Game World und wird nicht durch den Avatar wahrgenommen. Es zeigt ausschlie lich Information die nur für den Spieler bestimmt sind.

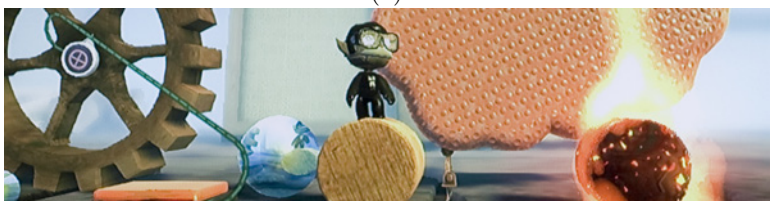
6.1.2 Sharing

Die Kreationen des Spielers werden auf dessen Planeten repräsentiert (siehe Abb. 6.8). Der Mond dient als lokaler Speicherslot, der Planet als Level-Portfolio. Ähnlich dem Mond kann der Planet durch den Spieler mit Hilfe des Popit Menüs und der Anwendung von Materialien und Sticker umgestaltet werden. Der Zugriff auf Levels anderer Spieler erfolgt ebenfalls über das Menü. Eine Kategorisierung erfolgt hier durch den Entwickler, Ratings und Zugriffe.

Neben dem Sharing eines Levels besteht auch die Möglichkeit einzelne im Editor erstellte Objekte oder Apparaturen zu veröffentlichen. Objekte können im Level als Preisblasen platziert werden. Ein Spieler der den Level



(a)



(b)

(c)

(d)

Abbildung 6.6: Verschiedene Modi des Editors von *LBP 2*. Je nach Modus unterscheidet sich das Verhalten des Avatars und der Game World. Im Kreativ-Modus (a) schwebt der Avatar frei auf einer Ebene über den Spielobjekten. Eine Interaktion mit diesen kann nur über ein Tool von statten gehen. Im Pause-Modus (b) kann mit dem Avatar auf Objekten gesprungen werden oder diese können festgehalten werden. Knöpfe, Logik und teilweise Physik zeigen noch keine Wirkung, jedoch kann sich der Avatar z. B. verbrennen (c). Der Play-Modus (d) entspricht dem Play-Part. Objekte werden durch die Physik beein usst und die Logik kann ausprobiert werden. Die Unterscheidung des nalen Play-Parts liegt in der fehlenden Anzeige von Punkten und sonstigen für das Editieren relevanten Informationen.

Abbildung 6.7: Beim Vor- und Zurückspulen entstehen Bildartefakte ähnlich dem Spulen einer VHS-Kassette.

(a) (b)

Abbildung 6.8: Der Avatar wählt die Etiketten für den Level aus (a). Das Sharing in *LBP 2* erfolgt durch Veröffentlichung erstellter Levels auf dem eigenen Planeten. Über den Avatar wird nach Planeten bzw. Levels anderer Spieler gesucht (b).

bestreitet und mit seinem Avatar über diese Preisblase läuft erhält dieses Objekt. Je nach Einstellung kann es wiederum weiterverbreitet werden oder bleibt nur dem einsammelnden Spieler vorbehalten.

6.1.3 Zusammenfassung

Little Big Planet 2 lebt von der allgegenwärtigen Möglichkeit zur Dekoration, Kreation und Individualisierung. Die Game World von Menü und Editoren

kommuniziert den Status des Spielers. *LBP 2* präsentiert sein Gesamtkonzept aus ästhetischer Sicht sehr stark über Analogien der Kinderzeit. Die Tools wie Klebstoff und Schere oder Materialien wie Karton und Stoff mit Nähten sind an eine Bastelstube angelehnt. Das Game beschränkt sich durch die Vielzahl an Tools jedoch nicht auf eine Zielgruppe. Neben den Tools die gleichzeitig als Toy betrachtet werden können, ist die auffälligste Ma nahme zur Steigerung der Satisfaction der Sackboy der Avatar. Dieser könnte als das Primärtoy angesehen werden. Der Spieler sieht sich über den Avatar selbst in der Game World. Der Avatar bringt das gesamte virtuelle Interface in die Game World. Er dient dem Play-Part, ist im Editor aber auch ein Tool zur Navigation, Kreation und zum Testen. Die Grenze zwischen Spielen und Erstellen wird regelrecht aufgehoben. Der Avatar ist der E ciency und Effectiveness des Editors zuträglich indem die Bedienung des Editors mit den bereits aus dem Play-Part gelernten Funktionen erfolgt. Mit dem optionalen HUD und dem Kreativ-Modus wurde zusätzlich versucht diese beiden Eigenschaften zu maximieren.

6.2 ModNation Racers

ModNation Racers ist ein Rennspiel von *United Front Games* und wurde für die *PlayStation 3* und *PlayStation Portable* 2010 veröffentlicht. Der Spielablauf und Feeling der Fahrzeuge erinnert an Vetreter des Genres wie z. B. *Mario Kart* von Nintendo. Es gilt mit dem Fahrzeug verschiedene Rennen zu absolvieren und in der Karriere aufzusteigen. Die Strecken sind bestückt mit Powerups, Fallen und Waffen. Diese können benutzt werden um sich gegenüber den Mitstreitern einen Vorteil zu verschaffen.

Wie *Little Big Planet* wurde *ModNation Racers* durch das *Play, Create and Share* beworben. Die Einteilung der Editoren erfolgt daher in selbiges Genre. Teile des UGC sind Mods (Charaktere), Cars (Fahrzeuge) und Tracks (Strecken). Der visuelle Stil von Mods und Cars ähnelt denen von Sammelspielzeugen. Vermutlich in Anspielung an das Sharing. Ebenso dürfte das Vorkommen von Mod im Namen und als Bezeichnung der Charaktere mit dem Modding als Hintergedanken entstanden sein.

Beim Zugriff auf Editoren und UGC stellt eine Besonderheit das Menü dar. Dieses wird als Teil der Game World dargestellt (siehe Abb. 6.9). Der Spieler navigiert mit seinem Fahrzeug und fühlt sich als Teil dieser. Als Tutorial dient das verp ichtende Startrennen. Vermutlich wegen der Effectiveness existiert ein alternativer und wesentlich schnellerer Weg der Navigation mittels Kreismenü (siehe Abb. 6.9).

6.2.1 Creation

In *ModNation Racers* existieren drei Editoren für Mods, Cars und Tracks. Beim Zugriff auf diese über die Game World besteht zunächst noch der Ein-

(a) (b)

Abbildung 6.9: Der ModSpot (a) bzw. das Hauptmenü. Der Spieler ist Teil der Game World des Menüs. Zur schnelleren Navigation dient alternativ ein Kreismenü (b).

Abbildung 6.10: Zugriff auf die Editoren im ModSpot bzw. in der Creation Station. v.l.n.r.: Gebäudefront, Car-Editor, Mod-Editor und Track-Editor.

druck Teil des Games zu sein (siehe Abb. 6.10). Mit Ausnahme des Track-Editors erfolgt das Editieren im Game Space des Menüs. Die Optionen werden in 2D au erhalb der Game World dargestellt (siehe Abb. 6.11). Die Illusion Teil dieser zu sein wird gestört.

Editoren von Mods und Cars sind primär als parameter-based anzusehen. Fortgeschrittene Spieler können die Accessoires jedoch auch dreidimensional anordnen. Selbiges Verhalten gilt beim Platzieren von Stickern. Der Cursor wird dabei als Diegetic Representation im Game Space und zu einem gewissen Grad auch in der Game World angezeigt. Zu einem gewissen Grad weil der Mod (Charakter) nicht direkt auf den Cursor des Spielers reagiert. Es wurden jedoch diverse Animationen eingesetzt um diesen Eindruck zu vermitteln. Zum Beispiel hebt der Mod die Hände wenn dessen T-Shirt dekoriert wird. Der Mod ist im Mod-Editor dennoch mehr ein Toy bzw. eine Sammelfigur als eine Repräsentation des Spielers. Es werden nicht die Outfits sondern vielmehr die Mods gespeichert. Der Car-Editor entspricht in Bedienung dem Mod-Editor. Auffälliger ist bei diesem der Wechsel von Parametern durch die Verwendung von poppenden Animationen als Feedback von Aktionen.

Abbildung 6.11: Nach Auswahl des Mod-Editors zoomt die Kamera hinein. Optionen werden im HUD dargestellt. Es ensteht der Eindruck der Bekleidung und weniger der Ankleidung. Die Bindung zwischen Spieler und Mod (Avatar) ist unterbrochen.

Solche Animationen sind bei den Mods nicht zu finden.

Beim Zugriff auf den Track-Editor wird das 3D-Menü verlassen. Der Spieler findet sich mit seinem Fahrzeug in einer unbearbeiteten Landschaft wieder. Durch Umherfahren kann ein Startpunkt für die Strecke gewählt werden. Dem Spieler wird durch die anfänglich bekannte Steuerung und durch seine Repräsentation als Avatar das Gefühl für die Game World und die Bindung zu seiner Kreation bewusst gemacht. Es wurde versucht den Spieler zunächst nicht mit Neuem zu konfrontieren und ein schnelles Erfolgserlebnis zu garantieren. Ein Video-Tutorial ist optional, jedoch erst durch eigenständiges Suchen au ndbar. Die Auswahl der Tools und Anzeige der Tooltips erfolgt im HUD (siehe Abb. 6.14).

Die Geleitung des Spielers bei der Erstellung erfolgt durch ein Wizard-System[1]. Dies trägt u. a. dazu bei zielgerichtet eine valide Strecke zu generieren. Durch eine Asphaltiermaschine (siehe Abb. 6.12), verwendet als Analogie, wird zunächst die Strecke gebaut. Dem Spieler wird der Zweck des aktuellen Tools bewusst. Die Steuerung erfolgt wie die eines Fahrzeuges. Zusätzlich kann während der Fahrt die Neigung für eine hohe oder tiefe Streckenführung gewählt werden. Die Audioausgabe beim Asphaltieren ergänzt den Einsatz der Maschine. Eine Linie im Game Space beschreibt den optimalen Weg zum Startpunkt bzw. den Weg der Autovervollständigung. Beim

[1]Ein *Wizard* leitet den User durch Fragestellungen Schritt für Schritt durch eine spezifische Aufgabe [5, S. 403]

Abbildung 6.12: Der erste Schritt im Editor zur Generation einer validen Strecke wird durch die bekannte Fahrzeugsteuerung ausgeführt. Als Analogie zur Realwelt wird dafür eine Asphaltiermaschine verwendet. Der Spieler asphaltiert den Streckenverlauf. Eine e ziente und effektive Lösung eine valide Strecke zu generieren. Die Repräsentation präsentiert die Funktion des Tools.

Betätigen der Autovervollständigung wird die Stra e im Schnelldurchlauf automatisch asphaltiert. Gleichzeitig ertönt ein Spulgeräusch ähnlich dem Spulen eines Audiobandes.

Im weiteren Verlauf bietet der Editor die Option an sämtliche Umgebungsobjekte wie Häuser oder Bäume automatisch zu populieren. Die Strecke wäre im Anschluss einsatzbereit und optisch ansprechend. Als Feedback der Autopopulation iegt die Kamera die Strecke ab. Gleichzeitig sieht der Spieler an den Seiten Bäume und Objekte mit Rauch und Schmutz durch Animationen aufpoppen (siehe Abb. 6.13). Ähnliches Feedback entsteht beim manuellen Platzieren von Objekten (siehe Abb. 6.13). Insbesondere wird passendes auditives Feedback beim Modifizieren der Landschaft gegeben. Das Feedback kommuniziert eine Bindung und Ein uss des Spielers auf die Game World, ähnlich der greifbaren Bindung mit der Natur beim Setzen einer P anze.

Der Spieler verbringt die weitere Zeit in einer Art Gott-Modus (siehe Abb. 6.14). Er kann sich frei bewegen und befindet sich über der Strecke. Der Spieler ist nicht mehr Teil der Game World sondern blickt auf diese hinab. Die Ansicht erhöht die Übersicht beim Editieren. Das Verhältnis der Dimension zwischen Car und Track wäre zu gro und eine übersichtliche Navigation nicht mehr gewährleistet. Die Möglichkeit mit dem Fahrzeug die Strecke zu

(a) (b)

Abbildung 6.13: Der Spieler iegt bei Autovervollständigung die Strecke ab (a). Neue Objekte poppen durch eine Animation auf (a) & (b) und durch Effekte wie Rauch- und Schmutzpartikel wird die Verbindung mit der Landschaft plastisch dargestellt.

Abbildung 6.14: Die Anpassung selbst geschieht in einer Art Gott-Modus mit Sicht auf die Strecke und Landschaft. Der Spieler fühlt sich nicht mehr als Teil der Game World, es wird jedoch mehr Übersicht geschaffen. Die Auswahl der Tools und Objekte sowie Darstellung von Tooltips erfolgt ausschlie lich über das HUD. Die Balken im oberen Bereich visualisieren das Ressourcenlimit.

verlassen und plötzlich auf Bergen zu fahren wäre einerseits schwer abzubilden und andererseits ein Bruch im vom Spieler gewöhnten Paradigma einer geschlossenen Strecke.

Die Anzeige von Informationen erfolgt über das HUD (siehe Abb. 6.14) und dem Cursor der je nach gewähltem Tool entsprechend dargestellt wird (siehe Abb. 6.15). Dieser ist im Game Space existent daher als Spatial Representation der Zugang zur Game World. Die Anzahl verfügbarer Objekte zur

Abbildung 6.15: Der Spieler führt Aktionen in der Game World durch einen Cursor aus. Dieser ist je nach Aktion anders dargestellt. v.l.n.r.: Autovervollständigung im Streckenbereich, Lichteinstellungen und logische Verbindung.

Dekoration der Umgebung erhöht sich mit dem Spielen des Karriere-Modus. Die erstellten Levels können daher auch den aktuellen Status des Spieler widerspiegeln. Die Verwendung der Objekte als auch die Manipulation der Landschaft wird durch ein Ressourcenmaximum begrenzt (siehe Abb. 6.14). Diese Form der Eingrenzung dürfte primär durch technische Gründe wie Performance oder Dateigrö e motiviert sein. Gewisserma en ensteht dadurch eine Challenge an den Spieler seine Vision innerhalb dieses Limits durch Verständnis und optimalen Einsatz der Tools umzusetzen.

Die Strecke kann nach Asphaltierung jederzeit per Preview- oder Race-Modus mit dem Car befahren werden. Der Spieler wird dabei wieder Teil der Game World. Im Preview-Modus ohne Waffen und Items befahren neben dem Spieler auch KI-Spieler die Strecke. Dadurch lässt sich das Verhalten der KI überprüfen. Es dient aber auch der Unterhaltung da nicht das Gefühl vorherrscht die Strecke für sich selbst zu erstellen.

6.2.2 Sharing

Der Zugriff auf das Sharing in *ModNation Racers* erfolgt im ModSpot (siehe Abb. 6.16). Durch die Sharing-Station kann der Spieler seine Kreationen veröffentlichen oder andere downloaden. Top Mods und Top Cars werden in Form von rie igen Statuen im ModSpot als Diegetic Representation dargestellt. Die besten Kreationen sind damit Teil der Game World eines jeden Spielers. Die Statuen drücken die Ehrung durch die Entwickler aus. Top Tracks werden ähnlich prominent auf einer Video Wall präsentiert. Die Repräsentationen dienen als eine Art Shortcut zum jeweiligen Sharing-Menü.

Die Auswahl der Ranglisten, Kategorisierung, Optionen und Rating wurden aus Gründen der Übersicht und der Masse an abzubildenen Informationen als 2D-Menü umgesetzt (siehe Abb. 6.17). Die Kreationen werden als gerenderters Standbild dargestellt. Die Ästhetik unterstreicht den Sammelcharakter der Figuren und Fahrzeuge. Die Aufteilung im Menü ähnelt dem eines Wandregales für Sammelfiguren.

