Hochschule fur Angewandte
Wissenschaften Hamburg
Hamburg University of Applied Sciences

STUDIENGANG **MEDIENTECHNIK**
FAKULTÄT DESIGN, MEDIEN UND INFORMATION
STUDIENDEPARTMENT TECHNIK

Hamburg, den 27. Juli 2006

Diplomarbeit

Das Computerspiel
als nichtlineare Erzählform

Entwicklung und Umsetzung eines dramaturgischen Konzepts

vorgelegt von Jan Müller-Michaelis

Betreuer: Prof. Gunther Rehfeldt
Zweitkorrektor: Prof. Dr. Norbert Witt

Erklärung

Hiermit versichere ich, dass ich die vorliegende Diplomarbeit eingeständig verfasst und keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe. Mit der Ausstellung der Arbeit in der Bibliothek erkläre ich mich nicht einverstanden.

Hamburg, den 27. 06. 2006

_____

Inhaltsverzeichnis

Abkürzungen

AGS	Adventure Game Studio
ARPA	Advanced Research Projects Agency
CAS	Character Animation Set
CD	Compact Disk
CRPG	Computer Roleplaying Game
EGA	Enhanced Graphics Adapter
EP	Erfahrungspunkte
GTGE	Golden T. Game Engine
GUI	Graphical User Interface
HSQL	Hypersonic Structured Query Language
IF	Interactive Fiction
INVO	Inventarobjekt
LARP	Life Action Rolplaying Game
MMORPG	Massively Multiplayer Online Roleplaying Game
MUD	Multiple User Dungeon
NSC	Nichtspielercharakter
ROD	Raumobjektdarstellung
ROI	Raumobjektinteraktion
RPG	Roleplaying game
SC	Spielercharakter
VGA	Video Graphics Array

1 Einleitung

1.1 Handlung in Computerspielen

„Was ist die Handlung des Videospiels Super Mario Bros.?“ – Eine naheliegende Antwort auf diese Frage lautet: „Der Klempner Mario rettet die entführte Prinzessin Toadstool aus den Klauen des Reptilienkönigs Bowser.“ (Super Mario Bros., 1985)

Dieser knappe Dialog enthält gleich mehrere interessante Aussagen über die Dramaturgie von Computer- und Videospielen.

Erstens: Computer- und Videospiele können über einen Plot verfügen.

Zweitens: Der Beitrag des Spielers ist für die Handlung offensichtlich unerheblich. Mario bleibt uns bekannt als der Klempner, der die Prinzessin rettet – unabhängig davon, ob der Spieler nur ein „Game over“ erzielt.

Mit dieser einfachen Betrachtung wird bereits eine häufig diskutierte Frage der Unterhaltungsbranche beantwortet. Narration und Interaktion schliessen sich nicht gegenseitig aus. Mit interaktiven Medien wie Computerspielen lassen sich genauso Geschichten erzählen, wie mit den klassischen Medien Buch und Film. Auch der Spieler kann, wie der Leser eines Romans oder der Zuschauer eines Films, einen Helden auf seine Abenteuer begleiten. Der einzige Unterschied ist, dass es vom Geschick des Spielers abhängt, ob er die gesamte Handlung bis hin zum Ende präsentiert bekommt. Durch einen Misserfolg ändert sich die Handlung des Spiels nicht. Niemand würde den Plot von „Super Mario Bros.“ folgendermaßen zusammenfassen: „Der Klempner Mario macht sich auf den Weg, die Prinzessin zu retten. In Level zwei fällt er allerdings in einen Abgrund und stirbt.“ Bei einem negativen Ausgang des Spiels trennen sich die Wege von Held und Spieler. Bei einem positiven Ausgang hingegen kann sich der Spieler selbst als Held fühlen.

So enttarnt sich im Computerspiel ein grundsätzliches Anliegen der Narration: In jeder Geschichte, egal ob in der Literatur, im Kino oder eben im Spiel, begleitet der Rezipient einen Protagonisten durch die Handlung. Die Probleme, die der Protagonist im Laufe

der Geschichte lösen muss, erlebt der Rezipient als seine eigenen. Und die erfolgreiche Bewältigung des Konfliktes fügt der Rezipient seinem eigenen Erfahrungsschatz hinzu, als hätte er die Situation selbst durchlebt.

Ein Hauptanliegen der Dramaturgie ist demzufolge, eine enge Bindung zwischen dem Protagonisten und dem Rezipienten zu erzielen. Der Rezipient soll sich mit dem Helden identifizieren – mit ihm leiden und sich mit ihm freuen. Dies muss umso mehr der Fall sein, wenn der Zuschauer regelrecht den Platz des Helden einnimmt, direkte Kontrolle über seinen Körper hat und die Entscheidungen, die zum erfolgreichen Ausgang der Geschichte führen, selbst treffen muss. Dieses Maß an Identifizierung kann nur in einem interaktiven Medium wie dem Computerspiel erreicht werden. Zurecht formulieren die Spielehersteller in ihren Produktbeschreibungen „Sie steuern April Ryan“ (The Longest Journey, 2000, Handbuch S. 6), „Sie übernehmen die Rolle von Indiana Jones“ (Indiana Jones and the Fate of Atlantis, 1992, Handbuch S. 2) oder sogar „Sie sind Kyle Katarn“ (Star Wars: Jedi Knight II, 2002, Cover).

Kann es also sein, dass das Computerspiel das bessere Erzählmedium ist? Auf den folgenden Seiten versuche ich, darüber Aufschluss zu geben.

1.2 Zielsetzung und Umfang

Ziel dieser Diplomarbeit ist es, zu untersuchen, inwieweit sich ein Computerspiel als Medium zum Erzählen einer Geschichte eignet. Zu diesem Zweck begleitet die Arbeit die Entstehung des Grafik-Adventures „Edna bricht aus“, das als praktischer Teil dieser Arbeit von mir konzipiert und mit Hilfe eines engen Mitarbeiterkreises in die Tat umgesetzt wurde. Die Diplomarbeit gliedert sich in einen theoretischen und einen praktischen Teil.

Im ersten Teil untersuche ich die Geschichte interaktiver Dramaturgien. Dabei werde ich den Ursprung des erzählenden Computerspiels und insbesondere den des Grafik-Adventures hervorheben. Um den historischen Kontext einzubeziehen, beginne ich damit, dass ich zunächst die theoretischen Grundlagen nicht interaktiver Erzählformen beleuchte.

Der zweite Teil der Diplomarbeit beschäftigt sich mit der Dramaturgie und Umsetzung von „Edna bricht aus“. Hier werden nacheinander die vier Arbeitsschritte begleitet, die schliesslich zu dem fertigen Spiel führten: Dramaturgie, Programmierung, Skripting und Layout.

Abschließend werde ich im Diskussionsteil anhand meiner Ergebnisse eine Einschätzung darüber wagen, inwieweit sich das Computerspiel als Erzählmedium eignet.

Als Zeugnis meiner praktischen Arbeit liegt der Diplomarbeit das Spiel „Edna bricht aus“ in Form einer CD bei. Im Anhang finden sich die Karten und der „Walkthrough“, die sich beim Durchspielen als hilfreich erweisen könnten.

„Edna bricht aus“ baut auf den Ergebnissen eines Hochschulprojekts auf, in dessen Zuge ein spielbares Demo mit Hilfe der Entwicklerumgebung und „Run-Time Engine“ AGS (Adventure Game Studio) entstand. Die spielbare Demo umfasste sechs „Räume“ mit insgesamt fünfzehn interaktiven Objekten (im Folgenden „Raumobjekte“).

Das fertige Spiel „Edna bricht aus“ umfasst etwa 122 Räume und über 2000 Raumobjekte. Damit ist es ein vollwertiges Adventure, dass im Umfang den Produkten der Computerspiele-Industrie entspricht. „Edna bricht aus“ wurde mit Hilfe der öffentlichen Klassenbibliothek GTGE (Golden T. Game Engine) auf Java programmiert. Zusätzlich zu den 29 Programmklassen wurde eigens eine primitive Skriptsprache entwickelt, die uns die Trennung von Programm und Content ermöglichte. Mit Ausnahme von einigen Java-Klassen für das grafische User Interface (GUI) befindet sich der gesamte Inhalt des Spiels, jedes Objekt, jede Animation und jede mögliche Ereignisbehandlung, auf den 20 Tabellen einer HSQL-Datenbank. So ist es möglich, mit dem entstandenen Programm unterschiedlichste Spiele zu erstellen, ohne in den Programmcode einzugreifen. Die einfache Skriptsprache mit ihren lediglich 39 Befehlen erlaubt es auch Laien, Geschichten mit der Eba-Engine („Eba“ steht hier für „Edna bricht aus“) zu erzählen.

1.3 Kooperation

An der Programmierung der Engine arbeiteten neben mir auch Felix Engel, Stefan Hütter und Andreas Endler, alles Studenten aus dem Fachbereich Medientechnik, sowie Olaf Casper.

Das Konzept für die Programmierung entwickelte ich ebenfalls zusammen mit Olaf Casper und Felix Engel.

Bei der Entwicklung des dramaturgischen Konzeptes halfen Felix Engel, Anne Beutel und Finn Seliger aus dem Fachbereich Medientechnik, sowie René Anhaus, Erkan Yilmaz und Daniela Pusch.

Alle Hintergrundgrafiken, Objekte, Charaktere, Animationen und Benutzeroberflächen wurden vom mir selbst entworfen und für das Spiel aufbearbeitet. Inspirationen holte ich mir hierbei in Entwürfen von Jan Bauer. Anne Beutel zeichnete ausserdem die Testgrafiken, die im Rahmen des Demo-Projektes entstanden.

Beim Skripting bekam ich großzügige Unterstützung von René Anhaus und Christina Lange. Beim Testskripting für das Demo half außerdem Christoph Bertram.

Die Hintergrundmusik, Soundeffekte und Sprachausgabe sind Thema der Diplomarbeit von Finn Seliger, die voraussichtlich im folgenden Wintersemester verfasst wird. Einige musikalische Layouts sind testweise bereits in das Spiel integriert.

2 Die interaktive Geschichte

2.1 Definition

Lee Sheldon beschreibt in seinem Buch „Character Development and Storytelling for Games“ (Sheldon, 2004) den Ursprung der Narration als einen interaktiven, dialogischen Hergang. In der Frühzeit versammelten sich die Stämme nach einer erfolgreichen Jagd abends in ihrer Höhle. Dort gesellten sie sich um die Jäger und ihre Beute, um Einzelheiten über den Verlauf der Hatz zu erfahren. Die Jäger waren also die frühzeitlichen Geschichtenerzähler. Das damalige Publikum saß allerdings nicht nur still da und lauschte: Durch Zurufe und Beifallsbekundungen konnte es den Erzählenden beeinflussen und den Verlauf, den die Erzählung nahm, in nicht unentscheidendem Maße steuern. Der Informationsaustausch fand also nicht nur in eine Richtung statt, vielmehr war es eine Wechselwirkung aus Actio und Reactio.

Wenn in dieser Arbeit von „Erzählung“ die Rede ist, dann ist damit die eine Hauptgattung der Literatur gemeint, die auch unter dem Begriff „Epik“ bekannt ist. Eine Erzählung kann eine Geschichte in sowohl mündlicher als auch schriftlicher Form sein. Zu ihr zählen unter anderem das Märchen, die Legende, die Sage, das Epos, die Kurzgeschichte und auch der Roman (Internetquelle 0001).

Die Erzählung ist eine Form der Kommunikation. In der Kommunikationstheorie werden grundsätzlich zwei Arten unterschieden: Monologisch (Aus dem Griechischen: Mono = ein; Logos = Wort) verläuft eine Kommunikation dann, wenn die Information nur in eine Richtung fliesst – also wenn ein Sender spricht und der oder die Empfänger lediglich rezipieren. Dies ist zum Beispiel beim Lesen eines Buches der Fall. Im Gegensatz dazu verläuft eine dialogische Kommunikation in beide Richtungen. Der Sender bekommt eine Reaktion vom Empfänger und kann diese Informationen zur Steuerung des weiteren Gesprächs nutzen (Israel, 1999).

Interaktiv ist eine Erzählung also dann, wenn sie dialogisch verläuft. Das heißt, der Rezipient ist in der Lage, effektiv auf die Erzählung einzuwirken und somit den Verlauf der Handlung mitzubestimmen. Diese Einwirkung kann direkt in dem Augenblick passieren, in dem die Geschichte entsteht. Das ist natürlich nur dann möglich, wenn sich

Autor und Rezipient zu diesem Zeitpunkt am selben Ort befinden oder zumindest telekommunikativ miteinander verbunden sind.

Es kann allerdings auch sein, dass der Autor seine Geschichte in einer Form verfasst, die ein späteres Eingreifen durch den Rezipienten ermöglicht. In diesem Fall verliert der Autor die direkte Kontrolle über die Handlung seiner Geschichte. Er kann nur noch versuchen, die Eingriffsmöglichkeiten so zu gestalten, dass die Geschichte in jeder Kombination von Einmischungen durch den Rezipienten ein erwartetes Resultat hervorbringt. Der Rezipient ist bei der Modifikation der Erzählung seinerseits auf die Möglichkeiten zur Interaktion beschränkt, die vom Autoren vorgesehen wurde.

Je nachdem, wie komplex die Eingriffsmöglichkeiten beschaffen sind, können auf diese Weise auch Handlungen entstehen, die vom Autoren so nie erdacht worden sind.

2.1.1 Verzweigte Narrationen

Die naheliegendste Methode, dem Rezipienten einen Eingriff zu gestatten, ist, ihm an einer bestimmten Stelle zwei Alternativen in Auswahl zu stellen, wie die Geschichte weitergehen soll. Je nachdem, wie sich der Rezipient entscheidet, kommt es zu zwei unterschiedlichen Handlungssträngen. Man kann aber auch eine Vielzahl von Entscheidungspunkten quer über die Erzählung verstreuen. Eine solche Geschichte nennt man „verzweigte Narration“.

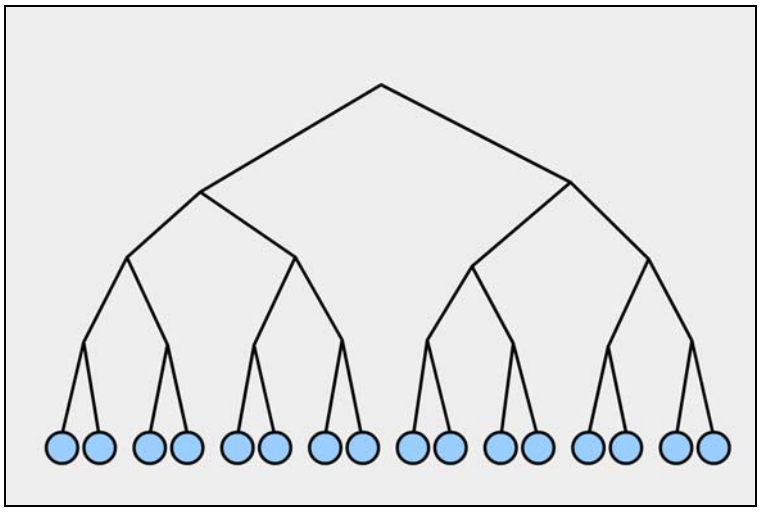

Abb. 2.1.: Baum-Struktur.

Das wohl bekannteste Beispiel für diese Art von Erzählung stellen die Bücher der „Choose Your Own Adventure“-Serie (Montgomery / Packard, seit 1979) dar. Am Ende jedes Einzelsegments der Geschichte muss sich der Leser hier entscheiden, wie es weitergehen soll. Es stehen ihm mehrere Optionen zur Verfügung. Je nachdem, wie er sich entscheidet, ist eine andere Seitenzahl vermerkt, auf der er nun weiterlesen muss.
Die einzelnen Verzweigungen haben in diesem Fall keinen Einfluss aufeinander. In Abbildung **2.1** ist eine solche „Baum-Struktur“ dargestellt. Die Entscheidungspunkte nennt man in einer solchen Darstellung „Knoten“. Weil die verschiedenen möglichen Geschichten für sich betrachtet noch immer Linear sind, die gesamte Struktur aber mehrere solcher Stränge beinhaltet, spricht man hier auch von einem „metalinearen System“.