Abbildung 6.16: Verschiedene Zugriffsmöglichkeiten auf die Sharing-Funktionalität im ModSpot. Diese dienen als eine Art von Shortcut aber auch als Ehrung der Top-Ersteller. v.l.n.r.: Sharing im Creation Center, Top Mods, Top Cars und Top Tracks.

Abbildung 6.17: Die Darstellung der Sharing-Funktionalitäten erfolgt als 2D-Menü. Die Aufbereitung des Menüs ähnelt dem eines Wandregals zur Aufbewahrung der Sammel guren.

6.2.3 Zusammenfassung

Im Fokus des UGC in *ModNation Racers* stehen die Erstellung von Rennstrecken und Individualisierung von Mods (Charaktere) und Cars (Fahrzeuge). Der Sammelgedanke des Sharings spiegelt sich in der ästhetischen Umsetzung von Cars und Mods wieder. Durch Mod und Car als Avatar im Hauptmenü wurde versucht den Spieler in die Game World zu integrieren. Insbesondere der Mod-Editor bricht das Konzept des Avatars. Der Mod wirkt auf Grund des Editor-Interfaces mehr als Sammelfigur. Der Spieler fühlt sich im Editor nicht durch diesen repräsentiert. Der Track-Editor greift das Konzept des Avatars nur zu Beginn mit der Analogie einer Asphaltiermaschine

auf. Der Spieler fühlt sich innerhalb der Game World existent, erhebt sich aber im weiteren Verlauf über diese durch eine Art Gott-Modus. Die Aufgabe der Verbindung zwischen Spieler und Game World übernimmt u. a. die editierbare Landschaft. Durch audiovisuelles Feedback wie Schmutzpartikel oder Erdgeräusche wird der Ein uss auf diese nahezu greifbar. Insgesamt ist der Track-Editor jedoch auf E ciency getrimmt. Dies gilt ebenfalls für den Zugriff auf die Kreationen anderer Spieler, die in einer Art Sammelregal als 2D-Menü dargestellt werden. Herausragende Kreationen sind Teil eines jeden 3D-Hauptmenüs durch die Darstellung als Statue. Diese dienen der Belohnung als auch eine Art Shortcut zum Sharing-Menü.

6.3 inFAMOUS 2

Durch *Sucker Punch Productions* entwickelt wurde *inFAMOUS 2* 2011 für *PlayStation 3* veröffentlicht. Es kann auf Grund seiner frei begehbaren Welt dem Genre der Open-World-Games bzw. Sandbox Games zugeordnet werden. Die Repräsentation des Spielers erfolgt durch Darstellung des Charakters in der Third-Person-Ansicht. Wann Missionen gespielt werden kann durch den Spieler nach anfänglicher Einführung in das Spiel selbst gewählt werden. Mit der Beendigung dieser erhält der Spieler auch Zugriff auf benutzergenerierte Missionen und den Missions-Editor. Der getroffen Einteilung nach ist dieser dem *Play, Create und Genre* zuzuordnen. Der *Mission-Editor* wird laufend erweitert, daher sei an dieser Stelle das Abschlussdatum der Analyse mit 14.11.2011 notiert.

6.3.1 Creation

Der Editor lässt sich nach abgeschlossener Einführung in das Game über das Menü öffnen. Die aktuelle Position im Play-Part dient sofern ein neuer Level erstellt wird als Startposition im Editor. Es besteht auch die Möglichkeit mit einem Missions-Template (Zielübung, Rettungsmission, Parkour, Rennen, Bewachen, usw.) zu beginnen. Diese können als Lernunterlage oder zur Erstellung schneller Ergebnisse genutzt werden. Sie sind vergleichbar mit der Nutzung von vorgebauten Gebäuden und Einheiten zum schnellen Missionsstart in RTS-Games. Das Abändern bestehender Missionen wird in *inFAMOUS 2* als Remixing bezeichnet.

Der Editor spielt in einer Art Parallelwelt zur eigentlichen Game World des Play-Parts. Vielleicht existiert diese Welt auf der Gedankenebene der Hauptfigur. Dem Spieler stehen die freigeschaltenen Areale und Objekte des aktuellen Spielstatus zur Verfügung, Missionen sind daher auch ein Abbild des aktuellen Spielfortschritts des Spielers. Im Hintergrund zirpen Grillen, Musik untermalt die Stimmung, Ampeln wechseln zwischen rot und grün, jedoch existiert kein organisches Lebewesen in der Game World des Editors.

Abbildung 6.18: Die Repräsentation des Spielers im Editor erfolgt als Cursor, im Play-Part als Charakter. v.l.n.r.: Cursor am Boden, Cursor in der Luft, Nach Beenden des Editors startet der Charakter an der letzten Position des Cursors. Er fällt zu Boden.

Die grundlegende Steuerung und die Map als HUD-Element wurden aus dem Play-Part übernommen. Der Spieler selbst wird durch einen Cursor im Game Space zumindest als Spatial Representation dargestellt (siehe Abb. 6.18). Der Cursor kann als eine Form eines Platzhalters des Charakters aus dem Play-Part betrachtet werden. Es ist mit diesem nicht möglich durch Hindernisse wie Zäune oder Gebäude hindurchzu iegen. Der Spieler muss explizit die Flüghöhe des Cursors anpassen um diese zu überwinden. Dies dient zugleich zur Vermeidung von Platzierungsfehlern als auch als eine Art von Kreativ-Modus. Beim Erreichen der Grenzen des Areals, welches durch den Spielerfortschritt im Play-Part begrenzt ist, trifft der Cursor bzw. Spieler auf eine unsichtbare Barriere. Gleichzeitig beginnt die Map analog dem Play-Part zu ackern (siehe Abb. 6.19). Es scheint als würde der Spieler an die Grenzen des ihm Bekannten sto en. Der Übergang zwischen dem Play-Part und der Gedankenwelt des Spielers (Editor) gestaltet sich ie end. So bestimmt die aktuelle Cursor-Position die neue Startposition im Play-Part. Der Editor bzw. Cursor kann als eine Art Teleport durch Gedankenkraft benutzt werden. Angesichts der übermenschlichen Fähigkeiten des Hauptcharakters eine passende Analogie.

Im HUD werden zusätzlich durchgängig die verfügbaren Ressourcen, d. h. verwendete Objekte im Verhältnis zur maximal zulässigen Zahl, angezeigt (siehe Abb. 6.19). Tooltips, Einstellungen und das Kreismenü zur Auswahl der gewünschten Operationen befinden sich im Gegensatz zu den Objekten ebenfalls im HUD (siehe Abb. 6.19). Die Objekte wie z. B. Charaktere, Wegpunkte oder Logikboxen werden als 3D-Element im Game Space dargestellt (siehe Abb. 6.20). Charaktere sind bis auf deren Idle-Animation leblos. Logikboxen sind in der Form von Projektoren holografischer Objekte dargestellt. Je nach Zustand von Selektion wird die Darstellung durch das Einfärben der Outlines oder durch zusätzliche Icons verändert oder ergänzt. Für alle Objekte u. a. den Logikboxen werden Schatten berechnet.

Die Verbindung von Logikboxen erfolgt durch Kabel und Steckdosen (sie-

(a) (b)

Abbildung 6.19: Jede Aktion wird zunächst per Kreismenü (a) im HUD gewählt. (a) zeigt weiters auf der rechten Seite die Ressourcenanzeige und die Minimap, sowie links die Tooltips. Die Platzierung von Objekten erfolgt in der Game World über den Cursor. Umfangreiche Informationen wie z. B. Einstellungen eines Objektes werden im HUD (b) angezeigt.

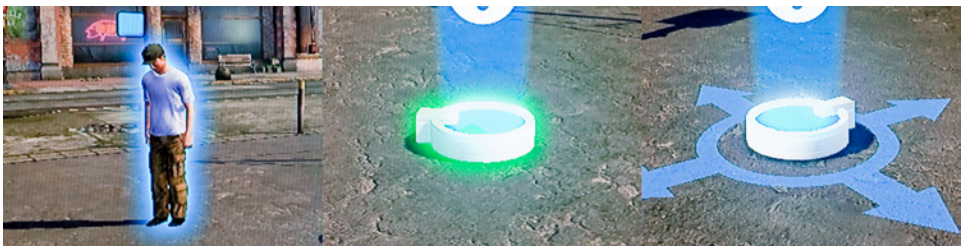


Abbildung 6.20: Objekte werden innerhalb der Game World dargestellt. Je nach Aktion wird die Farbe geändert oder ein Icon angefügt. v.l.n.r.: Darstellung einer Person, Selektierte Logikbox und Logikbox bei Verschiebung.

he Abb. 6.21). Die Kabel werden als Teil des Game Spaces dargestellt. Die Verbindungsrichtung wird im Kabel in Form von Pfeilen präsentiert. Die Verbindung erfolgt durch den Cursor. Der Spieler fühlt sich in der Position eines Elektrikers der Kabel und Steckdosen verbindet und gegebenfalls neue Elemente einfügt (siehe Abb. 6.21). Dies geschieht über den Cursor und Selektion eines Kabelpunktes. Die Analogie ist hierbei passend auf den Charakter zugeschnitten da dessen übernatürliche Fähigkeit die Beherrschung der Elektrizität ist.

Das Feedback auf Aktionen des Spielers erfolgt durch simple auditive Ausgabe. Als Ausnahme führt das Löschen eines Objektes in der Game World zu einer sofortigen massiven Explosion bei nicht humanen Objekten (siehe Abb. 6.22). Bei humanen Objekten kann zunächst ein Fluchen der Person vernommen werden, worauf anschlie end die Explosion erfolgt. Aus Sicht der Analogie einer Gedankenwelt kann die Explosion als Verlust

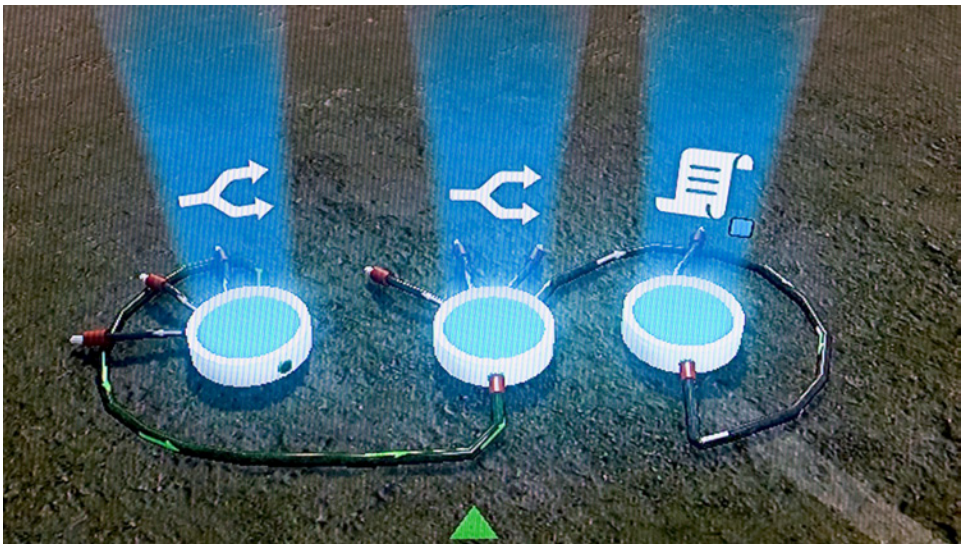


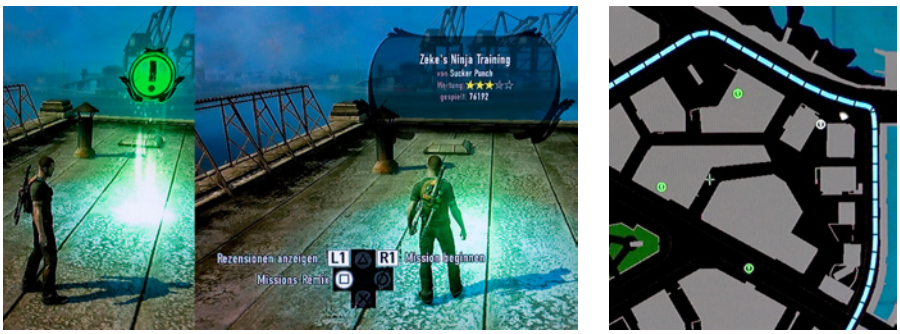
Abbildung 6.21: Die Logikboxen sind als Hologrammprojektoren gestaltet. Sie besitzen teilweise Ein- und Ausgänge. Diese werden durch in der Game World dargestellte Kabel verbunden. Die Pfeile geben die Signalrichtung an.

von Informationen oder als das aggressive Vergessen von Informationen betrachtet werden. Aus Sicht der Usabilty, inbesondere der Satisfaction, erhält der Spieler beim Korrigieren von Fehlern positives Feedback in Form einer Explosion als Belohnung. Aus Sicht der verfügbaren Ressourcen soll durch die Explosion lediglich das Freiwerden dieser verdeutlicht werden.

6.3.2 Sharing

Als Open-World Game liegt die Erkundung der Umgebung von *inFAMOUS 2* im Mittelpunkt zwischen der Absolvierung der Story-Missionen. Die benutzergenerierten Missionen haben für die Entwickler den primären Nutzen die Game World mit zusätzlichen Missionen während der Story als auch darüber hinaus zu populieren. Als solche befinden sich die Auswahlen dieser Missionen analog den Missionen der Kampagne als Diegetic Representations verteilt über die Game World (siehe Abb. 6.22). Bei Betreten werden zusätzliche Informationen wie Missionsname und Bewertung als Spatial Representation im Game Space dargestellt. Missionen anderer User werden je nach Standort des Spielers innerhalb der Game World aktualisiert und auf der Karte des Spiels dargestellt. Die Missionen werden Teil der Game World eines jeden Spielers. Einstellungen bezüglich Filterung und Präferenzen erfolgen über das Menü. Rezensionen werden im HUD dargestellt. Der Start der Mission erfolgt durch Druck auf den Controller. Alternativ kann der Spieler die Mission downloaden und im Editor selbst verfeinern (remixen).

Abbildung 6.22: Das Löschen von Objekten wird unabhängig von dessen Typ als massive Explosion und dem entsprechenden auditiven Feedback präsentiert.



(a) (b)

Abbildung 6.23: Die Game World des Spielers wird mit Missionen anderer Spieler populiert. Die Darstellung erfolgt analog den Kampagnen-Missionen (a). Bei Betreten der Missions-Auswahl wird die Missionsbeschreibung und Bewertung als Spatial Representation im Game Space angezeigt. Benutzergenerierte Missionen werden Teil eines jeden Spielers Game World und auch entsprechend auf dessen Karte (b) eingezeichnet.

6.3.3 Zusammenfassung

Der Editor ist eine Art Parallelwelt bzw. Gedankenwelt des Hauptcharakters. Diese kann zunächst leer oder durch Missions-Templates vordefiniert sein. Die Templates dienen als Hilfestellung und als schnelles Erfolgserlebnis. Wenngleich der Editor mit Hintergrundgeräuschen und Musik untermalt

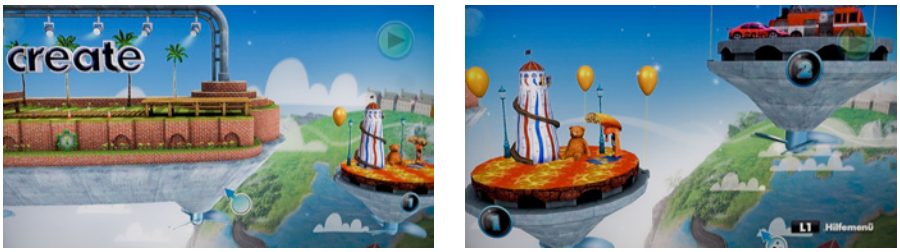
wird, fehlt jedwedes organisches Leben. Steuerung und Elemente des HUDs wurden aus dem Play-Part übernommen. Der Spieler wird als Cursor in der Game World repräsentiert, der auf Grund seiner Eigenschaften wie Kollisionen mit der Umgebung als einfache Form des Hauptcharakters angesehen werden kann. Die Darstellung von Einstellungen und Ressourcen als auch der Aufruf von Aktionen erfolgt über das HUD. Die Game World wird zur visuellen Unterstützung der räumlichen Platzierung verwendet. Dies erfolgt z. B. wie bei den Logikboxen durch Verknüpfung dieser durch Kabel. Das Löschen äu ert sich durch Explosion des jeweiligen gelöschten Objektes. Der Spieler wird geradezu verleitet Objekte zu löschen oder zumindest durch eine Explosion belohnt wenn er Objekte löschen muss. Das Sharing in *inFAMOUS 2* ist wie die Auswahl der Kampagnen-Missionen durch Repräsentationen auf der Map und in der Game World realisiert. Analog dieser wird dem Spieler über den Hauptcharakter die Wahl der Mission durch Betreten der Repräsentation in der Game World vermittelt.

6.4 Create

Create ist ein von *EA Bright Ligh*t entwickeltes Sandbox Game und wurde 2010 für diverse Platformen u. a. *PS3* veröffentlicht. Das Editieren und Erstellen von Inhalten ist Teil des Games und für das Foranschreiten in diesem essentiell. Die Bearbeitungsfunktionen des Games können daher als ein *Integrierter In-Game Editor* angesehen werden. Der Spieler muss durch die Verwendung und Kombination von Spielelementen durch das Game vorgegebene Herausforderungen meistern oder wird aktiv durch das Game aufgefordert die Game World zu verändern. Das Game ist unterteilt in verschiedene Szenen, der Spieler kann jedoch innerhalb dieser die Reihenfolge der Herausforderungen selbst bestimmen oder je nach Wunsch sich ausschlie lich der Game World widmen.

6.4.1 Creation

Das Hauptmenü ist zugleich die erste Szene und dient zur Navigation zu anderen Szenen. Nach einem einführenden Tutorial werden diese sichtbar (siehe Abb. 6.24). Der Spieler soll sich insgesamt als eine Art Künstler sehen der auf sein Werk blickt. Dies wird durch die Umsetzung der Szenen bewusst die an ein Diorama [72] erinnern. Die Architektur ist vorgegeben, die Ober ä-che, Hintergrund und der freie Raum für Objekte ist zunächst unbearbeitet. Der Spieler übernimmt die Aufgabe eines Innenarchitekten bzw. Dekorateurs und kann zu jedem Zeitpunkt nach Betreten der Szene diese gestalterisch mit Hilfe der Tools anpassen (siehe Abb. 6.25). Die Dekoration der Game World an sich repräsentiert in dieser Form auch den Fortschritt des Spielers, da mit weiterem Spielverlauf, weitere Ober ächen, Objekte und Pinsel freigeschaltet werden. Zusätzlich erhält der Spieler beim Abschluss der Szene seinen



(a) (b)

Abbildung 6.24: Das Hauptmenü und erste Szene (a) dient zugleich als erster Teil der Einführung und innerhalb der Game World als Navigation zu anderen Szenen (b).

(a) (b)

Abbildung 6.25: Der Spieler erhält zu Beginn eine unbearbeitete Szene (a). Der Aufbau der Szenen erinnert an Dioramen. Mit zunehmenden Spielfortschritt werden diese weiter dekoriert. Nach jeder Szene erscheint eine Statusübersicht des Spielers (b) basierend auf den durchgeführten Aktionen. Hierbei werden die Eigenschaften Erstellen, Lösen, Bearbeitung, Detail und Animation dargestellt.

kreativen Status angezeigt der sich auf Basis seiner durchgeführten Aktionen berechnet (siehe Abb. 6.25).

Die Repräsentation des Spielers ist der Cursor der ebenso wie die Auswahl der Tools und Anzeige der Tooltips im HUD gelöst ist (siehe Abb. 6.26). Die Interaktion mit der Game World erfolgt über eine zusätzliche Preview-Darstellung des zu setzenden Objektes im Game Space. Diese Darstellung folgt dem Cursor kollidiert jedoch mit Ausnahme der Darstellung von Spielobjekten nicht mit anderen Objekten innerhalb der Game World. Die Spielobjekte sind Teil der Challenges, deren Ziel meist die Erstellung einer umfangreichen Kettenreaktion ist. Im Play-Modus, welcher über das HUD oder Controller aktiviert wird, unterliegen die platzierten Spielobjekte der Physik. Die Kollisionen verdeutlichen diesen Sachverhalt bereits bei der Platzierung.

Abbildung 6.26: Alle für das Dekorieren relevanten Tools und Tooltips sind im HUD dargestellt. Die Auswahl der Tools erfolgt über ein Kreismenü. Die Auswahl der vorde nierten Challenges des Games werden über Icons im Game Space begonnen.

Spielobjekte für Challenges können nur im Vordergrund gebaut werden, die dekorierte Szene dient als Hintergrund der Action (siehe Abb. 6.27).

Challenges und Ziele des Games werden nicht im HUD sondern im Game Space bzw. bei Kettenreaktionen in der Game World dargestellt (siehe Abb. 6.26). Die Auswahl der Challenges zeigt sich als Form eines Zahnrades im Game Space. Der Spieler wird durch das Game zum Dekorieren durch so genannte Create-Ketten in Form eines Funken und Lichtstrahls aufgefordert (siehe Abb. 6.27). Der zu dekorierende Bereich steht in einer Art Spotlight im Mittelpunkt. Die notwendige Dekorationsform und Menge wird links oben im HUD durch ein funkelndes Icon dargestellt (siehe Abb. 6.27). Die insgesamte Menge an verfügbaren Objekten wird im HUD links als eine Art Ressourcenanzeige dargestellt. Diese Begrenzung dürfte auf Grund technischer Limitierungen eingeführt worden sein. Bei Abschluss der Create-Kette erhält der Spieler einen Create-Funken und u. a. neue Objekte. Diese tragen dazu bei eine neue Szene freizuschalten. Es kann hierbei also typisch für Sandbox Games zwischen Dekoration, Challenges zum Voranschreiten innerhalb einer Szene oder zwischen den Szenen frei gewählt werden.

Der Abschluss des Zieles wird durch ein Feuerwerksgeräusch mit Funken und Feuerwerksexplosionen unterstützt. Die Tools werden bei Aktionen durch auditives Feedback unterstützt wie z. B. Spray-Geräusche beim Zeichnen des Hintergrundes oder Tiergeräusche beim Platzieren von Kreaturen. Kreaturen werden beim Platzieren aus einem Meter Höhe hinabgeworfen und

(a) (b)

Abbildung 6.27: So genannte Create-Ketten (a) werden durch einen Lichtstrahl und ein Glühen dargestellt. Die Darstellung des zur Erfüllung der Create-Kette notwendigen Tools erfolgt links oben im HUD. Gameobjekte (a) wie Zielpunkt, Windräder, Balloons usw. be nden sich meist auf der vorderen Ebene und sind Teil der Game World. Hintergrund und Architektur dienen als Dekoration der eigentlichen Aufgaben im Vordergrund. Der Raum zur Platzierung dieser Game Objekte (b) kann je nach Wahl des Objektes bzw. Tools auf einen Bereich beschränkt sein.

Abbildung 6.28: Verschiedene Formen der Platzierung von Objekten und Feedback im Interface. v.l.n.r.: Platzierung einer Achterbahn (Objekt an Cursorposition), Verknüpfung von Spielobjekten z. B. Balloon mit Geschenk, Platzierung einer Kreatur, Bemalen des Hintergrundes, Platzierung von P anzen mit dem Dekorierpinsel, Feedback einer Feuerwerksexplosion durch erfolgreiche Create-Kette und Status der aktuellen Create-Kette.

verdeutlichen ihre Ankunft in der Game World durch eine Landeanimation (siehe Abb. 6.28). Ähnlich agieren die P anzen die beim Platzieren aus dem Boden sprie en (siehe Abb. 6.28). Wenngleich diese organischen Objekte lebendiger wirken, dienen sie nur als Dekoration der Szene und interagieren untereinander nicht. Ein Raptor reagiert z. B. nicht auf ein Pferd in der Nähe.

6.4.2 Sharing

Das Sharing in *Create* erfolgt durch so genannte Snapshots der eigenen Szene. Entsprechend dieser Analogie werden Szenen auch als Galerie dargestellt

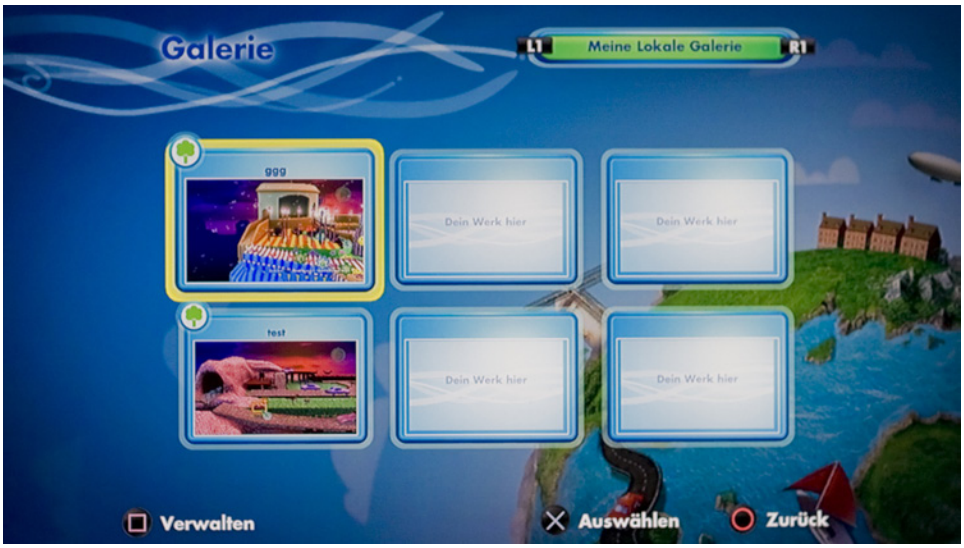


Abbildung 6.29: Sharing erfolgt in Form von Snapshots die zur Analogie passend in Galerien organsiert sind.

(siehe Abb. 6.29). Die Darstellung ist entsprechend dem geringen Ein uss auf die Spielerfahrung als klassiches Menü implementiert. Neue Szenen oder Challenges können nicht erstellt werden. Das Sharing dient ausschlie lich der Verteilung von dekorierten Szenen oder möglichst kreativen Lösungen der vorgegebenen Challenges.

6.4.3 Zusammenfassung

Create kombiniert das Spielen und Erstellen in einem Spielraum. Der Spieler wird durch das Game zur Dekoration und Lösung von Rätseln aufgefordert. Der Spieler schlüpft in die Rolle eines Innenarchitekten bzw. einer Art Schaufensterdekorateur. Zu keinem Zeitpunkt taucht der Spieler direkt in die Szene ein, vielmehr wirkt die Game World wie ein Diorama. Sie fungiert u. a. als Hintergrund der Rätsel die meist im Vordergrund ablaufen. Die Ergebnisse können Online getauscht werden. Bei der Bedienung stehen E ciency und Effectiveness im Fokus. Durch Funken, Animationen, Explosionen und eine auditive Untermalung wurde versucht das Feedback auf Aktionen und Änderungen im Status und der Ziele unterhaltsam zu gestalten.

Kapitel 7

Ergebnisse der Analyse

Dieses Kapitel befasst sich mit den in der Analyse in Kap. 6 identifizierten unterstützenden und unterhaltsamen Ma nahmen im User Interface von In-Game Editoren (Creation, Sharing). Auf Basis der Orientierungspunkte der Analyse (siehe Abschn. 5.2) wurden die Gemeinsamkeiten und Unterschiede zwischen den Analyseartefakten extrahiert. Die Gliederung erfolgt in konkrete Aspekte des Games deren gewählte Umsetzungen Ein uss auf die User Experience haben mögen und Aspekte von Editoren die diese bein ussen mögen. Sie können daher als Teil einer Menge von Aspekten betrachtet werden die bei der Konzeptionierung von unterhaltsamen In-Game Editoren und des gesamten Games durch den Designer evaluiert werden sollten. Die folgenden Abschnitte beschreiben, bewerten und begründen zusammenfassend diese Aspekte und die Unterschiede in den Artefakten.

7.1 Repräsentation des Spielers

Wie die Analyse der vier Games zeigte ist die Umsetzung der Repräsentation des Spielers im Game ein grundlegendes Kriterium bei der Konzeption eines In-Game Editors und dessen User Experience. Neben wichtigen Aufgaben wie der Navigation im Game Space und Interaktion beim Erstellen, Editieren und Löschen definiert die Repräsentation auch den möglichen Grad der Immersion des Spielers bzw. die Sicht des Spielers auf die Game World selbst.

Als Beispiel für eine gelungene Verschmelzung zwischen Spieler und Game World kann der Avatar in *Little Big Planet 2* herangezogen werden. Dieser ist allgegenwärtig in der Game World und mimt jede Aktion des Spielers. Dadurch entsteht eine verstärkte Projektion der eigenen Person auf den Avatar und steigert dadurch den Grad der Immersion. Die Konzeption des Avatars als Toy kann durch dessen Mechanics wie die Fähigkeiten des Laufens, Springens, Greifens, Verwendung der Werkzeuge und Individualisierung verdeutlicht werden. Zugleich dienen Teile dieser Mechanics auch der unterstützenden Bedienung des Editors. Der Avatar kann direkt mit der Game World

interagieren. Er ermöglicht es dem Spieler die Umgebung des Levels permanent zu validieren und ohne Zwischenschritte den Level zu testen. Auch dient der Avatar dem Feedback der Eingaben des Spielers durch seine Reaktionen oder Verhalten bei verschiedenen Materialitäten der Umgebung. Durch das Verhalten des Avatars im Bezug zur Game World und den Objekten innerhalb dieser lässt sich auf den aktuellen Modus des Editors schlie en. Werden die Modi für *LBP 2* im Detail betrachtet offenbaren sich die Nachteile bzw. Kompromisse die bei dieser Lösung eingegangen werden mussten. So tritt der Avatar im Kreativ-Modus auf eine Ebene vor dem eigentlichen Level, kann sich frei bewegen bzw. schweben. Hier wird es dem Spieler auch erlaubt den Bildausschnitt seinen Bedürfnissen anzupassen. Durch den Kreativ-Modus verliert der Avatar einen Teil seiner Bindung zur Game World. Aus Gründen der E ciency musste dieser Modus wohl integriert werden um bei steigendem Umfang den Anforderungen des Spielers gerecht zu werden und die Übersicht und schnelle Navigation zu bewahren.

Die dargestellte Lösung von *LBP 2* zeigt sehr gut wie ein Toy zugleich unterhaltsam und unterstützend genutzt werden kann. Wie unter Betrachtung der weiteren Games *ModNation Racers*, *inFAMOUS 2* und *Create* festzustellen ist, kann oder muss die Repräsentation des Spielers je nach Genre oder ästhetischem Ansatz variieren. Gegebenheiten eines Genres wie z. B. Steuerung, Perspektive oder Grenzen für den Spieler haben Ein uss auf die Realisierbarkeit der notwendigen Usability des Editors. Nicht jede Repräsentation aus dem Play-Part kann somit direkt im Editor verwendet werden. Ob eine Konzeption als Toy umsetzbar ist muss von Fall zu Fall evaluiert werden.

ModNation Racers z. B. versucht zunächst ähnlich *LBP 2* die Steuerung im Menü über den fahrenden Avatar zu realisieren. In dem Charakter- und Fahrzeug-Editor geschieht bereits ein Bruch zwischen Avatar und Game World. Dieser wird ausgelöst durch den ästhetischen Ansatz von Sammelfiguren und dem vom Charakter distanzierten HUD zur Bearbeitung von diesem. Die Kombination von Fahrzeug und Fahrer im Play-Part bilden den eigentlichen Avatar, welcher in diesen Editoren jedoch getrennt behandelt wird. Der Spieler wird dadurch in seiner Projektion auf den Avatar gestört.