Es ist denkbar, dass bei einer verzweigten Narration Handlungsstränge wieder zusammengeführt werden. So kann es vorkommen, dass trotz aller getroffenen Entscheidungen die einzelnen Äste der Geschichte zum selben Ende führen. Im Extremfall steht jeder Knotenpunkt in der Geschichte mit jedem anderen Punkt in Verbindung (Glassner, 2004).

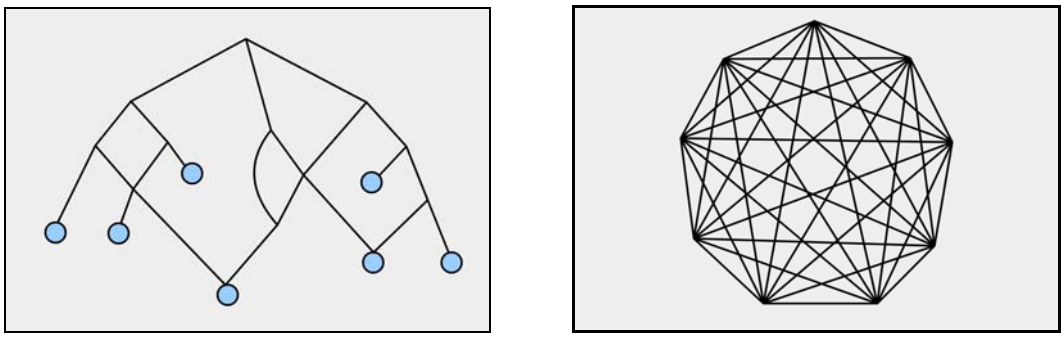

Abb. 2.1.: a) Baum-Struktur mit partiellen Ausweitungen („partial bulging tree“) und mehreren Endknoten. b) Absolutes Netzwerk.

Die verzweigte Narration hat weder in Printform noch auf dem Sektor der Computerunterhaltung jemals ein großes Publikum erlangt. Computerspiele sind im allgemeinen viel zu Verstrickt, um sich durch diese linearen Abläufe beschreiben zu lassen. Allerdings liegt die Baumstruktur der verzweigten Narration oftmals den Multiple-Choice-Auswahllisten der Dialogführung in Adventurespielen zugrunde.

2.1.2 Hypertext

Das Konzept von Hypertext ist dank des Internets heute fast jedem ein Begriff. Die dem WorldWideWeb zugrunde liegende Textbeschreibungssprache HTML (Hyper Text Markup Language) ist entwickelt worden, um Textdokumente über Querverbindungen (Links) miteinander zu koppeln. Auf diese Weise ist man nicht gezwungen, dem Text linear zu folgen, sondern kann je nach Interesse zwischen den unterschiedlichen Themengebieten hin- und herspringen. Das selbe Prinzip liegt allen Texten zugrunde, die auf die ein oder andere Weise (zum Beispiel über Quellenangaben) auf entfernte Dokumente verweisen.

Ein frühes Beispiel für dieses Vorgehen findet man bereits um 1000 v.Chr. in dem indischen Epos „Das Ramayana“, in dem laufend zwischen den Geschichten hin- und herverwiesen wird, so dass der Leser gezwungen ist, die gewohnte, lineare Lesegewohnheit aufzugeben.

Hypertext hat mit Sicherheit einen grossen Nutzen für Recherchen und wissenschaftliche Arbeiten. Zum Erzählen einer Geschichte ist es jedoch weitgehend ungeeignet.

2.2 Der Multimedia-Held und sein Weg durch die Geschichte

Computerspielhelden gehören mittlerweile zu den fiktiven Charakteren, die ihren Weg in das kulturelle „Common Knowledge“ gefunden haben. Fast jeder kennt „Super Mario“, „Lara Croft“ oder „Pac-Man“. Es gibt sogar einen regen Austausch mit anderen Medien: Leinwandhelden finden ihren Weg ins Computerspiel (Indiana Jones, Star Wars, Matrix), Computerspielhelden wiederum bekommen ihre eigenen Kinofilme (Tomb Raider, Doom, Final Fantasy). Aber auch ein Austausch mit der Literatur ist vorhanden: So ist zum Beispiel Sir Arthur Conan Doyles Superkriminalist „Sherlock Holmes“ der Protagonist gleich mehrerer Computerspiele (The lost Files of Sherlock Holmes, 1992; Mystery of the Mummy, 2003; Das Geheimnis des silbernen Ohrrings, 2004), aber auch Terry Pratchetts „Scheibenweltromane“ (Discworld, 1995; Discworld 2, 1997; Discworld Noir 1999), Alexandre Dumas „Die drei Musketiere“ (The Three Musketeers, 2005) oder Jules Vernes „In 80 Tagen um die Welt“ (In 80 Tagen um die

Welt, 2005) dienten bereits als Vorlage für Computerspiele. Die Liste ließe sich noch beliebig erweitern. Eine spezielle Sonderstellung scheint J.R.R. Tolkiens Werk „Der Herr der Ringe“ (1954) zu haben. Sein Einfluss auf die Geschichte des narrativen Computerspiels ist von der ersten Stunde an nachweisbar (vgl. Abschnitte **2.4**, **2.5**).
Der umgekehrte Fall ist zwar selten (und meist von wenig literarischem Wert), trotzdem existieren mehrbändige Buchadaptionen zu erfolgreichen Computerspielen wie „World of Warcraft“ (Knaack, 2005) „Sacred“ (Whitton, 2005) oder „Doom“ (ad Hugh, 2005).

In manchen Fällen spannen Literaturvorlage, Film und Computerspiel, sowie weitere Medien (Comichefte, TV-Serien, Gesellschaftsspiele, Rollenspielumsetzung) eine komplette Mythologie auf. So entstehen ganze Universen, die durch Konsum eines einzigen Produktes kaum erfasst werden können. Die Fans sprechen vom „Star-Wars-Universum“ oder vom „Vampire-Universum“. In Internetforen tauschen sie ihre eigenen Beiträge zu diesen Mythologien aus. Die sogenannte „Fan-Fiction“ kann eine Kurzgeschichte, ein Roman und selbst ein vollständiger Spielfilm sein.

Dieses Phänomen spricht Bände über das auftauchen interaktiver Medien. Die Konsumenten von Narrationen aller Art wollen nicht länger nur unbeteiligte Zuhörer sein. Sie fühlen sich den Geschichten und ihren Protagonisten so verbunden, dass sie selbst aktiv an der fiktiven Welt teilnehmen und mitgestalten wollen. Im Computerspiel bekommen sie eine Gelegenheit dazu.

2.2.1 Der Quest

Dem Autoren eines Computerspiels fällt eine Aufgabe zu, die viel komplexer ist, als die des herkömmlichen Autors. Er soll zwar eine Geschichte erzählen, doch er darf nur sehr bedingt darüber bestimmen, wie sein Protagonist handelt. Was das anbetrifft, muss er sich bereitwillig vom Spieler ins „Handwerk pfuschen“ lassen. Der Spieler wiederum stellt sehr widersprüchliche Ansprüche: Einerseits möchte er in der Spielwelt so viel Handlungsfreiheit wie möglich haben: Wenn er seinem Avatar einen Befehl gibt, so soll dieser die gewünschte Aktion auch ausführen. Andererseits möchte der Spieler aber, dass sich das Spiel lösen lässt und schreibt dem Autoren den Fehler zu, wenn er in eine erzählerische Sackgasse gerät.