Im Strecken-Editor von *ModNation Racers* wird die Trennung zwischen Avatar aus dem Play-Part und unterstützender Nutzung von diesem im Editor offensichtlich. Zunächst wird die Strecke mit der bekannten Steuerung aus dem Play-Part mit einer Asphaltiermaschine asphaltiert. Ein Ansatz der sich vom klassischen Bau über Puzzle-Teile entfernt. Die Verwendung dieser Analogie ermöglicht es dem Spieler ohne Aneignung von neuem Wissen den Modus des Editors und das Ziel zu erkennen und die Aufgabe durch die bekannte Steuerung schnell auszuführen. Zugunsten der E ciency erfolgt jedwede weitere Bedienung des Editors aus einer frei bewegbaren und drehbaren Vogelperspektive mit der Repräsentation des Spielers als Cursor im Game Space. Der Reiz und Fun mit dem Toy der Asphaltiermaschine

zu arbeiten wird nicht weiter fortgesetzt. Die Verwendung des Avatars stö t z. B. bei der Landschaftsdeformation an seine Grenzen und wäre mit dem Paradigma des Fahrzeuges auf der Strecke nicht vereinbar.

Ebenfalls nicht vereinbar wäre ein 1:1 aus dem Play-Part übernommener Avatar im Editor von *inFAMOUS 2*. Die Game World wird auf die relevanten Inhalte, primär die Umgebung, reduziert. Personen und Fahrzeuge des Open-World Games könnten den Spieler in seinem Werken ablenken oder unvorhergesehe Zustände im Editor hervorrufen die vom Spieler nicht reversierbar wären. Zumal Personen und Fahrzeuge fehlen wäre es durch den Spieler nicht akzeptabel und nicht vereinbar diesen in Form seines Avatars darzustellen.

Der Spieler wird im Editor von *inFAMOUS 2* durch einen Cursor in der Game World repräsentiert. Durch die Verbindung und dem Wechsel zwischen der Position des Spielers im Play-Part und dem Cursor im Editor entsteht eine für den Spieler plausible Projektion auf den Cursor. Der Cursor kann auf Grund seines dem Play-Part analogen Verhaltens wie z. B. die Kollision mit Wänden oder Hindernissen, als Teil der Game World und nicht ausschlie lich des Game Spaces gesehen werden. Der Spieler fühlt sich dadurch stärker mit dem Game verbunden. Zusätzlich hilft die Repräsentation durch das Verhalten auch beim Ausschluss von Fehlerquellen bei der Platzierung. Durch die dem Play-Part analoge Steuerung muss allerdings das Fehlen jeglichen Ein usses auf die Kameraeinstellungen der Darstellung kritisiert werden.

Create distanziert den Spieler vollkommen von der Game World. Diesen durch jegliche Repräsentation innerhalb der Game World darzustellen wäre unplausibel auf Grund der Ästhetik. Diese orientiert sich an Dioramen. Der Spieler blickt von au en in die Szene. Er setzt und manipuliert Inhalte ausschlie lich über den Cursor auf der Ebene des HUDs. Er nimmt dadurch eine distanzierte und beobachtende Position zur Game World ein. Neue Objekte innerhalb der Game World folgen dem Cursor. Dadurch entsteht selbst beim Platzieren von Objekten keine direkte Verbindung.

7.1.1 Zusammenfassung

Eine Wahl der verschiedenen Repräsentationsmöglichkeiten des Spielers im Game bzw. Editor sollte in Abhängigkeit zwischen dem Play-Part und dem ästhetischen Zugang, sowie den Eigenheiten des Genres und der gewünschten E ciency des Editors durch den Entwickler abgewägt werden. Wie *LBP 2* zeigt ist es erstrebenswert die Repräsentation als Toy und analog dem Play-Part zu konzeptionieren und den Spieler möglichst tief in den Editor eintauchen zu lassen. Kompromisse und Zugeständnisse zugunsten einer besseren Bedienbarkeit können nicht verhindert werden und sollen als solche nicht als negative Entscheidung angesehen werden. Insbesondere für fortgeschrittene Spieler wird diese zunehmend wichtiger, daher ist auch das Angebot von schnelleren und weniger spielhaften Darstellungen sinnvoll.

7.2 Repräsentation von Tools

Die Repräsentation der Tools steht in Abhängigkeit der Repräsentation des Spielers. Die Tools sind das Interaktionsmittel des Spielers zur Erstellung, Bearbeitung und Verknüpfung der Inhalte bzw. der Spielobjekte. Mit Einbeziehung der Mechanics der Tools selbst könnten diese als Toy betrachtet werden. Zum Beispiel ist das experimentierende Verwenden von Seilen in *Little Big Planet 2* durchaus unterhaltsam. Zwischen der Repräsentation der Tools und den Mechanics der dahinterstehenden Funktionen sollte jedoch differenziert werden.

Die Repräsentation von Tools bzw. damit verbundener Spielobjekte erfolgt in allen vier Games zumindest innerhalb der Game Spaces. *ModNation Racers* und *inFAMOUS 2* verwenden die Repräsentation des Spielers zur Anzeige dieser. *Create* verwendet wohl auf Grund des au erhalb der Game World befindlichen Spielers je nach Tool ein zusätzliches Vorschau-Objekt zur genauen Platzierung der Spielobjekte. Diese Arten der Repräsentation sind vergleichbar mit grafischen Editoren in Anwendungssoftware. Die Wahl der Repräsentation der Tools scheint ein unmittelbares Produkt der Repräsentation des Spielers, zumal die Aufmerksamkeit des Spielers permanent auf dieser liegt.

Die Verbindung zwischen Spieler und Avatar bzw. die Immersion wäre bei *LBP 2* gestört wenn sich der Avatar z. B. zu einem Spielobjekt transformieren würde. Gewisserma en ist der Avatar ein Toy, dass ausschlie lich zur Navigation im Editor dient. Zusätzlich wurde daher das Platzieren und Selektieren über einen Cursor gelöst. Dessen Darstellung variiert je nach Auswahl des Tools. Das Problem aus Sicht des Spielers ist die Trennung zwischen den Orientierungspunkten Avatar und Cursor. Der für den Spieler nachvollziehbare und plausible Wechsel der Kontrolle zwischen Avatar und Cursor dürfte hierbei der Grund für die Einführung des Lassos gewesen sein. Das Lasso verbindet den Avatar mit dem Cursor des ausgewählten Tools. Es ist damit zugleich das Merkmal des Wechselns der Kontrolle als auch Orientierungslinie zurück zum Avatar. Die Steuerungsbewegung des Avatars verstärkt das Gefühl der Kontrolle über das Tool. Das Lasso selbst könnte durch seine auffällig schwingende Bewegung als Toy bezeichnet werden.

Das Lasso wurde zudem als Verbindung zu komplexeren Einstellungen, die in Form eines 2D-Interfaces im HUD dargestellt werden, verwendet. Dadurch wird jedes Optionenmenü Teil der Game World was insgesamt der Immersion zugutekommt. Eine ähnliche Verbindung zu den Einstellungen fehlt in *ModNation Racers*, *inFAMOUS 2* und *Create*. Deren Einstellungen werden im HUD dargestellt. Im Sinne der Effectiveness und E ciency kann hierbei kein Unterschied zwischen *LBP 2* und den anderen Games festgestellt werden. Ohne dem Bezug zur Game World entsteht jedoch keine tiefere Bindung zu dieser. *ModNation Racers*, *inFAMOUS 2* und *Create* versuchen diesen Umstand beim Platzieren oder Löschen von Objekten durch Feedback in

Form von Partikeleffekten, Animationen und passenden Sounds entgegenzuwirken. Zum Beispiel sprie en P anzen bei *Create* nach dem Platzieren aus dem Boden und wachsen. Insbesondere bei *Create* lockert dieses Feedback die sonst statische Szenerie.

Ein weiterer gemeinsamer Ansatz zwischen den Games zeigt sich in der Darstellung von logischen Verknüpfungen im Game Space. Es erscheint als logischer Schritt Verknüpfungen der Spielobjekte direkt im selben Raum darzustellen und keine abstraktere Form z. B. textuelle Darstellung der Ein- und Ausgänge zu benutzen. *ModNation Racers* benutzt eine simple und funktionelle Linie zur Gestaltung der Verbindung. Diese ist ausreichend um die Funktion abzubilden. *LBP 2* und *inFAMOUS 2* hingegen verwenden dreidimensional dargestellte Kabel analog des übergeordneten ästhetischen Konzeptes des Games bzw. der Analogie zur realen Welt. Letztere beiden Lösungen sind innerhalb der Game World, benötigen aber zusätzliche Ressourcen des Systems zur Darstellung. Die Lösung von *LBP 2* dient unterdessen durch die Präsentation der Signalwege im Kabel als Mittel zur Fehlerfindung, der Präsentation des aktuellen Modus des Editors und als Feedback auf die Zustandsänderung von z. B. einem Schalter. Auf Grund der frei wählbaren Blickrichtung und der weitreichenden Umgebung in *inFAMOUS 2* kann diese Umsetzung schnell unübersichtlich werden.

7.2.1 Zusammenfassung

Die Darstellung der Tools und damit verbundenen Spielobjekten sind indirekt abhängig vom Genre, vielmehr direkt abhängig von der Repräsentation des Spielers. Durch die Tools kann je nach Repräsentation des Spielers die Verbindung mit der Game World verstärkt werden. Das kann z. B. durch Feedback wie Partikeleffekten oder Animationen entsprechend dem Tool oder Spielobjekt bewerkstelligt werden. Verbindungen zwischen Spielobjekten sollten im gleichen Raum dargestellt werden. Die Tools als auch Spielobjekte sollten interne Zustände von z. B. Verbindungen, Modus des Editors als auch eventuelle Fehler beim Platzieren abbilden. Insgesamt sind die Repräsentationen der Tools mehr auf Funktionalität als auf Umsetzung in Form von Toys ausgelegt. Ob eine Umsetzung als Toy unter Bewahrung der E ciency möglich ist und der Aufwand im Verhältnis zum Nutzen steht sollte bei der Konzeption geklärt werden.

7.3 Repräsentation von UGC

Die Repräsentation von UGC beschreibt die Abbildung der Funktionalitäten des Sharings wie das Verteilen, Erhalten, Filtern und Bewerten von Inhalten. Wie sich bei Betrachtung der vier Games zeigt ist das Sharing inbesondere mit dem Augenmerk auf E ciency umgesetzt.

ModNation Racers und *Create* lösen das eigentliche Sharing mit einem klassischen Ansatz über 2D-Interfaces. Die Umsetzung zeigt dabei die Vorteile einer guten Gesamtübersicht und schneller Erreichbarkeit der angebotenen Inhalte. Zumindest *Create* versucht das Sharing als Analogie in Snapshots und Galerien zu benennen und damit eine ästhetische Glaubhaftigkeit zu gewährleisten. Während in *Create* das Sharing als Feature an sich niedrigere Priorität besitzt und durch den 2D-Charakter mit der Repräsentation des Spielers harmoniert, verhindert die Darstellung in *ModNation Racers* eine Bindung unter den Spielern selbst. Vielmehr stehen hier die Inhalte im Vordergrund. Die Darstellung ist mehr Mittel zum Zweck und ist wenig mit dem eigentlichen Game bzw. Editor verbunden.

Eine gelungene Verbindung zwischen Game und dem Erhalt von Content zeigt *inFAMOUS 2*. Story-Missionen werden in gleicher Weise wie Missionen anderer Spieler innerhalb der Game World dargestellt. Diese Variante der Inhaltsbeschaffung und Kombination zwischen Game und UGC ist abhängig vom Genre. Sie eignet sich besonders für Games deren Spielumgebung wie z. B. Landschaften und Gebäude zwischen Play-Part und Editor unverändert bleibt.

Wie *Little Big Planet 2* zeigt, kann das Sharing an sich aber auch unterhaltsam und Teil der Erfahrung mit dem Spiel sein ohne auf E ciency zu verzichten. Der UGC eines jeden Spielers wird auf dessen Planeten (online) oder Mond (lokal) repräsentiert. Diese können durch ihre Gestaltungsmöglichkeiten selbst als Toy angesehen werden. Andere Spieler verschaffen sich über den Planeten zutritt zu den Inhalten eines Spielers. Der Planet dient als Visitenkarte des Spielers und repräsentiert dessen investierten Aufwand und Kreativität. In *LBP 2* schlie t sich hier das ästhetische Gesamtkonzept zu einem homogenen Ganzen. Das Konzept eines eigenen Spielraumes für jeden Spieler zur Präsentation liese sich auch auf andere Genres übertragen, z. B. durch eine Garage in *ModNation Racers*. Der Spieler selbst würde wieder zunehmend in der Vordergrund rücken.

Der beschriebene Fokus von *ModNation Racers* auf die Inhalte der Spieler zeigt sich auch in der besonderen Umsetzung der Top 3 Charaktere, Fahrzeuge und Strecken innerhalb der Game World des Hauptmenüs. Die Darstellung erfolgt durch Statuen auf einem Siegertreppchen die den erfolgreich produzierenden Spielern ein Denkmal innerhalb der Game World anderer Spieler setzen. Ein Nutzen aus Sicht der E ciency wird durch eine Art von Shortcut zum jeweiligen Werk bzw. dem 2D-Menü erreicht.

7.3.1 Zusammenfassung

Benutzergenerierte Inhalte bzw. die Sharing-Funktionalitäten sollen durch die Repräsentation von UGC schnell erreichbar sein. Bei der Konzeption sollte die Darstellung über ein Toy evaluiert werden und weiters welche Anreize dem Spieler allgemein zur Erstellung von Inhalten durch die Umsetzungen

gegeben werden können. Der Fokus der Repräsentation von UGC kann auf den erstellten Inhalten oder übergeordnet auf den Spielern selbst liegen. Die Wahl der Umsetzung ist weniger abhängig vom Genre als vielmehr vom gewählten ästhetischen Gesamtkonzept.

7.4 Repräsentation von Progress und Status

Die Analyse brachte neben dem Spieler, Tools und UGC weitere Aspekte bzw. Elemente des Interfaces hervor die insbesondere im Zusammenhang mit der Darstellung von Progress und Status stehen. Progress und Status stehen nicht auschlie lich im Zeichen des Spielfortschrittes und Status des Spielers im Play-Part sondern sind vielmehr auch Ausdruck für den aktuellen Fortschritt und Zustand bzw. Modus des Editors und dessen Game World selbst. Abschlie end werden daher stellvertretend auffällige Aspekte herausgegriffen.

7.4.1 Editormodus

Little Big Planet 2 und *Create* kommunizieren den Zustand bzw. Modus des Editors mitunter durch die Physik der Spielobjekte die sich je nach Spielmodus unterscheidet. Bei *inFAMOUS 2* übernimmt das Fehlen von Spielobjekten diese Aufgabe. Ähnlich wie in *inFAMOUS 2* und *LBP 2* kommuniziert auch die Repräsentation des Spielers und dessen Verhalten den aktuellen Modus des Editors. Im Preview-Modus von *ModNation Racers* z. B. befindet sich der Spieler wieder auf der Strecke. Dem Spieler wird die Information des Status und Progress nicht ausschlie lich über das HUD mitgeteilt. Allgemein lässt sich daraus ableiten, dass unter Beachtung des übergeordneten Konzeptes des Editors und Games jedwedes Element innerhalb der Game World zur Darstellung des aktuellen Zustandes genutzt werden sollte. Weitere Interface-Elemente im HUD sollen dadurch minimiert werden und der Grad der Immersion des Spielers gesteigert werden. Wie die vier Games jedoch auch zeigen sind die Möglichkeiten die Elemente für den Spieler sofort erkennbar und offensichtlich zu gestalten je nach Genre und Spielwelt begrenzt. Beispielsweise benötigt *inFAMOUS 2* das Weglassen von Personen nur zur Feststellung einer Trennung zwischen Play-Part und Editor. Der subtile Ansatz ist vertretbar. Für *LBP 2* und *Create* mussten als Kompromiss zusätzliche Elemente z. B. ein Play oder Pause Icon im HUD integriert werden um dem Spieler den Modus unmissverständlich zu signalisieren.