Das Spiel muss sich zu einem erfolgreichen Ende bringen lassen. Wie ein solches Ende beschaffen sein muss, weiß der Spieler intuitiv: Das Abenteuerspiel „Zak McKracken“ (LucasFilm Games, 1988) startet in der Wohnung des Protagonisten. Außer einem wirren Traum in einer einleitenden Szene sind keine Konflikte erkennbar. Das Spiel ist aber dadurch noch nicht gelöst. Eine lange Tradition des Geschichtenerzählens verrät dem Spieler, dass der Held, um den Plot abzuschliessen, auf die Reise gehen, dabei an seine Grenzen stoßen und trotz aller widrigen Umstände wohlbehalten aus einem großen Konflikt gegen ein drohendes Übel hervorgehen muss. Diese Reise ist der „Quest“, der von Joseph Cambell beschriebene Weg des Helden (vgl. Kapitel **2.2.5**), der in allen Kulturen vorkommt und seit Urzeiten immer wieder erzählt werden muss. Diesem Quest widmet sich die Lehre der Dramaturgie. Kaum in einem anderen Medium sind die Umrisse dieses Heldenweges so unverfälscht und auf die wesentlichen Stationen reduziert wie im Computerspiel.

2.2.2 Kampf gegen Götter

Wie erwähnt, ist dieser Weg des Helden allen Kulturen zu Eigen und lässt sich anhand diverser Mythologien ableiten. Am besten eignet sich hierfür jedoch die griechische Mythologie, weil sie die erste ist, die sich selbst umfassend dokumentiert und analysiert. In ihren Dramen und Gedichten erzählten sich die Griechen von tapferen Helden, die in das Ränkespiel der olympischen Götter verstrickt wurden und den Kampf aufnahmen.
Sie erzählten diese Geschichten aber nicht nur, sie stellten auch Regeln auf, nach denen sich der Autor beim Schreiben zu richten hatte. Als wichtigstes Werk für das Fachgebiet der Dramaturgie ist hier mit Sicherheit Aristoteles „Poetik“ zu nennen. Bereits 350 v.Chr. formuliete er so elementare Konzepte, wie die Einheit von Zeit, Ort und Handlung, die poetischen Gerechtigkeit oder das Konzept des Wendepunktes.
All diese Begriffe finden noch heute Verwendung. Besonders Hollywood-Filme richten sich stark nach Aristoteles’ Vorgaben. Wenn mit ihnen gebrochen wird (wie zum Beispiel „Pulp Fiction“ (Tarantino / Avary, 1994) mit dem Gebot von der Einheit der Zeit bricht), so passiert das sehr bewusst und gilt als provokant.

2.2.3 Drama und Komödie

Aristoteles nimmt für handelnde Personen einer Geschichte eine grundsätzliche Unterteilung in drei Kathegorien vor:

- Erstens: Menschen, die besser sind als Menschen im wahren Leben
- Zweitens: Menschen, die schlechter sind als Menschen im wahren Leben
- Drittens: Menschen, die nicht besser oder schlechter sind als Menschen im wahren Leben.

Den ersten Typ Mensch ordnet Aristoteles der Tragödie zu. Sein Schicksal ist es, ein unglückliches Ende zu finden. Dadurch, dass das Publikum um das Schicksal des Helden bangt und mit ihm mitfühlt (Eleos = Mitleid, Phobos = Furcht), gelangt es schließlich zu einer seelischen Reinigung, der Katharsis. Das Publikum hat die Prüfungen mit dem Helden zusammen bestanden, der Held ist für sie gestorben.
Heutzutage ist die nicht tragische Variante des Übermenschen sehr beliebt: James Bond, Sherlock Holmes oder Superman überleben ihre Abenteuer stets unbeschadet. Trotzdem fallen sie in diese Kathegorie. Auch mit ihnen hat das Publikum die Strapazen gemeinsam durchgestanden. Das durch den ausgelassenen Tod fehlende, abschliessende Mahnmal, das uns an den Helden kettet, wird hier dadurch erstzt, dass der Held in einer späteren Inkarnation wiederkommt, um diese „Katharsis Interruptus“ zu erneuern.
Dieser Heldentypus ist bevorzugt auch in Action-Computerspielen und im CRPG wiederzufinden. „Lara Croft“ (Tomb Raider, 1996) oder „Indiana Jones“ (Indiana Jones and the Infernal Machine, 1999) sind hier die augenscheinlichen Vertreter.

Der zweite Typ Mensch gehört nach Aristoteles in die Komödie. Auch dem komischen Helden war zu Aristoteles’ Zeiten ein böses Ende zugedacht. Der „poetischen Gerechtigkeit“ ist allerdings auch dann genüge getan, wenn wie in der Hollywood-Komödie dieser Schlechtmensch sich durch Einsicht bessert, oder der Verlierer sich dank aller Anstrengungen zum Gewinner wandelt.
Das Adventure-Spiel kennt diese Art Held durch Vertreter wie „Larry Laffer“ (Leisure Suit Larry, 1987), „Roger Wilco“ (Space Quest, 1986) oder „Guybrush Threepwood“ (The Secret of Monkey Island, 1990).

Die dritte Sorte Mensch hat für die klassische Dramaturgie keine weitere Bedeutung. Heutzutage erfreut er sich hingegen bei dem wachsenden Publikum der Seifenoper größter Beliebtheit. Von ihm handelt auch das erfolgreiche Computerspiel „Die Sims“ (2000), eine Simulation des alltäglichen Lebens. Manche Heldinnen und Helden neuerer Abenteuerspiele haben keine Qualitäten, anhand derer man sie einer der ersten beiden Kathegorien zuordnen könnte und wären darum hier aufzuführen: „April Ryan“ (The Longest Journey, 2000), „Kate Walker“ (Syberia, 2003), “Brian” (Runaway, 2004).

2.2.4 Psychoanalyse

Doch nicht nur die Analyse fiktiver Figuren ist für die Dramaturgie von Bedeutung, sondern natürlich auch die Analyse ihrer Vorbilder im realen Leben. Wie wir im nächsten Abschnitt sehen werden, liegt der Ursprung aller Mythen und Geschichten in der menschlichen Psyche verborgen. Seit der Antike war das Unterbewusstsein Thema vieler Arbeiten von Dichtern, Philosophen und bildenden Künstlern. Erst im ausklingenden 19. Jahrhundert wurde jedoch ein Schlüssel gefunden, um in diesen verborgenen Bereich einzutreten.

1890 begründeten die Wiener Ärzte Joseph Breuer und Siegmund Freud die Psychoanalyse, als sie eine Patientin behandelten, die von bis dahin unerklärlichen, neurotischen Störungen geplagt wurde. Brauer und Freud fanden jedoch heraus, dass ein verdrängtes Ereignis in der Vergangenheit der Patientin die Ursache für ihre hysterischen Anfälle war. Auch wenn die Patientin nicht in der Lage war, sich bewusst an diese Vorkommnisse zu erinnern, so schienen sie sich doch an der Oberfläche einer unbewussten Schicht ihres Denkens resident eingenistet zu haben. Durch Methodiken wie Traumanalyse und spontane Assoziationen erlangten Brauer und Freud Zugriff auf deren Inhalt. Mit der Bewusstmachung der verdrängten Sorgen der Patientin war ein erster Schritt in Richtung ihrer Heilung getan (Mertens, 2004).

Freud schreibt psychische Störungen einem Ungleichgewicht in einem Spannungsfeld zu, das allen Menschen gleichenteils zueigen ist. In jedem Menschen herrscht nach Freud ein ständiger Konflikt zwischen drei Instanzen, dem „Über-Ich“, das alle moralischen Ansprüche eines Charakters vertritt, dem „Es“, das alle Trieb-Ansprüche eines Charakters vertritt und dem „Ich“, das zwischen den anderen beiden Instanzen als Verrteter des Realitätsprinzips vermitteln muss.

Neben vielen anderen Zeitgenossen, beschäftigt sich auch Carl Gustav Jung im Kommenden mit der Psychonalayse, insbesondere mit der Traumdeutung. Jung nimmt hierbei eine abweichende Einteilung der menschlichen Psyche vor: Er unterscheidet das „Ego“, das mit dem bewussten Denken gleichzusetzen ist, das „persönliche Unterbewusstsein“, in dem alle Erinnerungen gespeichert werden, ob zugänglich oder verdrängt, und schliesslich das „kollektive Unterbewusstsein“, das alle Menschen miteinander verbindet. Dieses kollektive Unterbewusstsein ist die Quelle von Mythen und Geschichten, eine Art ständig abrufbare Lexikothek für Symbole, und jeder Mensch steht über die Psyche mit ihr in Verbindung. In seinem letzten Werk „Der Mensch und seine Symbole“ (Jung, 1968) fügt Jung diese global gültigen „Archetypen“ zu einer Art Almanach zusammen, mit dessen Hilfe sich nicht nur Psychosen und Träume, sondern auch Mythen und Kunstwerke als anhand solcher Symboliken kodierte Ausdrucksformen des kollektiven Unterbewusstseins entschlüsseln lassen.

2.2.5 Der Heros in tausend Gestalten

Die Psychoanalyse und insbesondere Jungs Konzept vom kollektiven Unterbewusstsein schafft eine Basis, anhand derer Soziologen, Ethnologen und Anthropologen später Theorien entwickeln, warum sich Erzählungen und Mythen in unterschiedlichen, von einander getrennten Kulturen trotzdem mit auffälligen Parallelen entwickeln.

So erkennt der französische Ethnologe Claude Lévi-Strauss mehrere „Short Skripts“ in den Legenden Nord- und Südamerikas, Prototypen einer Geschichte, die in ihren Variationen Kontinentübergreifend erzählt wird (Internetquelle 0017).

1949 geht der Amerikaner Joseph Campbell sogar noch weiter. In seinem stark von C.G. Jung beeinflussten Werk „Der Heros in tausend Gestalten“ entwickelt er den „Monomythos“, das eine Muster einer Heldenreise, das allen Geschichten, Mythen und Religionen der Welt zugrunde liegt.

Die Reise das Helden hat drei Abschnitte: Aufbruch, Initiation und Rückkehr. Die Einzelnen Etappen auf dieser Reise werden im Folgenden genauer beschrieben.

I. Aufbruch

Die gewöhnliche Welt

Der Held ist zu Beginn der Geschichte ein unbedeutender Charakter einer unbedeutenden Region der noch heilen Welt.

Konflikt

Ein Ereignis stört die Ballance der gewöhnlichen Welt. Der Frieden ist bedroht.

Berufung

Fast zeitgleich mit der Bedrohung präsentiert sich der Held als Kandidat, um diesen Konflikt zu beheben.

Weigerung

Der Held weigert sich oftmals zunächst, dem Ruf des Abenteuers zu folgen.

Der Mentor: Übernatürliche Hilfe

„Wer sich der Berufung nicht verschlossen hat, begegnet auf seiner Fahrt zuerst einer schützenden Figur, oft einem kleinen alten Weiblein oder einem alten Mann, die ihn mit Amuletten gegen die Kräfte der Drachen, die er zu bestehen haben wird, versieht.“

(Campbell,1949, S. 72)

Das Überschreiten der ersten Schwelle

Mit Hilfe des Mentors kann der Held schliesslich die Welt der normalsterblichen verlassen und begiebt sich in eine andere Sphäre. Das Übertreten dieser Schwelle gleicht einer Art umgekehrten Geburtsprozess. Der Übergang in die andere Welt ist hierbei wie ein Tor, und es wird wie die Argo, die der antike Seefahrer Jason durchsegeln muss, von zwei Felsen oder Torflügeln markiert.

Der Wächter

Auf der Schwelle zur Anderen Welt wacht der Torwächter. Ein Dämon, der den Helden testet. „Wird dieser nicht besiegt, öffnet sich der Eingang nicht.“(Campbell,1949, S. 91)

II. Initiation

Der Bauch des Walfisches

Auf der anderen Seite der Schwelle findet sich der Held im Bauch des Walfisches wieder, einer „Sphäre der Wiedergeburt“. Anstatt aus dem Kampf an der Schwelle als triumphierender Sieger hervorzugehen, wird der Held „ins unbekannte geschlungen und scheint getötet zu sein“.

Der Weg der Prüfungen

In diesem Traumland muss der Held nun verschiedene Prüfungen absolvieren. Dabei stehen ihm die Helfer und Gegenstände beiseite, mit denen er vor dem Übertreten der Schwelle ausgerüstet wurde. Diese Attribute gilt es nun nach und nach einzusetzen, um erfolgreich aus den Prüfungen hervorzugehen.

Am Ende seines Prüfungsweges wartet die Belohnung, das Elexir zur Rettung der Welt. Campbell unterscheidet hier vier Varianten, wie dieses Elexir sich darstellt und von wem es dargeboten wird.

Variante 1: Heilige Hochzeit

Im Zuge seiner Prüfungen findet der Held schliesslich die Prinzessin, die „göttliche Weltmutter“, den „Inbegriff aller Schönheit, die Antwort auf jedes Begehren“. Campbell hat noch viele weitere blumige Umschreibungen parat. Im positiven Fall ist sie das unschuldige Ideal der Vollkommenheit und der Held rettet sie aus ihrem Dornröschenschlaf. Ihr sind allerdings auch negative Aspekte inne, so zum Beispiel ihre Unerreichbarkeit. Sie zu erwecken rettet sie zwar, macht sie aber gleichzeitig sterblich und unvollkommen.

Variante 2: Versöhnung mit dem Vater

Wenn der Held die ultimative Segnung und das damit verbundene Elexir zur Rettung der Welt nicht in der Eroberung der Weltenmutter bekommt, so erlangt er es in der Konfrontation mit einem grimmigen gottgleichen Charakter, der den Vater repräsentiert. Der Held kann den Kampf nur siegreich überstehen, wenn er seinen infantilen Eifer ablegt und die Einheit zwischen sich und dem Vater erkennt.

Variante 3: Apotheose

Der Held wird selbst im Zuge einer Vergöttlichung zur heilspendenden Kraft, die in der Lage ist, der Welt ihren Segen zu geben.

Variante 4: Raub des Elexirs

Die Kräfte der anderen Welt versagen dem Helden das Elexir. Darum stiehlt er es einfach ohne ihre Zustimmung.

III. Rückkehr

Die magische Flucht

Der Held flieht mit dem Elexir. Besonders in Variante 4 wird er hierbei von den Wächtern der anderen Welt verfolgt.

Kampf an der Schwelle

Es kann vorkommen, dass dem Helden die Rückkehr in seine alte Welt verwehrt wird. In diesem Fall findet ein weiterer Kampf gegen einen Widersacher an der Schwelle statt.

Auferstehung

Der Held kommt zurück auf die Welt. Im Gepäck hat er das Elexir.

Segnung der Welt

So gelingt es ihm schliesslich, die alte Ordnung wieder herzustellen.

2.2.6 Hollywood

Wie bereits angedeutet, bedienen sich Hollywood-Autoren oft und gerne am Werk Campbells. Die berüchtigte Drei-Akt-Form, die von Drehbuch-Theoretikern wie Syd Field gepredigt wird, ist nichts Anderes als eine vereinfachte Version des im letzten Abschnitt dargestellten Heldenweges. Kritiker bemängeln, dass die entsprechenden Drehbuchautoren Campbells Bemühungen, eine einheitliche Topologie für Narrationen

aller Art zu schaffen, als „Schema F“ und Gussform für eine gelungene Geschichte missinterpretieren.

Syd Fields nennt sein Schema „Paradigma“. Es ist derart simpel und reduziert, dass es hier in aller Kürze aufgezeichnet werden kann. (Die Angaben in Drehbuchseiten lassen sich einfach in Zeiteinheiten umrechnen. Man rechnet, dass eine Drehbuchseite einer Minute Spielfilmhandlung entspricht.)

Anfang **Erster Akt**	**Mitte** **Zweiter Akt**	**Ende** **Dritter Akt**
Setup / Exposition (Drehbuchseite 1 – 30) **Plot Point I** (Drehbuchseite 25-27)	**Konfrontation** (Drehbuchseite 30-90) **Plot Point II** (Drehbuchseite 85-90)	**Auflösung** (Drehbuchseite 90 - 120)

Abb. 2.3.: Syd Fields „Paradigma“.