7.4.2 Fehlerprävention

Das Vermeiden von Fehlern bei der Platzierung von Spielobjekten gestaltet sich über alle Games hinweg in ähnlicher Form durch z. B. rotes Einfärben der Repräsentation des Tools und Abspielen eines Audioclips. *ModNation Ra-*

cers bindet ähnlich *inFAMOUS 2* den Cursor je nach Tool an den validen Bereich im Game Space wie z. B. die Stra e beim Asphaltieren oder beim Autopopulieren der Spielobjekte. Auffällig für alle Games ist die schlichte Darstellung einer invaliden Platzierung. Die Verwendung von Effekten und Animationen würde auf Dauer störend wirken. Ein Fehlen von Feedback des Avatars in *LBP 2* erklärt sich durch den Verlust der Kontrolle des Spielers. Die Projektion auf den Avatar wäre gestört.

7.4.3 Undo und Redo

LBP 2 unterscheidet sich zu den anderen Games durch eine hybride Undo-Funktionalität. *Create* und *ModNation Racers* benutzen einen traditionellen und schnellen Ansatz mit diskreten Sprüngen zwischen den Zuständen. *LBP 2* lässt sich zum letzten Zustandswechsel zwischen Play und Pause zurückversetzen. Die Funktion wird auditiv und durch ein visuelles Feedback in Form eines Videospul-Effektes bekannt aus VHS-Systemen als Spatial Representation unterstützt. Die Zustände sind diskret, die durchgeführten Bewegungen sind aber kurzzeitig sichtbar. Der Spieler kann dadurch seinen zurückgelegten Weg und seine Aktionen nachvollziehen. Er erhält einen schnellen Überblick über die aktuelle Situation nach dem Wechsel. Die Umsetzung von Zustandswechseln kann prinzipiell als genreunabhängig betrachtet werden. Während *ModNation Racers* und *Create* einen klassischen Ansatz durch schnelle Reaktion umsetzen, versucht *LBP 2* die Veränderung durch die ästhetische Umsetzung zu verdeutlichen. Dem Spieler wird hierbei ein besonders gutes Gefühl der Reise zurück in der Zeit vermittelt. Die Lösung von *inFAMOUS 2*, eine bildschirmfüllende Explosion beim Löschen, verdeutlicht auf übertriebene Weise eine Veränderung. Zu Beginn wirkt diese unterhaltsam, im weiteren Verlauf durch Repetition zunehmend als störend.

7.4.4 Ressourcen

Eine Gemeinsamkeit über alle vier Games hinweg zeigt sich in der Anzeige der zur Verfügung stehenden Ressourcen. Diese werden permanent als Interface Element des HUDs dargestellt. Es wird kein Versuch unternommen diese in der Game World darzustellen. Es handelt sich hierbei um eine Information die ausschlie lich für den realen Spieler bestimmt ist. Ressourcen sind eine limitierende Einheit im jeweiligen Editor die den Spieler in seinem Schöpfungsdrang einschränken können. Es ist daher wichtig diese Information permanent und sichtbar zu vermitteln.

Kapitel 8

Diplomprojekt Little Long Finger

Das Diplomprojekt *Little Long Finger* (LLF) umfasste die Umsetzung eines In-Game Editors des *Play, Create und Share Genres*. Dieses Kapitel beschreibt den Ein uss der Ergebnisse aus der Analyse auf das Diplomprojekt. Die aus der Entwicklung des Projektes gewonnenen Erkenntnisse im Bezug auf die Repräsentation von Spieler, Tools, Status und Progress ergänzen das Ergebnis der Diplomarbeit. Das Kapitel beginnt zunächst mit der Einführung in das Diplomprojekt und der gesetzten Zielsetzung und geht anschlie end über zur Beschreibung des *Play*-Parts. Im Anschluss folgt in Abschn. 8.3 die Beschreibung des *Create*-Parts und die Au istung funktionaler Anforderungen. Der Verlauf der Konzeption und Umsetzung sowie der Vergleich zwischen den Konzepten und mit den Ergebnissen der Diplomarbeit bilden den Abschluss dieses Kapitels.

8.1 Einleitung und Zielsetzung

Der Kern des Diplomprojektes *Little Long Finger* ist der In-Game Editor zur Erstellung neuer Levels für das Game. Die Zuordnung kann in das *Play, Create und Share Genre* getroffen werden. In-Game Editoren können unter den Voraussetzungen (siehe Abschn. 4.1.3) von z. B. Regeln und Grenzen als Sandbox Games bezeichnet werden. Wie Breslin in [44] feststellte, erfordert die Entwicklung von Sandbox Games, einen erhöhten Produktionsaufwand. Der *Play*-Part wurde daher in Abschn. 8.2 spezifiziert und als fiktiv angenommen. Diese fiktive Spezifikation diente als Ausgangslage zur Konzeptionierung des In-Game Editors.

Wie die Ergebnisse der Analyse in Kap. 7 hervorbrachten, können die Repräsentationen von Spieler, Tools, UGC, Progress und Status im User Interface Ein uss auf den Unterhaltungswert eines In-Game Editors haben. Die Repräsentationen im User Interface sollten bei der Konzeptionierung

eines In-Game Editors evaluiert werden. Primäres Ziel des Projektes war daher die Umsetzung und Evaluierung von Konzepten zur Repräsentation des Spielers und der Tools im In-Game Editor. In die Konzeptionierung sollten die Ergebnisse der Diplomarbeit ein ie en. Das Ergebnis des Projektes sollte als Ergänzung zum Ergebnis der Analyse dienen.

Zum Erhalt der Projektergebnisse wurden verschiedene Konzepte des *Create*-Parts (siehe Abschn. 8.3) zur Repräsentation des Spielers und der Tools prototypisch umgesetzt. Auf Sharing-Funktionalitäten und damit Repräsentation von UGC wurde auf Grund des Umfanges explizit verzichtet. Der Begriff prototypisch bezieht sich auf die Implementierung wesentlicher Funktionen eines Editors (siehe Abschn. 8.3.1). Der Entwicklungsverlauf der Konzepte wurde dokumentiert (siehe Abschn. 8.3.2).

Als Projektergebnis wird in Abschn. 8.4 ein Vergleich zwischen dem Entwicklungsverlauf der Konzepte und Vor- und Nachteile dieser mit Bezug zu den Repräsentationen von Spieler, Tools, Progress und Status aufgestellt.

8.2 Little Long Finger: *Play*

LLF kann als ein *Third-Person Puzzle Game* bezeichnet werden. Das ästhetische Gesamtkonzept orientiert sich an einem Brettspiel am heimischen Wohnzimmertisch. Mit dem Avatar muss das Ende des Levels erreicht werden. Levels bestehen aus Räumen und freien Flächen. Diese sind bestückt mit Hindernissen wie z. B. Lichtschranken die das Ankommen erschweren.

Der Avatar wird durch einen magischen Ring zum Leben erweckt. Er selbst wird als Finger dargestellt. Zum einen soll der Avatar den Spielnamen und Spielablauf verdeutlichen und zum anderen bereits als Analogie zum Spielbrett am heimischen Wohnzimmertisch dienen. Die Bewegung des Avatars erfolgt analog zwischen den Feldern des diskret gerasterten Spielfeldes (siehe Abb. 8.1). Die Bewegung kann ausschlie lich horizontal oder vertikal und auf bebauten Feldern erfolgen. Die Ansicht der Kamera zeigt Avatar und Spielfeld in einer perspektivischen Seitenansicht in einer 60° Rotation zum Tisch (siehe Abb. 8.1).

Für ein optimales Ergebnis des Levels müssen alle Sammelgegenstände wie z. B. Diamanten eingesammelt werden und der finale Ring erreicht werden. Um das Ziel sicher zu erreichen und den Fallen auszuweichen, besitzt der Avatar eine Re ex-Fähigkeit um die Spielzeit zu verlangsamen.

Als Zielgruppe sind Casual-Player im Alter von 8 14 Jahren definiert. Die Zielplattform des Games ist der PC. Von einer Umsetzung für z. B. eine mobile Plattform wurde abgesehen, um spezielle Unterschiede im User Interface die sich durch die Touch-Eingabe ergeben würden zu vermeiden.

Abbildung 8.1: Das Spielfeld de niert sich durch ein quadratisches Raster. Die Kamera be ndet sich in einer 60° Rotation über dem Avatar.

8.3 Little Long Finger: *Create*

Als Ausgangslage des *Create*-Parts (In-Game Editor) von *LLF* dient der zuvor beschriebene fiktive *Play*-Part. Der *Create*-Part ist als Vertreter des *Play, Create und Share Genres* als Core-Feature des Games zu betrachten. Gemä der Zielsetzung für das Projekt wurden für den Editor zunächst drei mögliche Konzepte *Classic*, *Master* und *Avatar* (siehe Abschn. 8.3.2) zur Realisierung formuliert. Die Konzepte wurden Schritt für Schritt mit den Erkenntnissen (siehe Kap. 7) aus der Analyse (siehe Kap. 6) modifiziert und erweitert. In den folgenden Abschnitten werden die gesetzten funktionalen Anforderungen des Editors aufgelistet und der Entwicklungsverlauf der Konzepte beschrieben.

8.3.1 Funktionale Anforderungen

Die Editor-Konzepte benötigten eine einheitliche Liste an umzusetzenden funktionalen Anforderungen. Dadurch konnte ein besserer Vergleich der Konzepte zueinander gewährleistet werden. Die folgenden funktionalen Anforderungen definieren den Kern des umgesetzten Editors:

- Navigation durch Bewegung der Ansicht
- Selektieren von Tools und Objekten
- Platzieren von Objekten
- Editieren von Objekten
- Rotieren von Objekten
- Löschen von Objekten
- Wechsel zwischen Play und Create

8.3.2 In-Game Editor Konzepte

Die folgenden Abschnitte beschreiben die Idee hinter den Konzepten *Classic*, *Avatar* und *Master*[1]. Der Verlauf der Entwicklung und Argumentation hinter den Entscheidungen wird zusammengefasst.

Classic

Der Ausgangspunkt des *Classic*-Konzeptes ist vergleichbar mit der Umsetzung des User Interfaces in *EAs Create*. Der Spieler wird repräsentiert durch einen 2D-Cursor im HUD. Die Selektion von Tools und Objekten erfolgt ebenfalls im HUD. Damit ist der Spieler au erhalb der Game World und des Game Spaces repräsentiert. Der äu ere Blickpunkt in *EAs Create* entstand durch die Analogie der Dioramen. Der *Play-Part* von *LLF* spielt sich jedoch direkt am Spielfeld durch die Steuerung des Avatars. Es stellte sich die Frage ob und wie der Cursor ein Teil der Game World oder zumindest des Game Spaces werden kann, um die Bindung und das Eintauchen in den Editor zu verstärken. In einer Weiterentwicklung des Konzeptes wurde der Cursor als 3D-Hand (siehe Abb. 8.5) und das Menü als 3D-Menü innerhalb des Game Spaces abgebildet. Dadurch sollte der Immersionsgrad des Spielers erhöht werden. Die Repräsentation des Spielers als Hand im *Create*-Part und als Avatar im *Play*-Part lässt sich ausgezeichnet mit dem Aufbau eines Brettspieles und dem eigentlichen Eintauchen in das Spielgeschehen vergleichen.

Bei der Umsetzung einer 3D-Hand wie sie aus *Black & White 2*[2] (2005, Lionhead Studios) bekannt ist und einem 3D-Menü innerhalb des Game Spaces wurden mehrere Probleme des Interfaces festgestellt. Die Darstellung des aktuell gewählten Tools durch eine in der Hand befindliche Seifenblase mit Inhalt (siehe Abb. 8.2) konnte schwer identifiziert werden. Die gewählte Form der Darstellung des Vorschau-Objektes limitiert die mögliche Abbildungsgrö e. Ein 1-Feld-Objekt wie z. B. ein Stuhl kann je nach Markanz der Form noch erkannt werden. Zur erkennbaren Darstellung von Mehrfeld-Objekten wie z. B. einem Raum reicht der verfügbare Platz nicht aus. Versuche das Menü im Game Space abzubilden konnten nicht ausreichend e zient gelöst werden. Es wurden Ansätze einer Toolbox neben dem Spielfeld oder einer beweglichen Toolbox auf dem Spielfeld geprüft (siehe Abb. 8.3). In beiden Varianten konnte die Toolbox nicht schnell genug durch den Cursor erreicht werden. Es wurde daher eine dritte Variante durch eine ins das Bild iegende 3D-Spielbox prototypisch umgesetzt. Diese war e zienter als die ersten beiden Ansätze führte jedoch zu einer Sichtblockierung des Spielfeldes. Die Frage nach einer strukturierten Abbildung des Menüs wurde durch die Probleme der E zienz nicht tiefer behandelt. Das Konzept das Menü als Toy umzusetzen musste letztendlich verworfen werden.

[1] Wurde in der Konzept-Phase verworfen.

[2] http://www.lionhead.com/bw2

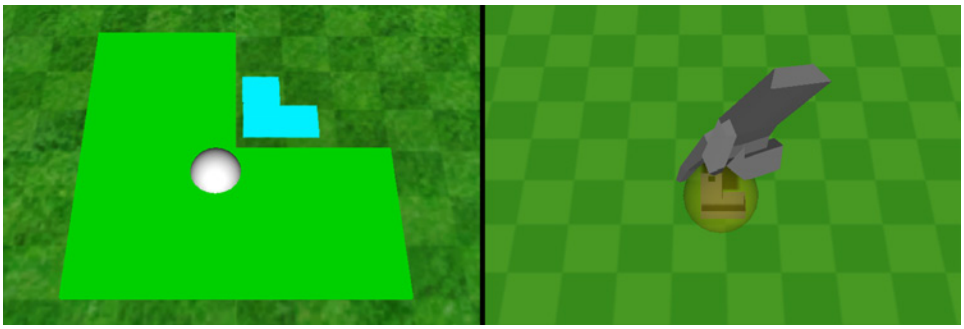


Abbildung 8.2: Die Vorschau des aktuellen Objektes (vor allem von Mehrfeld-Objekten) oder Tools in einer Seifenblase konnte auf Grund der limitierten Abbildungsgrö e und Sichtbehinderung nicht identi ziert werden. v.l.n.r.: Erste Umsetzung zur Feststellung der Grö enverhältnisse und Umsetzung mit einer vorläu gen 3D-Hand mit Blase.

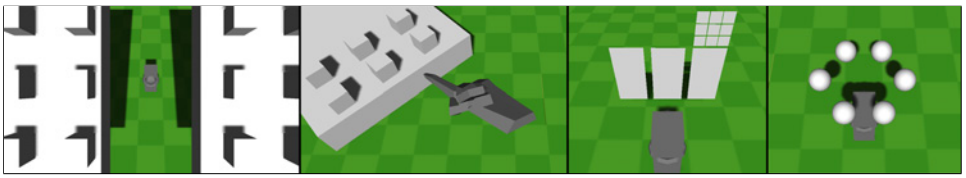


Abbildung 8.3: Eine Auswahl verschiedener Varianten zur Darstellung des Menüs innerhalb und au erhalb der Game World. Die Varianten verursachten Nachteile in der Visualisierung der aktuellen Auswahl und Sichtbereich des Spielfeldes. v.l.n.r.: Classic: Animierte Spielboxen, Classic: Toolbox am Spielfeld, Avatar: Spielkarten in Front und Avatar: 3D-Repräsentationen über dem Avatar.

Als Konsequenz daraus wurde die Abbildung des Menüs wieder im HUD durchgeführt. Die 2D-Darstellung des Menüs wurde mit einem dreidimensionalen Look versehen (siehe Abb. 8.5). Der Look ist als Anspielung auf eine Spielbox mit Spielsteinen gedacht. Die Anzeige erfolgt im unteren Bereich auf Grund des geringeren Weges des Cursors und durch die Präferenz der horizontalen Objektanordnung am Spielfeld. Das Menü selbst wurde hierarchisch in Gruppen und Untergruppen organisiert. Es dient der Kommunikation von Progress und Status indem sich beispielsweise Inhalt und Farbe der Schachtel je nach verfügbaren Optionen verändern.