Ein Drehbuch besteht also laut Field aus drei Akten. Der erste Akt dauert etwa 30 Minuten. Hier werden die wichtigsten Rollen und Orte der Handlung vorgestellt. Am Ende des ersten Aktes findet eine einschneidende Wendung statt („Plot Point“). Der Hauptteil des Films heißt Konfrontation. Er ist etwa eine Stunde lang und wird durch einen erneuten Wendepunkt abgeschlossen. Schließlich kommt es im dritten Akt zu einer Auflösung (Field, 1991).

2.2.7 Computerspiele

Bei Computerspielen lässt sich die Handlung nicht so einfach darstellen. Die meisten Computerspiele bewegen sich schon lange nicht mehr in einer vorgegebenen Geschwindigkeit von links nach rechts, wie es noch bei Supermario zu Arcade-Zeiten üblich war (Arcade-Games = Automatenspiele). Es gibt die unterschiedlichsten Spielgattungen, und die meisten gestatten es dem Benutzer inzwischen, sich in einer dreidimensionalen Welt in „Echtzeit“ (Im Gegensatz zu vom Spiel vorgegebenen Runden-, Schritt- oder einem anderen Takt) zu bewegen. Die Einteilung in erzählerische und nicht-erzählerische Spielgattungen wird in Kapitel **2.3** vorgenommen. Hier findet

sich auch eine detaillierte Beschreibung der beiden wichtigsten, narrativen Spieleformen mit einer ausführlichen historischen Einordnung. Nur eine Sache soll an dieser Stelle schon einmal vorweg genommen sein: Es herrscht große Uneinigkeit darüber, ob ein Computerspiel überhaupt Erzählcharakter haben sollte, und ob die vorhandenen, erzählerischen Spielgattungen nicht in absehbarer Zeit vom Markt veschwunden sein werden.

2.2.8 Narrativisten und Ludologisten

Dass ein Streit besonders ernst ist, kann man auch daran erkennen, dass sich die Anhänger der unterschiedlichen Fraktionen mit einem wissenschaftlichen Namen voneinander abgrenzen.
Mit Ludologie (lat. Ludos = das Spiel) bezeichnet man allgemein die Lehre vom Spiel. Im Speziellen ist hiermit das Lager der Computerspiel-Theoretiker gemeint, die der Auffassung sind, dass die Basis eines Spiels immer die Simulation ist, die sich mit den Prinzipien der Erzählung widerspricht. Der geistige Anführer dieser Bewegung, Espen Aarseth, drückt das so aus:

> *"Simulation ist das hermeneutische Gegenstück zur Erzählung; der alternative Diskursmodus, bottom-up und emergent, während Geschichten top-down vorgeplant sind. In Simulationen werden Wissen und Erfahrung durch die Aktionen und Strategien des Spielers erzeugt, anstatt vom Schriftsteller oder Filmemacher nachgebildet zu werden." (Aarseth, 2004).*

Auf der Gegenseite stehen die Narrativisten oder Narratologen, zu deren Fürsprechern unter Anderem Janet Murray zählt, die diesen Standpunkt in ihrem vielbeachteten Buch „Hamlet on the Holodeck“ (1997) verdeutlicht.

Wie verhärtet die beiden Fronten in diesem Disput sind, wird anhand des folgenden Schlagabtauschs sichtbar, den sich Janet Murray und Espen Aarseth 2004 in einem Internetforum lieferten:

> *“Aarseth misinterprets my work […]. Just as Aarseth is uncomfortable with what he (mistakenly) sees as my attempt to assimilate digital gaming experiences into the category of story, I am uncomfortable with attempts to assimilate all participatory*

narratives into the category of game. Some digital environments, like the IF traditions that Nick Montfort describes, or like online text-based role-playing environments, are only marginally gamelike. Without demeaning gaming in any way, I would rather leave the category of participatory artifacts more open. Hence the need for a term like "interactor" or "participant" (which Bryan Loyall uses) rather than "gamer" to describe the digital experience."(Murray 2004)

" 'Games are always stories' Janet Murray claims. If this really were true, perhaps professional baseball and football teams would do well to hire narratologists as coaches." (Aarseth, 2004)

2.3 Computerspiele mit und ohne Handlung

Das Gros der aktuellen Computerspielgattungen behandelt den Plot als ein nötiges Übel. Ego-Shooter wie „Doom" (1997) kommen zwar nicht ohne eine Rahmenhandlung aus, trotz aufwendiger Filmsequenzen und teurer Grafik-Effekte befindet sie sich jedoch inhaltlich selten über dem Niveau der Supermario-Thematik: Am Ende des Levels steht ein (der jeweiligen Spielewelt angepasstes) Äquivalent des Pilzmännchens, das uns sagt: „Thank you Mario, but our princess is in another castle".

Wenn man über den Inhalt von Computerspielen redet, muss man stark differenzieren. Der Begriff „Computerspiel" ist so weit gefasst, dass eine Beurteilung des Inhalts nach einheitlichen Kriterien unmöglich ist. Übertragen auf andere Sektoren der Unterhaltungsbranche, käme dies dem Versuch gleich, über die Dramaturgie von „Gedrucktem" oder über die von „Fernsehbeiträgen" zu schreiben. Im letzteren Fall müsste man die Handlung eines Spielfilms genauso berücksichtigen wie die eines Wetterberichts.

Zur Unterscheidung der unterschiedlichen Spielprinzipien benutzt die Fachpresse den Ausdruck „Genre". Typische Beispiele für diese Art der Einteilung wären „First-Person-Shooter", „Autorennspiel" oder „Rollenspiel" (RPG = Roleplaying game).
Der Begriff „Genre" ist allerdings irreführend, weil er sich bei erzählenden Computerspielarten wie dem „Adventure" mit seiner ursprünglichen Wortbedeutung in die Quere kommt. Ein Adventure kann zum Beispiel dem Genre „Fantasy" (King's Quest, 1984; Simon the Sorcerer, 1993), „Road Movie" (Runaway, 2004; Full Throttle,

1995) und sogar „Film Noir“ (Discworld Noir, 1999) angehören. Wenn es den Aufstieg eines Rennfahrers in einem Formel1-Turnier zum Thema hätte, wäre es dadurch noch lange kein Autorennspiel. Um Verwechslungen zu vermeiden, benutze ich zur Abgrenzung verschiedener Spielprinzipien im Folgenden den Begriff „Spielgattung“. Kann einem Spiel sowohl eine Spielgattung als auch ein Genre zugeordnet werden, so handelt es sich wahrscheinlich um ein narratives Spiel.

Im Allgemeinen lässt sich bereits an der Spielgattung erkennen, ob ein Computerspiel Erzählcharakter hat oder nicht. Geschicklichkeitsspiele wie „Tetris“ kommen zum Beispiel ohne jeden erzählerischen Aspekt aus. Es erzählt nicht die Geschichte eines geschickten Kistenstaplers. Das Vorhandensein eines Protagonisten ist also ein weiteres Kriterium, um zu entscheiden, ob ein Spiel Erzählcharakter hat oder nicht. Der Tetris-Klon „Doktor Mario“ für die Spielekonsole NES bemühte sich zum Beispiel, den herabtrudelnden Blöcken einen Handlungsrahmen zuzuordnen: Der Spieler kämpft hier als Doktor Mario mit Hilfe von verschiedenfarbigen Pillen gegen blockförmige Viren, die sich in einem Reagenzglas befinden, das frappierende Ähnlichkeit mit der Tetris-Spielfläche aufweist. Damit hat Doktor Mario eine einfache Handlung. Sie steht jedoch so weit im Hintergrund, dass man nicht sagen kann, dass das Computerspiel in diesem Fall als Erzählmedium genutzt wird.

Andere Spielgattungen, die in der Regel keine dramaturgischen Elemente aufweisen, sind Sportspiele, Knobelspiele, Brettspiel-Adaptionen und Edutainment.
Zu den Spielgattungen, die nur bedingt über eine Handlung verfügen und in dessen Ausgestaltung stark variieren, zählen Strategiespiele und Simulationen.
Spiele mit Erzählcharakter sind Action-Spiele (Ego-Shooter, Jump’n’Run, Action-Adventure), das RPG und das Adventure.

Wie bereits erwähnt, sind Action-Spiele in der Ausarbeitung ihrer Dramaturgie oft sehr einfach gestrickt. Wendepunkte in der Handlung und persönliche Konflikte des Protagonisten werden hier oft vollständig vermieden. Darum wird diese Spielgattung in der vorliegenden Arbeit auch nicht weiter untersucht.

Auf die anderen beiden Spieletypen hingegen, RPG und Adventure, soll an dieser Stelle genauer eingegangen werden.

2.4 Das RPG

2.4.1 Die Geschichte des RPG

In den späten 60er Jahren entwickelte sich an den amerikanischen Universitäten eine Subkultur, die sich mit einer speziellen Art des Gesellschaftsspiels beschäftigte: Dem Rollenspiel. Studentenverbindungen, die ursprünglich militärische Schlachten nach strengen Regeln mit Miniaturen auf sogenannten Tabletops nachstellten, entwickelten diese weiter, um auf Schlachtplänen mit sechseckigen Spielfeldern (Hex-Felder) beliebige - auch fiktive - Szenarios darzustellen. Als 1965 Tolkiens „Der Herr der Ringe“ erschien und in den USA eine regelrechte Fantasy-Lawine auslöste, war es Gary Gygax, der auf die Idee kam, die Table-Tops umzufunktionieren, um auf ihnen mittelalterliche Fantasy-Schlachten als eine Art Brettspiel auszutragen. Gygax gründet noch im selben Jahr mit Freunden die Firma Tactical Studio Rules (TSR) in der er 1973 mit „Dungeons & Dragons“ das erste „Pen & Paper“ Rollenspiel überhaupt publizierte (Internetquelle 002).

2.4.2 Das „Pen & Paper“-Prinzip

„Pen & Paper“ – Rollenspiele wie „Dungeons & Dragons“ funktionieren nach dem Prinzip, dass ein Spielleiter die Moderation des Spiels übernimmt, während die übrigen Teilnehmer je einen Spielercharakter (SC) spielen. Diese SCs sind anders als bei Brettspielen wie „Mensch-Ärgere-Dich-Nicht“ lebendige, individuelle Charaktere, von denen jeder ein komplexes Arsenal an Attributen , Fähigkeiten und Charakterzügen mitbekommt, die auf einem Blatt Papier, dem „Charakterbogen“ festgehalten sind. Da sich die Werte im Laufe des Spiels ändern können, wird zusätzlich ein Stift benötigt. Zusammen betritt die Gruppe eine imaginäre Spielewelt (bei „Dungeons & Dragons“ wurde sie noch durch einen variablen Spielplan optisch dargestellt – der Großteil fand allerdings auch hier bereits im Kopf der Teilnehmer statt). Hidernisse werden durch Würfelergebnisse, die sich gegen die individuellen Fähigkeiten rechnen, überwunden. Im Computer Rollenspiel (CRPG) übernimmt der Rechner die Rolle des Spielleiters.

2.4.3 Aktuelle Strömungen

Im Laufe seiner Entwicklung wurden die martialischen Anteile des Rollenspiels immer weiter in den Hintergrund gedrängt. Aus dem Pen & Paper – Rollenspiel entwickelte sich das „Storytelling“, in dem die Teilnehmer hauptsächlich die Konflikte ihrer imaginären Charaktere unter Moderation des Spielleiters zu Abenteuer-Handlungen verflechten und nur noch gelegentlich zu den Würfeln oder zum Charakterbogen greifen. Eine extremere Form findet unter freiem Himmel statt: Rollenspieler treffen sich auf sogenannten „LARPs“ zum Life Action Roleplaying, wo sie persönlich als Darsteller in die Rolle ihres Helden schlüpfen.

Die taktischen Aspekte aus denen das Rollenspiel ursprünglich entstand, finden sich heute hauptsächlich in den Computeradaptionen wieder.

2.4.4 Die Geschichte des CRPG

Da sich die Klientel der 60er-/70er Fantasywelle mit den Usern und Entwicklern der ersten Computerspiele weitgehend deckte, ließ das erste Computer Rollenspiel nicht lange auf sich warten: „Dungeon“ war stark von „Dungeons & Dragons“ inspiriert. Es wurde 1975 von Don Daglow, einem Studenten der Claremont Graduate University geschrieben und lief auf einem PDP-10 Rechner, dem Computer der Universität. Auch wenn die Grafik mehr als primitiv war und alle Eingaben in Textform gemacht werden mussten, war es das erste Spiel, das eine First-Person-Perspektive implementierte, wie sie heute in jedem Ego-Shooter Standard ist. Einzelne weisse Vektorlinien auf schwarzem Grund deuteten hierbei die Perspektive eines Ganges an.
Fast zeitgleich entwickelte Rusty Rutherford ein ähnliches Spiel namens „pedit5“ für das PLATO System (*Programmed Logic for Automatic Teaching Operations*) an der University of Illinois. Dank der zentralistischen Rechnerstruktur konnten die Studenten über entfernte Terminals auf das Spiel zugreifen. Der eigenartige Name rührte daher, dass es unerlaubt war, den Speicherplatz des Zentralrechners durch Computerspiele zu belegen. Das Spiel wurde trotz seiner Beliebtheit kurze Zeit später gelöscht. Eine zweite Version des selben Spiels namens „Orthanc“ (nach einem der Türme in Tolkiens „Der Herr der Ringe“) wurde ebenfalls wieder gelöscht. Im dazwischen liegenden Zeitraum

entstanden an der University of Illinois, die Spiele „m199h“ und „dnd“ (kurz für „Dungeons & Dragons“) von Gary Whisenhunt und Ray Wood (Internetquelle 0005).
Das erste Spiel, das auch der Öffentlichkeit zugänglich war, hieß „Akalabeth“, lief auf einem Apple II und wurde 1980 von dem damals 18-Jährigen Richard Garriott programmiert, der bereits im Alter von 15 Jahren erste Dungeon-Spiele auf dem Fernschreiber seiner Schule, der Creek High School in Houston, programmierte. (Internetquelle 0003)

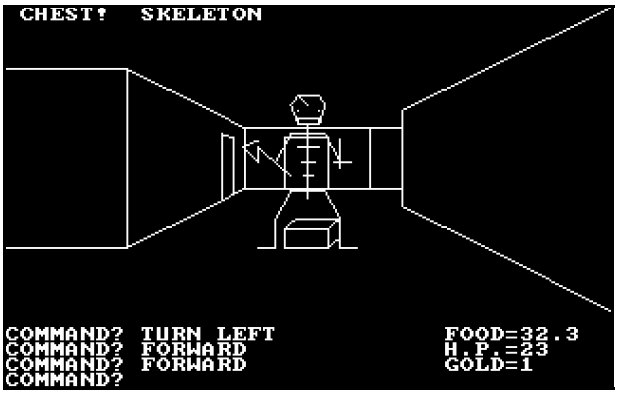

Abb. 2.4.: Richard Garriots „Akalabeth“.

Richard Garriott wurde später unter dem Pseudonym „Lord British“ für seine Ultima-Reihe berühmt, die das Computerspiel aus den 80ern bis ins Zeitalter der MMORPGs (Massively Multiplayer Online Roleplayinggames) begleitete: Ultima Underworld war das erste CRPG, welches das schrittweise Fortbewegen der Abenteurergruppe (ein Relikt aus dem „Dungeons & Dragons“-System) durch freihe Beweglichkeit in einer 3D-Umgebung ersetzte, was bei First-Person-Shootern schon länger üblich war. Ultima Online war eines der ersten MMORPGs.