Die 3D-Hand wird auch im Menü dargestellt und nicht durch einen 2D-Cursor ersetzt. Es soll dadurch von dem Menü der Eindruck entstehen Teil des Game Spaces zu sein. Es könnte als Shortcut zu einer anderen Position im Game Space angesehen werden.

Die Hand übernimmt u. a. durch seine Animationen die Rolle eines Toys. Diese sollten den Cursor beim z. B. Aufheben oder Platzieren von Objekten

Abbildung 8.4: Durch intelligentes Ein- und Ausblenden von Objekten können auch verdeckte Objekte selektiert werden.

im Vergleich zu einem klassischen 2D-Cursor zusätzlich unterhaltsam gestalten. Weitere ein Toy unterstützende Funktionsteile wie z. B. das Löschen durch eine Wurfgeste wurden getestet, erwiesen sich jedoch für den Spieler als zu fehleranfällig und aufwendig bei wiederholter Anwendung. Die 3D-Hand gibt durch die Animationen und durch Farbe und Darstellung Feedback auf die durchgeführten Aktionen und aktuelle Optionen. Wird ein Objekt oder ein Tool selektiert wird der Zeigepunkt der Hand mit einer 3D-Darstellung des Objektes oder Tools ergänzt. Im Gegensatz zu der Darstellung der Objekte als Teil einer Seifenblase in der Hand kann der Spieler durch diese Umsetzung das aktuelle Objekt in seinem Level besser visualisieren. Die Entscheidung für die nächste Aktion kann besser getroffen werden. Zusätzlich zu der 3D-Darstellung der Objekte wird am Spielfeld eine 2D-Darstellung angezeigt um Fehler bei der Platzierung zu erkennen.

Die Kamera-Ansicht entspricht jener aus dem *Play*-Part mit zusätzlich vergrö ertem Sichtbereich. Durch die Neigung der Kamera entstand ein Problem mit der Sichtbarkeit von Objekten hinter Objekten. Als eine mögliche Lösung wurde die Verwendung der Vogelperspektive evaluiert. Diese wurde durch die Einführung eines intelligenten Ein- und Ausblendens von Objekten vor dem Spieler oder Cursor wieder verworfen (siehe Abb. 8.4). Der Spieler kann dadurch, ohne in den *Play*-Modus zu wechseln, den Level in finaler Perspektive betrachten.

Der Wechsel in den *Play*-Modus erfolgt analog der Platzierung von Objekten. Über einen Hauptpunkt im Menü kann die Spielfigur ausgewählt werden und auf einer validen Stelle am Spiefeld platziert werden. Der Spieler kann von dieser Stelle aus sofort losspielen.

Master

Das Grundkonzept des *Master* sah eine Trennung der Repräsentation des Spielers als Cursor (Master) und des Avatars (Slave) vor. Der Avatar sollte

Abbildung 8.5: Im *Classic*-Konzept wurde der 2D-Cursor zu einem 3D-Cursor im Game Space modi ziert. v.l.n.r.: 3D-Cursor im Game Space und der Cursor transformiert zum Tool während das Menü Selektion/Zustand zeigt.

ähnlich einem RTS-Game Anweisungen des Spielers ausführen. Er sollte dabei als Toy konzipiert sein und als Feedback auf die Aktionen des Spielers reagieren oder in einem passiveren Modus die Aktionen des Spielers durch Animationen kommentieren. Das Menü sollte durch den Avatar oder ein eigenständiges 2D-Menü erreicht werden.

Dieses Konzept wurde in der Anfangsphase verworfen. Durch eine getrennte Repräsentation des Spielers und dem Avatar im gleichen Raum gäbe es einen Kon ikt bei der Bindung zum Avatar. Die Darstellung des Wechsels zwischen *Play* und *Create* wäre nicht plausibel zumal der Spieler dem Avatar Befehle erteilt anstatt diesen als sich selbst zu sehen. Zusätzlich wurden für diesen Modus E zienzprobleme bei den Laufwegen des Avatars und dem unmittelbaren Feedback auf Aktionen des Spielers erkannt. Das Konzept sollte sich für einen *Integrierten In-Game Editor* eignen. Es wurde für einen Editor des *Play, Create and Shares Genres* als zu wenig e zient eingestuft.

Avatar

Das *Avatar*-Konzept wurde zunächst als Kontrast zum *Classic*-Konzept entwickelt. Die Steuerung erfolgt wie im *Play*-Part direkt über den Avatar. Die Kamera-Ansicht wechselt im Gegensatz zum *Play*-Part in die Vogelperspektive um eine bessere Übersicht zu erhalten. Zunächst musste eine Anpassung der Boundaries vorgenommen werden um die Bewegungsfreiheit des Avatars zu erhöhen. Der Spieler darf im *Create*-Part auch Felder betreten die noch keine Boden-Objekte enthalten. Diese Veränderung ähnelt dem Kreativ-Modus von *LBP 2* und vermeidet den kompletten Verzicht auf den Avatar wie in *ModNation Racers*. Hindernisse wie Wände bleiben für den Spieler bei der Bewegung des Avatars erhalten. Zur Kommunikation des Zustandes (*Play* oder *Create*) wird die Bekleidung des Avatars verwendet. Zur schnelleren Bewegung wurde zusätzlich die Re ex-Fähigkeit des Avatars

verwendet. Hierbei verlangsamt nicht wie im *Play*-Part die Zeit im Spiel sondern die Bewegungsgeschwindigkeit des Avatars erhöht sich. Dadurch konnte die benötigte Zeit zur Bewegung von grö eren Distanzen minimiert werden.

Die gro en Herausforderungen stellten sich durch die Selektion und Platzierung von Objekten als auch durch die Darstellung des Menüs. Das Problem bei der Selektion und Platzierung lag in der Vereinigung der Repräsentation von Spieler und Tool. Die Selektion war an die beschränkten Bewegungsfähigkeiten des Avatars gebunden. Dies führte zur langwierigen Platzierung von Objekten, unvorhersehbaren Selektierungen und Spezialfällen bei z. B. Mehrfeld-Objekten wie Räumen. Die Platzierung und Selektion dieser hätte ein wechselndes Paradigma im Ablauf der Bedienung erfordert.

Auf Grund dieser Umstände wurde wie in *LBP 2* eine Trennung zwischen Avatar und Tool vorgenommen. Als ästhetisches Element und Repräsentation des Tools wurde der magische Ring des Avatars aufgegriffen. Ein Teil von diesem kann sich vom Avatar loslösen und übernimmt die Funktion eines Cursors (siehe Abb. 8.6). Der Cursor kann die Position der Kamera innerhalb des Sichtbereiches des Avatars beein ussen. Parallel kann der Avatar bewegt werden. Dadurch konnte der sichtbare Bereich des Spielers für die Platzierung von Objekten erhöht werden. Ähnlich wie in *LBP 2* wurde eine Verbindung zwischen dem Hauptring am Avatar und dem freien Ring hergestellt um die für die Immersion wichtige Verbindung darzustellen und nicht den Bezugspunkt zu verlieren.

Der Avatar fungiert bei aktiviertem Tool zugleich als Interface-Element zur Deaktivierung des aktuellen Tools und zum Löschen eines aufgehobenen Objektes. Das Menü wird durch eine Aktionstaste aktiviert. Der Ring zieht den Avatar in die Luft. Bei dieser Abbildung des Menüs kann im Gegensatz zu den Versuchen als Abbildung mit Spielkarten oder direkt am Spieler (siehe Abb. 8.3), die Übersicht über das Spielfeld bewahrt werden. Gleichzeitig kann die Vorschau des selektierten Objektes erkannt werden.

Zum Wechsel in den *Play*-Modus wurden verschiedene Abläufe z. B. Auswahl *Play*, Bewegung zu einem validen Punkt und automatisches *Play*, versucht. Nachdem der Avatar im *Create*-Part die Grenzen der Boden-Objekte verlassen darf musste ein Weg gefunden dem Spieler die Option zu spielen zu signalisieren. Die getroffene Lösung signalisiert dem Spieler die Möglichkeit in den *Play*-Modus zu wechseln über die Helmfarbe des Avatars. Die Aktivierung erfolgt anschlie end jederzeit durch eine Aktionstaste.

8.4 Ergebnisse

Die folgenden Abschnitte beschreiben die Erkenntnisse aus den Konzepten *Classic* und *Avatar* in Bezug auf die Repräsentationen von Spieler, Tools, Progress und Status. Die Erkenntnisse wurden aus der Konzeption und durch den Vergleich der Konzepte gewonnen.

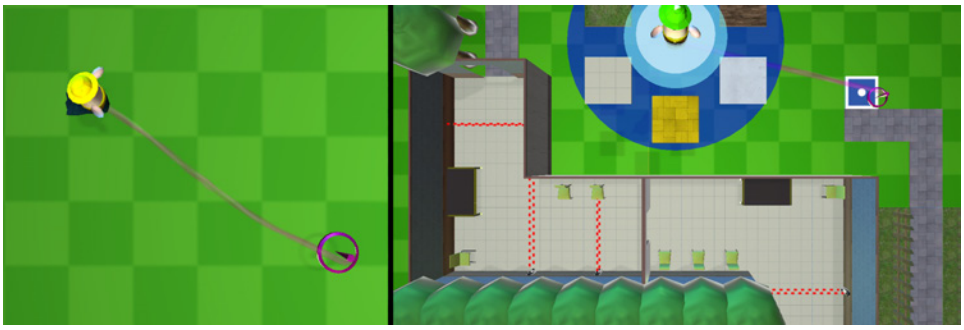


Abbildung 8.6: Im *Avatar*-Konzept übernimmt die Repräsentation des Spielers der Avatar aus dem *Play*-Part. v.l.n.r.: Repräsentation des Spielers beim Bauen durch Trennung zwischen Avatar/Tool und beim Aufruf des Menüs zieht der Ring den Avatar Richtung Kamera.

8.4.1 Repräsentation des Spielers

Die zwei umgesetzten Konzepte *Classic* und *Avatar* und deren Repräsentation des Spielers unterschieden sich in ihrer Grundform voneinander. Im *Classic*-Konzept war der Spieler im Game Space nicht präsent. Im *Avatar*-Konzept wurde die Repräsentation des Spielers direkt aus dem *Play*-Part übernommen. Beide stellten zunächst eine nicht getrennte Abbildung des Spielers und des Tools dar.

Der ästhetisch etablierte Schauplatz ermöglichte es den Cursor aus dem *Classic*-Konzept zumindest im Game Space abzubilden. Damit unterscheidet sich der Cursor von dem von z. B. *EAs Create*, *inFAMOUS 2* oder *ModNation Racers*. Diese Repräsentation ermöglichte es mit den Objekten auf dem Spielfeld direkt zu interagieren. Die Interaktion konnte durch Animationen unterstützt werden und dadurch ebenfalls unterhaltsamer gestaltet werden. Die Abbildung des Spielers kann in dieser Form als Toy bezeichnet werden. Die Navigation geschieht durch ein Festhalten des Spielfeldes und verdeutlicht ebenfalls den Ein uss des Spielers.

Im Gegensatz zum *Classic*-Konzept ist die Navigation im *Avatar*-Konzept durch die Limitation des Avatars weniger e zient. Der Spieler bleibt allerdings Teil der Game World. Die Steuerung ist bereits aus dem *Play*-Part bekannt. Mit Hilfe des Avatars kann der Level wie z. B. in *LBP 2* bereits beim Bauen auf Laufwege und Distanzen geprüft werden ohne explizit in den *Play*-Modus wechseln zu müssen.

Wie auch die Ergebnisse in Kap. 7 feststellen kann die Repräsentation des Spielers nicht direkt aus dem *Play*-Part übernommen werden. Beispielsweise kann der Avatar auch auf nicht bebauten Feldern navigieren um den Spieler bei der ohnehin weniger e zienten Navigation nicht einzuschränken. Die markantesten Kompromisse entstanden im Zusammenhang mit der Repräsentation des Tools und werden im folgenden Abschnitt beschrieben.

8.4.2 Repräsentation von Tools

Die Repräsentation der Tools im *Classic*- und *Avatar*-Konzept bestätigen die direkte Abhängigkeit zwischen Tool und der Repräsentation des Spielers. In keinem von beiden Konzepten konnte das Tool direkt am Spieler abgebildet werden. Im *Classic*-Konzept musste die Selektion der Objekte in das HUD zurückversetzt werden um die Übersicht und Anzeige der notwendigen Informationen zu gewährleisten. Um den Immersionsgrad nicht zu beinträchtigen wurde der 3D-Cursor ebenfalls im HUD abgebildet. Die Animationen des Cursors dienen zusätzlich als Feedback auf die Aktionen eines Tools. Die Darstellung dieser wird neben dem Cursor durch eine 3D-Vorschau des aktuellen Objektes ergänzt. Im Lösch-Modus übernimmt die 3D-Hand selbst die Darstellung des Tools.

Für den Avatar mussten die Kompromisse weitaus grö er ausfallen. Dies lag am Avatar selbst und dessen eingegrenzter Bewegungsfähigkeit. Eine genauere Selektion oder Platzierung von Objekten konnte in dieser Form nicht realisiert werden. Zur Erhöhung der Bewegungsfreiheit wurden die Boundaries angepasst und damit eine vereinfachte Variante des Kreativ-Modus wie er in *LBP 2* verwendet wird eingeführt. Das Problem der ungenauen Selektion blieb hierbei bestehen.

Das aktuelle Tool oder Objekt wurde auf Grund der Grö enverhältnisse nicht direkt am Avatar abgebildet. Die Repräsentation des Spielers und die Repräsentation des Tools wurde durch den am Avatar bereits ästhetisch etablierten magischen Ring getrennt. Wie in *LBP 2* wurde zur Orientierung und Darstellung der Verbindung zwischen Tool und Spieler eine Linie benutzt. Die Präzision der Selektion und Selektionsgeschwindigkeit im Sichtbereich des Avatars konnte dadurch gesteigert werden und zugleich der Ring durch seine Animationen und Effekte als Toy etabliert werden.

Die Abhängigkeit des Tools im Bezug zur Repräsentation des Spielers und damit zum jeweiligen Genre zeichnete sich insbesondere beim *Avatar*-Konzept ab. Bei einer Realisierung über das *Classic*-Konzept wurde die Usability mit weniger Produktionsauswand gewährleistet. Im Vergleich dazu ist der Immsersionsgrad durch den Avatar gegenbüber der 3D-Hand höher einzustufen, da sich der Spieler direkt am Spielfeld repräsentiert fühlt. Es besteht beim *Avatar*-Konzept vor allem das Problem wie die E ciency auf einem zufriedenstellenden Level gehalten werden kann ohne sich durch Kompromisse vom Avatar weiter zu entfernen. Wird die Entscheidung nicht aus ökonomischen sondern aus designtechnischen Gründen getroffen wäre die Verwendung von *User Studies* zur weiteren Entscheidungsfindung und Verbesserung zu empfehlen.

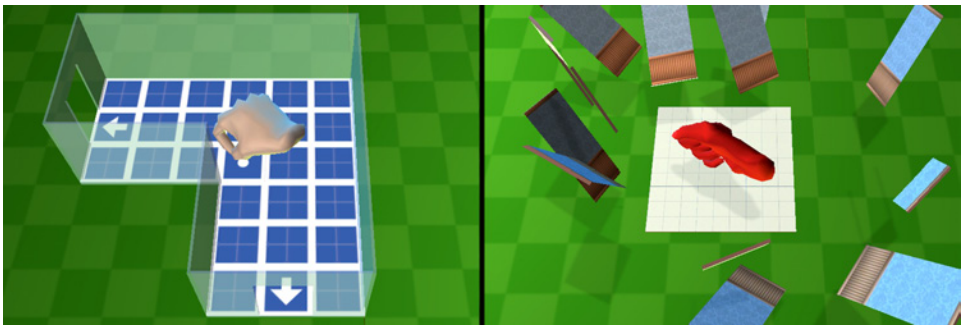


Abbildung 8.7: Der Status wurde durch Kameraeinstellungen, Partikeleffekte, Animationen und Darstellung der Spielobjekte kommuniziert. v.l.n.r.: 2D-/3D-Vorschau bei der Platzierung und Animation beim Zerstören eines Objektes.

8.4.3 Repräsentation von Progress und Status

Zur Darstellung von Progress und Status im In-Game Editor wurden primär die Kameraeinstellung, Partikeleffekte, Animationen, Soundeffekte und Darstellung der Spielobjekte verwendet (siehe Abb. 8.7). Das *Classic*-Konzept kann neben der Repräsentation des Spielers zusätzlich die aktuelle Darstellung des Menüs im HUD nutzen um den Zustand von z. B. dem aktuellen Tool anzuzeigen. Beispielsweise verändert sich das Menü wenn ein Objekt aufgehoben wurde. Die Hand selbst verändert die Farbe und Geste wenn der Lösch-Modus aktiviert wurde. Bei der Darstellung des Avatars wurde versucht den Modus über die Kleidung und Farbe des Helmes zu kommunizieren.

Animationen und Partikeleffekte wurden benutzt um Aktionen wie das Platzieren und Aufheben von Objekten am Spielfeld zu verdeutlichen. Nach Möglichkeit sollten Avatar und 3D-Hand auch entsprechend auf die Aktionen reagieren. Zur Vermeidung von Fehlern beim Platzieren wurde eine dem Raster angepasste 2D-Vorschau mit entsprechender farbiger Kennzeichnung benutzt. Zusätzlich wurde das aktuelle Objekt als 3D-Vorschau-Objekt angezeigt. Die Vorschau plausibel durch das ästhetische Gesamtkonzept zu erklären wurde als zweitrangig eingestuft. Die Vermeidung des Fehlers und Kommunikation zum Spieler war erstrangig.

Der Versuch Fehler oder Nachrichten im *Avatar*-Konzept darzustellen zeigte dass die zur Darstellung von Progress und Status notwendigen Elemente in Abhängigkeit der Repräsentation von Spieler und Tool stehen. Sofern keine HUD-Elemente vorhanden sind müssen kreative Wege gefunden werden diese im Game Space oder innerhalb der Game World darzustellen. Sind keine entsprechenden Wege verfügbar, sollte der Kompromiss mit der Darstellung im HUD eingegangen werden.

8.5 Zusammenfassung

Little Long Finger ist ein Game mit In-Game Editor des *Play, Create und Share Genres*. Es wurde versucht die aus der Analyse erlangten Erkenntnisse auf zwei unterschiedliche Grundkonzepte *Classic* und *Avatar* anzuwenden. Die Realisierung des Editors beschränkte sich auf dessen Grundfunktionen. Es zeigte sich vor allem dass zur Präsentation des Editor-Status Effekte wie Partikel, Animationen und Sounds als Feedback auf Aktionen, sehr einfach aus anderen In-Game Editoren übernommen werden können. Für die Repräsentation von Spieler und Tools konnten insbesondere Interface-Konzepte und Lösungen aus Games mit ähnlichem *Play*-Part aufgegriffen werden. Dieser Umstand verdeutlicht die starke Abhängigkeit und damit Ähnlichkeit und Handlungsspielraum der Repräsentationen zwischen den Genres. Kompromisse, die gegen den Unterhaltungswert für bessere E ciency getroffen werden, fallen je *Play*-Genre ähnlich aus.

Kapitel 9

Zusammenfassung und Ausblick

Programme, Strukturen und UGC im Kontext der Games entwickelten sich von extern erstellten *Hacks* und mit externen Editoren erstellten *Mods* hin zu massentauglichen In-Game Editoren. Diese sind Teil des Gameplays oder dienen als Ergänzung des Games zur Verlängerung der Spielzeit. Games bieten dem Entwickler durch deren grundlegende Bestandteile *Mechanics*, *Aesthetics*, *Story* und *Technology* im Vergleich zu Anwendungssoftware eine Reihe von Möglichkeiten den Spieler während der Bedienung des Editors zu unterhalten. Dadurch kann einer höheren Anzahl an Spielern das Erstellen neuer Inhalte schmackhaft gemacht werden. Der Kern dieser Arbeit beschäftigte sich mit der Frage nach den Spieler unterstützenden und unterhaltsamen Ma nahmen im User Interface von In-Game Editoren die durch Entwickler ergriffen werden können.

9.1 Zusammenfassung

Basierend auf einer für diese Arbeit vorgenommenen Einteilung von UGC im Kontext der Games wurden zur Beantwortung der Fragestellung drei Games (*Little Big Planet 2*, *ModNation Racers* und *inFAMOUS2*) mit optional verwendbarem In-Game Editor des so bezeichneten *Play, Create und Share* Genres ausgewählt. Als Ergänzung wurde *Create* herangezogen, bei dem das Editieren und Erstellen zum Erreichen der Spielziele erforderlich ist. Vier Teilbereiche des User Interfaces eines Games konnten identifiziert werden die durch *Mechanics* und *Aesthetics* zur Steigerung der Unterstützung und Unterhaltung des Spielers beitragen können.

Die *Repräsentation des Spielers* bietet die ein ussreichste und die weitreichendste Möglichkeit des User Interfaces den Spieler zu unterhalten und zugleich zu unterstützen. Von ihr hängt der Grad der Immersion des Spielers ab. Sie bestimmt die Position des Spielers in der Game World und die

Fähigkeit des Spielers im Editor e zient zu navigieren und die Steuerung zu lernen. Durch die Konzeption als Toy, einem einzelnen Bestandteil des Games dessen Verwendung von selbst Spa machen sollte, hängt von dieser mitunter der entscheidende Impuls zur Verwendung des Editors ab. Die Abbildung sollte sich dabei am eigentlichen *Play*-Part orientieren. Die Repräsentation des Spielers ist damit zugleich abhängig vom Genre des Games. Dadurch entstehen vergleichbare Problemstellungen zwischen diesen. Die Lösungsansätze lassen sich auf Games des selben Genres anwenden sind jedoch zwischen Games verschiedenen Genres nicht universell übertragbar.

Die *Repräsentation der Tools* beschreibt das User Interface Element zur Erstellung und Bearbeitung der Game World. Wie das aktuelle Tool im Game dargestellt wird und durch den Spieler verwendet werden kann hat im wesentlichen Ein uss auf die *E ciency* der durchgeführten Aktionen. Durch die Verknüpfung des Tools mit der Repräsentation des Spielers ist die Repräsentation des Tools direkt abhängig von dieser und damit indirekt vom Genre. Der Vorteil einer Realisierung als Toy muss im Verhältnis zu den eventuell daraus entstehenden Nachteilen stehen wie z. B. verlängerter Antwortzeiten auf Aktionen, die Verlängerung der Zeit eine Aufgabe abzuschlie en oder einer erhöhten Fehleranfälligkeit bei Aktionen des Spielers.

Die *Repräsentation des UGC* ist verantwortlich für die Darstellung der UGC-Funktionalitäten und der eigenen sowie fremden Inhalte im Game. Der Spieler soll möglichst schnell Zugriff auf erstellte Inhalte haben. Die Repräsentation von UGC kann als Anreiz für den Spieler genutzt werden sich mit UGC bzw. dem Editor zu befassen. Die Wahl der Umsetzung und Präsentation von UGC soll das visuelle und auditive Gesamtkonzept der Game World des Play-Parts unterstützen.

Die *Repräsentation von Progress und Status* beschreibt die Kommunikation von Modus des Editors, verfügbarer Optionen, Fehlern oder ausgeführten Aktionen des Spielers. Die Umsetzung sollte dabei in Abhängigkeit der ästhetischen Vorgaben konzipiert werden. Insbesondere für die Usability kritische Informationen wie beispielsweise der Hinweis fehlerhafter Platzierungen sollten auf die Verwendung von zusätzlichen Effekten verzichten. Im Gegensatz dazu kann das Feedback auf Aktionen des Spielers die gesamte Palette an Effekten wie Partikel, Animationen oder Sounds nutzen. Das Feedback lässt sich universell auf andere Genres übertragen.

Im Diplomprojekt *Little Long Finger* wurden die Erkenntnisse aus der Analyse zur Umsetzung von zwei Editor-Konzepten *Classic* und *Avatar* verwendet. Die daraus gewonnenen Ergebnisse sind als Teil in das Gesamtergebnis einge ossen. Wie sich vor allem zeigte, sind entgegen der Möglichkeiten bei der Darstellung von Feedback auf Aktionen, die Repräsentation des Spielers und Tools nicht von Genre zu Genre universell übertragbar. Es lassen sich aber Gemeinsamkeiten in den Repräsentation zwischen Games des selben Genres feststellen.

9.2 Ausblick

Im Gesamtbild eines Games ist das User Interface des In-Game Editors nur ein Teilaspekt der für Spieler motivierenden Faktoren zur Erstellung von UGC. Eine ausführliche Ergründung der Thematik in der Gesamtheit bleibt für die Zukunft noch offen. Beispielweise sind die Fragen nach dem Ein uss der verfügbaren Funktionen im Editor und der Ebene der Content Creation im Game oder dem Beitrag des *Play*-Parts selbst für die Erstellung von UGC in Spielen durch In-Game Editoren noch unbeantwortet. Die Aufstellung von Leitfäden je Genre zur Konzeption von In-Game Editoren wäre ein nächster logischer Schritt.

Externer UGC im Kontext der Games wird bereits ausführlicher untersucht. Zum Beispiel behandelt Behr aktuell in [2] die Entwicklung von Mods aus aneignungstheoretischer Sicht. Bei Mods sind die qualitativen Unterschiede zwischen Game und Mod zusehends schwerer auszumachen. Die Popularität von Mods steigt weiter an. Sie sind ein fester Bestandteil von vielen Computer Games.

UGC in Games und damit verbunden auch In-Game Editoren werden für Spieler und Entwickler zugleich zunehmend wichtiger. Sie ermöglichen auch Spielern auf Konsolen oder mobilen Devices und Spielern ohne umfangreichem technischen Know-How das Erstellen und Verteilen von selbst erstellten Inhalten. Abseids des *Play, Create und Share Genres* zeigen Sandbox Games wie *Minecraft* dass Spieler die Freiheit schätzen kreative Inhalte zu erstellen. Daraus entstehen dem Entwickler Vorteile wie eine Verlängerung der Produktlebenszeit oder neuer kreativer Input von den Spielern.

Die Grenze zwischen dem Spielen und dem Modden bzw. dem Spieler und dem Entwickler verschwimmt zunehmend. Der Fokus der Spielproduktion liegt verstärkt auf zusätzlichen, modernen, gebrauchstauglicheren, externen Modding-Tools wie z. B. der *CryENGINE 3 Sandbox* von Crytek, die den Spieler dazu ermutigen selbst das Design in die Hand zu nehmen und spielend neue Inhalte zu erstellen. In-Game Editoren werden in Zukunft als Feature eines Spieles erwartet. Aktuelle Spiele wie z. B. *Portal 2*[1] (2011, Valve Software) werden mit In-Game Editoren ergänzt. Die Zukunft wird zeigen welche Faktoren dazu beitragen eine breite Masse an Spielern für die Erstellung von UGC mit In-Game Editoren zu begeistern. Zusätzlich unterhaltsame Ma - nahmen zu den unterstützenden Ma nahmen sind wie insbesondere *Little Big Planet 2* zeigt ein erster Schritt hin zu interessanteren In-Game Editoren. Ob und wie dieser Trend weitergeführt wird bleibt offen. Das Rennen der Entwickler nach neuen Konzepten für In-Game Editoren hat begonnen.

[1] http://www.thinkwithportals.com

Anhang A

Inhalt der DVD

Format: DVD-R, Single Layer, UDF/ISO 13346-Format

A.1 Diplomarbeit

Pfad: /

Pippan_Alexander_2012.pdf Diplomarbeit

Pfad: /Quellen

articles/	Zeitungs- oder Zeitschriftenartikel
books/	Einbändige Bücher
incollection/	Beiträge als Teil von Büchern
inproceedings/	Artikel aus Konferenzberichten
manuals/	Technische Dokumentationen
masterthesis/	Diplom-, Magister- oder andere Abschlussarbeiten
online/	Online-Quellen
phd/	Doktor- oder andere Promotionsarbeiten
techreports/	Berichte von Hochschulen und Institutionen
unpublished/	Nicht formell veröffentlichte Dokumente

A.2 Diplomprojekt Little Long Finger

Pfad: /Diplomprojekt/Doc

Game_Design_Document.pdf Game Design Document

Pfad: /Diplomprojekt/Media

/screenshots	Screenshots von *Little Long Finger*

Pfad: /Diplomprojekt/Unity Unity Projektordner

Pfad: /Diplomprojekt/Builds

Little_Long_Finger_32.exe Windows (32bit)
Little_Long_Finger_64.exe Windows (64bit)
Little_Long_Finger_Mac.app MacOS X (Universal)

Quellenverzeichnis

Literatur

[1] Inc. Apple Computer. *ResEdit Reference*. Kopie auf DVD (Datei Quellen / manuals / applecomputer1995 / applecomputer1995 . pdf). Apple Computer, Inc. 1995.

[2] Katharina Maria Behr. *Kreativer Umgang mit Computerspielen - Die Entwicklung von Spielmodi kationen aus aneignungstheoretischer Sicht*. Verlag Werner Hülsbusch, 2010.

[3] Heather Desurvire, Martin Caplan und Jozsef A. Toth. Using heuristics to evaluate the playability of games . In: *CHI 04 extended abstracts on Human factors in computing systems*. CHI EA 04. Vienna, Austria: ACM, 2004, S. 1509 1512.

[4] Heather Desurvire und Charlotte Wiberg. Game Usability Heuristics (PLAY) for Evaluating and Designing Better Games: The Next Iteration . In: *Proceedings of the 3d International Conference on Online Communities and Social Computing: Held as Part of HCI International 2009*. OCSC 09. San Diego, CA: Springer-Verlag, 2009, S. 557 566.

[5] Alan Dix u. a. *Human Computer Interaction*. 3. Au . Pearson Education Limited, 2004.

[6] Erik Fagerholt und Magnus Lorentzon. Beyond the HUD User Interfaces for Increased Player Immersion in FPS Games . Magisterarb. Department of Computer Science und Engineering Division of Interaction Design Chalmers University of Technology Göteborg, Sweden, 2009.

[7] Melissa A. Federoff. Heuristics and Usability Guidelines for the Creation and Evaluation of Fun in Video Games . Magisterarb. Department of Telecommunications of Indiana University, Dez. 2002.

[8] Marc Hassenzahl, Michael Burmester und Franz Koller. Der User Experience (UX) auf der Spur: Zum Einsatz von www.attrakdiff.de . In: *Usability Professionals 2008*. 2008, S. 78 82. URL: http : / / www . attrakdi .de/ les/up08_ux_auf_der_spur.pdf.

[9] Julian Kücklich. Precarious Playbour: Modders and the Digital Games Industry The History of Modding The Economy of Modding . In: *The Fiberculture Journal* 5 (2005).

[10] Raph Koster. *A Theory of Fun for Game Design*. Paraglyph Press, Inc., 2005.

[11] Greg Lastowka. User-Generated Content & Virtual Worlds . In: *Vanderbilt Journal of Entertainment and Technology Law, Forthcoming* 10 (2007), S. 893 912.

[12] Steven Levy. *Hackers - Heroes of the Computer Revolution*. O Reilly Media Inc., 2010.

[13] Thomas W. Malone. Heuristics for designing enjoyable user interfaces: Lessons from computer games . In: *Proceedings of the 1982 conference on Human factors in computing systems*. CHI 82. Gaithersburg, Maryland, United States: ACM, 1982, S. 63 68.

[14] David B. Nieborg. *Am I Mod or Not? - An analysis of First Person Shooter modi cation culture*. Techn. Ber. Institute of Media & Re/Presentation Utrecht University, 2005.

[15] OECD. *Participative Web: User-created Content*. Techn. Ber. Directorate for Science, Technology, Industry Committee for Information, Computer und Communications Policy, Apr. 2007. URL: http://www.oecd.org/dataoecd/57/14/38393115.pdf.

[16] Olli. FCJ-109 Play, Create, Share? Console Gaming, Player Production and Agency . In: *The Fibreculture Journal* 16 (2010).

[17] Cory Ondrejka. *Escaping the Gilded Cage: User Created Content and Building the Metaverse*. Techn. Ber. University of Southern California - Annenberg School for Communication, Mai 2004.

[18] Walt Scacchi. *Computer Game Mods, Modders, Modding, and the Mod Scene*. Techn. Ber. Institute for Software Research, Mai 2010. URL: http://www.ics.uci.edu/~wscacchi/GameLab/ModSquad-Scacchi.pdf.

[19] Noah Schaffer. *Heuristics for Usability in Games*. Kopie auf DVD (Datei Quellen/manuals/scha er2007/scha er2007.pdf). Apr. 2007.

[20] Jesse Schell. *The Art of Game Design - A Book of Lenses*. Morgan Kaufmann, 2008.

[21] Tanja Sihvonen. *Players Unleashed! Modding The Sims and the Culture of Gaming*. Techn. Ber. Annales Universitatis Turkuensis Universität Turku, 2009.

[22] Olli Sotamaa. Computer Game Modding, Intermediality and Participatory Culture . Kopie auf DVD (Datei Quellen/manuals/sotamaa/sotamaa.pdf).

[23] Daniel Volk. Co-creative Game Development in a Participatory Metaverse . In: *Proceedings of the Tenth Anniversary Conference on Participatory Design 2008*. 2008, S. 264 265.

[24] Daniel Volk. Game development 2.0 . In: *Proceedings of the 2007 conference on Future Play*. 2007, S. 225 228.

[25] Andreas Schwill Volker Claus. *Duden. Informatik. Ein Fachlexikon für Studium und Praxis*. Bibliographisches Institut, Mannheim, 2003.

Online-Quellen

[26] URL: http://www.abandonia.com/ les/games/390/Castle Wolfenstein_ 2.gif.

[27] URL: http : / / www . video - games - museum . com / en / game / Castle - Smurfenstein/37/3/12293.

[28] URL: http : / / pcmedia . ign . com / pc / image / unrealtournament2004 _ 031204_008.jpg.

[29] URL: http : / / www . redorchestragame . com / images / 08 - 02 - 06 / full / shot0107.jpg.

[30] URL: http://www.thereeltodd.com/img44x0123/LodeRunner_editor_ key.gif.

[31] URL: http://www.udk.com/elements/img/galleries/features-editing-hero.jpg.

[32] URL: http://mycryengine.com/assets/html/gfx/lightbox/visuals_01_ NLDSS.jpg.

[33] URL: http://mycryengine.com/assets/html/gfx/lightbox/sandbox_02_ FG.jpg.

[34] URL: http://www.pixologic01.com/zbrush/features/zbrush4r2/gallery/ Nicolas_Garilhe_03.jpg.

[35] URL: http : / / upload . wikimedia . org / wikipedia / en / 7 / 7d / Utopia _ (Intellivision).png.

[36] URL: http://www.abandonia.com/ les/games/393/SimCity Classic_4. png.

[37] URL: http://www.gamesaktuell.de/screenshots/original/2008/10/little_ big_planet_editor_54_.jpg.

[38] Ernest Adams. *The Designer s Notebook: Sandbox Storytelling*. Kopie auf DVD (Datei Quellen/online/adams2010/adams2010.pdf). Gamasutra UBM TechWeb. Aug. 2010. URL: http://www.gamasutra.com/view/ feature/6039/the_designers_notebook_sandbox_.php (besucht am 05. 08.2011).

[39] Leigh Alexander. *City Of Heroes s Mission Architect Births 20,000 Arcs In First Week*. Kopie auf DVD (Datei Quellen/online/alexander2009/alexander2009a.pdf). Gamasutra UBM TechWeb. Apr. 2009. URL: http://gamasutra.com/view/news/23260/City_Of_Heroess_Mission_Architect_Births_20000_Arcs_In_First_Week.php (besucht am 18.06.2011).

[40] Leigh Alexander. *EA Bright Light Unveils New User-Generated Content IP*. Kopie auf DVD (Datei Quellen / online / alexander2010 / alexander2010.pdf). Gamasutra UBM TechWeb. Aug. 2010. URL: http://gamasutra.com/view/news/29703/EA_Bright_Light_Unveils_New_UserGenerated_Content_IP.php (besucht am 17.06.2011).

[41] Leigh Alexander. *Interview: Making User-Generated Content Friendly But Deep With ModNation Racers*. Kopie auf DVD (Datei Quellen/online/alexander2010/alexander2010a.pdf). Gamasutra UBM TechWeb. Juni 2010. URL: http://gamasutra.com/view/news/28787/Interview_Making_UserGenerated_Content_Friendly_But_Deep_With_ModNation_Racers.php (besucht am 25.08.2011).

[42] Marcus Andrews. *Game UI Discoveries: What Players Want*. Kopie auf DVD (Datei Quellen/online/andrews2010/andrews2010.pdf). Gamasutra UBM TechWeb. Feb. 2010. URL: http://www.gamasutra.com/view/feature/4286/game_ui_discoveries_what_players_.php (besucht am 19.10.2011).

[43] Matt Barton und Bill Loguidice. *The History of Elite: Space, the Endless Frontier*. Kopie auf DVD (Datei Quellen/online/mattbarton2009/mattbarton2009.pdf). Gamasutra UBM TechWeb. Apr. 2009. URL: http://www.gamasutra.com/view/feature/3983/the_history_of_elite_space_the_.php (besucht am 05.08.2011).

[44] Steve Breslin. *The History and Theory of Sandbox Gameplay*. Kopie auf DVD (Datei Quellen/online/breslin2009/breslin2009.pdf). Gamasutra UBM TechWeb. Juli 2009. URL: http://www.gamasutra.com/view/feature/4081/the_history_and_theory_of_sandbox_.php (besucht am 27.04.2011).

[45] Simon Carless. *Opinion: Why LittleBigPlanet Is Web 2.0 For Games, Ful lled*. Kopie auf DVD (Datei Quellen/online/carless2008/carless2008.pdf). Gamasutra UBM TechWeb. Okt. 2008. URL: http://gamasutra.com/view/news/20561/Opinion_Why_LittleBigPlanet_Is_Web_20_For_Games_Ful lled.php (besucht am 19.06.2011).

[46] Frank Cifaldi. *D.I.C.E: Will Wright & The YMC-Spore Team*. Kopie auf DVD (Datei Quellen/online/cifaldi2007/cifaldi2007.pdf). Gamasutra UBM TechWeb. Feb. 2007. URL: http://gamasutra.com/view/news/12700/DICE_Will_Wright__The_YMCSpore_Team.php (besucht am 19.06.2011).

[47] ComputerAndVideoGames. *The complete history of open-world games (part1)*. Kopie auf DVD (Datei Quellen / online / computerandvideogames2008 / computerandvideogames2008 . pdf). ComputerAndVideoGams.com Future Publishing Limited. Mai 2008. URL: http://www.computerandvideogames.com/189591/features/the-complete-history-of-open-world-games-part-1/ (besucht am 05.08. 2011).

[48] ComputerAndVideoGames. *The complete history of open-world games (part2)*. Kopie auf DVD (Datei Quellen / online / computerandvideogames2008a / computerandvideogames2008a . pdf). ComputerAndVideoGames.com Future Publishing Limited. Mai 2008. URL: http://www.computerandvideogames.com/189599/features/the-complete-history-of-open-world-games-part-2/ (besucht am 05.08. 2011).

[49] *CryENGINE 3 FAQ*. Kopie auf DVD (Datei Quellen / online / cryengine3faq/cryengine3faq.pdf). Crytek UK Ltd. URL: http://www.crydev.net/ les/CryENGINE_3_Free_FAQ.pdf (besucht am 18.08. 2011).

[50] Mattthew Frassetti. *Play, Create, and Share Is the New Sandbox Game*. Kopie auf DVD (Datei Quellen/online/frassetti2009/frassetti2009.pdf). Gematsu. März 2009. URL: http://gematsu.com/2009/06/play-create-and-share-is-the-new-sandbox-genre (besucht am 02.08.2011).

[51] *GDCE 2010 - Interview mit Sean Tracy über den Cryengine-3 Editor Sandbox*. Aug. 2010. URL: http://www.youtube.com/watch?v=_4SbDkyLyBI (besucht am 18.08.2011).

[52] Kris Graft. *ModNation Racers PSP Dev: It s All In The Tools*. Kopie auf DVD (Datei Quellen/online/graft2010/graft2010.pdf). Gamasutra UBM TechWeb. Apr. 2010. URL: http://gamasutra.com/view/news/28099/ModNation_Racers_PSP_Dev_Its_All_In_The_Tools.php (besucht am 18.06.2011).

[53] Elizabeth Harper. *GDC09: Mission Architect in-depth*. Kopie auf DVD (Datei Quellen/online/harper2009/harper2009.pdf). Joystiq.com AOL Inc. Apr. 2009. URL: http://massively.joystiq.com/2009/04/06/gdc09-mission-architect-in-depth/ (besucht am 18.06.2011).

[54] John Harris. *Game Design Essentials: 20 Open World Games*. Kopie auf DVD (Datei Quellen/online/harris2007/harris2007.pdf). Gamasutra UBM TechWeb. Sep. 2007. URL: http://www.gamasutra.com/view/feature/1902/game_design_essentials_20_open_.php (besucht am 05. 08.2011).

[55] *inFAMOUS 2 - The O cial Site*. Kopie auf DVD (Datei Quellen / online/infamous2/infamous2.pdf). Sony Computer Entertainment Europe Limited. URL: http://www.infamousthegame.com/de_DE/infamous2.html (besucht am 14.10.2011).

[56] Stephen Jacobs und Christ Remo. *The Anti-Auteurs: Developers Speak On User-Generated Content*. Kopie auf DVD (Datei Quellen/online/jacobs2008/jacobs2008.pdf). Gamasutra UBM TechWeb. Aug. 2008. URL: http://gamasutra.com/view/news/19846/The_AntiAuteurs_Developers_Speak_On_UserGenerated_Content.php (besucht am 19.06.2011).

[57] *K-Glider (CryEngine 3 Flowgraph Prototype)*. Juni 2011. URL: www.youtube.com/watch?v=JSK2oQOxeRM (besucht am 18.08.2011).

[58] Tom Kim. *In Depth: Media Molecule On LBP s Genesis And Future*. Kopie auf DVD (Datei Quellen/online/kim2008/kim2008.pdf). Gamasutra UBM TechWeb. Nov. 2008. URL: http://gamasutra.com/view/news/20974/In_Depth_Media_Molecule_On_LBPs_Genesis_And_Future.php (besucht am 19.06.2011).

[59] Raph Koster. *User created content*. Kopie auf DVD (Datei Quellen/online/koster2006/koster2006.pdf). Juni 2006. URL: http://www.raphkoster.com/2006/06/20/user-created-content/ (besucht am 21.06.2011).

[60] David Kushner. *The Mod Squad*. Kopie auf DVD (Datei Quellen/online/kushner2002/kushner2002.pdf). Feb. 2002. URL: http://www.popsci.com/gear-gadgets/article/2002-07/mod-squad (besucht am 05.05.2011).

[61] Eric Levine. *LittleBigPlanet 2 for PS3 O cially Announced for Winter 2010!* Kopie auf DVD (Datei Quellen/online/levine2010/levine2010.pdf). PlayStation Blog Sony Computer Entertainment America LLC. Mai 2010. URL: http://blog.us.playstation.com/2010/05/10/littlebigplanet-2-for-ps3-o cially-announced-for-winter-2010/#comments (besucht am 18.06.2011).

[62] Alex Moore. *Opinion: UI Is The Game, The Game Is UI*. Kopie auf DVD (Datei Quellen/online/moore2011/moore2011.pdf). Gamasutra UBM TechWeb. Aug. 2011. URL: http://gamasutra.com/view/news/36464/Opinion_UI_Is_The_Game_The_Game_Is_UI.php (besucht am 27.10.2011).

[63] Simon Parkin und Staff Gamasutra. *Develop: Media Molecule On The Power Of Constraints*. Kopie auf DVD (Datei Quellen/online/parkin2008/parkin2008.pdf). Gamasutra UBM TechWeb. Juli 2008. URL: http://gamasutra.com/view/news/19651/Develop_Media_

Molecule_On_The_Power_Of_Constraints.php (besucht am 19.06. 2011).

[64] Mike Rose. *GDC 2011: User-Generated Missions For Infamous 2*. Kopie auf DVD (Datei Quellen/online/rose2011/rose2011.pdf). Gamasutra UBM TechWeb. März 2011. URL: http://gamasutra.com/view/news/33318/GDC_2011_UserGenerated_Missions_For_Infamous_2.php (besucht am 17.06.2011).

[65] Brandon She eld und Staff Gamasutra. *Paris GDC: Media Molecule On Making LittleBigPlanet*. Kopie auf DVD (Datei Quellen/online/she eld2008/she eld2008.pdf). Gamasutra UBM TechWeb. Juni 2008. URL: http://gamasutra.com/view/news/19135/Paris_GDC_Media_Molecule_On_Making_LittleBigPlanet.php (besucht am 19.06.2011).

[66] Aldo Tolino. *Beyond Play: Analyzing Player-Generated Creations*. Kopie auf DVD (Datei Quellen/online/tolino2009/tolino2009.pdf). Gamasutra UBM TechWeb. Mai 2009. URL: http://www.gamasutra.com/view/feature/4008/beyond_play_analyzing_.php (besucht am 18.06. 2011).

[67] Valve. *Polycount Pack*. Kopie auf DVD (Datei Quellen/online/valve/valve.pdf). Valve Corporation. URL: http://wiki.teamfortress.com/wiki/Polycount_Pack (besucht am 06.01.2012).

[68] Valve. *Source Brochure*. Kopie auf DVD (Datei Quellen/online/valve2008/valve2008.pdf). Valve Corporation. 2008. URL: http://source.valvesoftware.com/SourceBrochure.pdf (besucht am 18.08.2011).

[69] Valve. *The Mann-Conomy Update FAQ*. Kopie auf DVD (Datei Quellen/online/valve2010/valve2010.pdf). Valve Corporation. 2010. URL: http://www.teamfortress.com/mannconomy/FAQ (besucht am 02. 08.2011).

[70] James Au Wagner. *Triumph of the Mod - Player-created additions to computer games aren t a gobby anymore they re lifeblood of the industry*. Kopie auf DVD (Datei Quellen/online/wagner2002/wagner2002.pdf). Salon Media Group. Apr. 2002. URL: http://www.salon.com/technology/feature/2002/04/16/modding (besucht am 05.05.2011).

[71] Dan Whitehead. *Born Free: the History of the Openworld Game*. Kopie auf DVD (Datei Quellen/online/whitehead2008/whitehead2008.pdf). Eurogamer. Feb. 2008. URL: http://www.eurogamer.net/articles/born-free-the-history-of-the-openworld-game-article (besucht am 05.08.2011).

[72] Wikipedia. *Diorama*. Kopie auf DVD (Datei Quellen/online/wikipediaa/wikipediaa.pdf). Wikimedia Foundation Inc. URL: http://de.wikipedia.org/wiki/Diorama (besucht am 07.01.2012).

[73] Wikipedia. *Sturgeon s Law*. Kopie auf DVD (Datei Quellen / online / wikipedia / wikipedia.pdf). Wikimedia Foundation Inc. URL: http://en.wikipedia.org/wiki/Sturgeon%27s_Law (besucht am 14.10.2011).

[74] Chris Zimmermann. *inFAMOUS 2: User-Created Missions Give In nite Possibilities, Limited Beta in April*. Kopie auf DVD (Datei Quellen/online/zimmermann2011/zimmermann2011.pdf). PlayStation Blog Sony Computer Entertainment America LLC. März 2011. URL: http://blog.us.playstation.com/2011/03/01/infamous-2-user-created-missions-give-in nite-possibilities-limited-beta-in-april/ (besucht am 17.06.2011).