2.4.5 Vom MUD zum MMORPG

MUD ist die Abkürzung für „Multiple User Dungeon“ und bezeichnet eine textbasierte Online-Umgebung, in die sich mehrere Spieler gleichzeitig einloggen können. Das erste MUD entstand bereits 1979. Es wurde von Roy Trubshaw an der University of Essex auf einer DEC-10 entwickelt und umfasste 20 Räume und 10 Kommandos. Über ein einfaches Chat-System standen die Teilnehmer miteinander in Verbindung. (Internetquelle 0004).

Nach Trubshaws Abgang von der Universität trat Richard Bartle in seine Fußstapfen und erneuerte die Spielumgebung von Grund auf. Die spielerischen Aspekte wurden deutlich ausgebaut: Es gab neue Interaktionsmöglichkeiten, ein Punktesystem, Aufträge für die Spieler und insgesamt über 400 Räume.
1980 wurde Essex dann ans ARPANet angeschlossen. Die Nachfrage aus Übersee sprengte die Bandbreite. Die Universität war gezwungen, die maximale Zugriffszeit auf das Spiel zu limitieren.
MUD ist mittlerweile bei CompuServe unter Lizenz und noch immer über das Internet erreichbar. Die deutsche MUD-Liste (www.mud.de/dml) hat 50 deutschsprachige Einträge.

Es sollte über zehn Jahre dauern, bis der nächste logische Schritt getan wurde: 1991 erschien „Neverwinter Nights“ von SSI / AOL. Es spielte in der Dungeons & Dragons Welt „Forgotten Realms“ (Vergessene Reiche) und war ein grafisch umgesetztes CRPG, an dem zunächst nur bis zu 50, dann bis zu 300 und später sogar 500 Spieler gleichzeitig Online teilnehmen konnten. Neverwinter Nights wurde 1998 trotz ununterbrochener Auslastung vom Netz genommen. Seither gewinnt das MMORPG mit Titeln wie „Kingdom of the Winds“ (1996), „Ultima Online“ (1997) oder „Everquest“ (1999) ständig an Bedeutung für den Computerspiel-Markt (Internetquelle 0007).
2005 kam mit „World of Warcraft“ der bislang erfolgreichste Titel in die Läden. Über 6,5 Mio. Mitspieler sind weltweit registriert.

2.4.6 Die Dramaturgie des CRPG

Das CRPG dreht sich inhaltlich, wie sein Pen & Paper-Vorgänger, klassischer Weise um die Charakterentwicklung. Erzählt wird die Geschichte eines oder mehrerer Helden, die sich von harmlosen Nichtskönnern zu den Rettern der Spielewelt entwickeln. Der Protagonist wird im Spielverlauf mit Fähigkeiten und Ausrüstungsgegenständen für den Kampf gegen das elementare Böse gewappnet. Dieser Kampf entlädt sich schließlich in einer epischen Auseinandersetzung gegen einen Übermenschen, der die Spielewelt bedroht. Damit verläuft das RPG kongruent zu dem von Joseph Campbell formulierten Monomythos. Die Geschichte des CRPG weicht so gut wie nie von diesem Schema ab.

2.4.7 Custom Hero

Eine weitere Eigenheit, die das CRPG von anderen Computerspielen abgrenzt, ist die Tatsache, dass die Engine dem Spieler in der Regel anbietet, einen eigenen Helden zu generieren und ihn mit einem selbst gewählten Namen, benutzerdefinierten Fähigkeiten und einem unikaten Aussehen auszustatten. Hierbei handelt es sich letztlich um eine konsequente Umsetzung des Pen&Paper – Konzeptes, nämlich des Charakterbogens. In MMORPGs verschwinden so allerdings auch die letzten Grenzen zwischen dem Helden und dem Spieler, wenn Letzterer mit seinem selbst erstellten Avatar in die Online-Welt einsteigt. Hier präsentiert sich der Spieler gegenüber tausenden von Mitspielern als die von ihm entwickelte Figur. Dadurch verliert der Held seine Übertragbarkeit. Er ist zum Symbol genau dieses einen Menschens in der realen Welt geworden, der ihn steuert.

Diese maximierte Übereinstimmung von Spieler und Protagonist führt wieder fort vom narrativen Charakter des Spiels. Die Frage nach der Handlung eines MMORPGs wie zum Beispiel „World of Warcraft“ läßt sich nicht mehr eindeutig beantworten. Es existiert zwar ein überrgeordneter Handlungsfaden, aber dem Spieler ist freigestellt, ob er sich daran beteiligen will, oder lieber etwas ganz Anderes macht, zum Beispiel angeln geht. Durch diese Freiheit tritt der Erzählcharakter des Spiels in den Hintergrund. Es gibt keinen Anreiz, das Spiel zu einem Abschluss zu bringen – es ist nicht einmal möglich. Dadurch wird der Erzählung die Auflösung entzogen. Ein Handlungsbogen kommt nicht zustande.

2.5 Das Adventure

Als Abenteuerspiel oder Adventure bezeichnet man ein Computerspiel dann, wenn es Erzählcharakter hat und ohne physische Fähigkeiten wie Geschichlichkeit und Koordionationsvermögen, also nur durch das Lösen von Rätseln zu einem positiven Ende zu bringen ist. Es sollte über einen abgeschlossenen Handlungsbogen und über einen Protagonisten verfügen. Die ersten Beispiele dieser Gattung basierten rein auf Textein- und ausgabe. Ihre Bezeichnung ist dementsprechend „Textadventure“. Später wurde die rein textliche Darstellung der Ereignisse durch eine grafische ersetzt. Vom Textadventure unterscheidet man daher das „Grafikadventure“. Ab Mitte der 90er Jahre war es möglich, bei der Darstellung auf vektorbasierte 3D-Grafiken anstatt auf Pixelgrafiken zurückzugreifen. Daher gibt es eine weitere Unterscheidung des Grafikadventure in „2D-Grafikadventure“ und „3D-Grafikadventure“. Der Begriff „Point and Click Adventure“ bezieht sich auf ein spezielles User-Interface, welches die Eingabe durch Text über einen Parser ablöste. Die Geschichte des Adventure beginnt 1972 wie die Geschichte der interaktiven Erzählung – in einer Höhle.

2.5.1 Colossal Cave

Lange vor dem CRPG wurde von dem Informatiker und Höhlenforscher William Crowther ein interaktives, textbasiertes Programm zur Begehung des real existierenden Tropfsteinhöhlensystems Mammoth-/ Flint-Ridge geschrieben. Über die Eingabe der Himmelsrichtung konnte man von Raum zu Raum gehen und eine Beschreibung des entsprechenden Bereichs der Höhle lesen. 1975 fiel das Programm seinem Studenten Don Woods in die Hände. Er erweiterte den Quellcode um einen Piraten, einen Zwerg, verschiedene Schatztruhen und ein Punktesystem und verteilte es unter dem Namen „Adventure“ (oder „ADVENT“; Dateinamen durften damals nur sechs Zeichen lang sein) über das ARPANet, das damals als Vorläufer des Internets die Universitätscomputer verband. Das Spiel erfreute sich größter Beliebtheit bei den Rechenzentrumsmitarbeitern der angeschlossenen Universitäten.
Auch Woods bediente sich hierbei – fast als sei dies eine Voraussetzung zur Erschaffung einer neuen Spielgattung – an Elementen des Buchs „Der Herr der Ringe“. Doch damit schrieb Woods, ohne es vielleicht zu wollen, selbst Geschichte: „Adventure“ diente den ersten kommerziellen Textadventures als Vorbild, aber auch

den MUDs, an deren Entwicklung Crowther beteiligt war, und der ganzen IF-Community (IF = Interactive Fiction, s. **2.5.4**). „Adventure“ hob letztlich die gleichnamige Computerspielgattung aus der Taufe und wurde der Prototyp für eine Vielzahl von ausgezeichneten Computerspielen. Heute ist das Spiel auch unter dem Namen „Colossal Cave“ bekannt (Internetquelle 006).

2.5.2 Infocom

Die ersten, die sich von Woods aberwitzigem Höhlenrundgang inspirieren ließen, waren Marc Blank, Bruce K. Daniels, Tim Anderson, and Dave Lebling vom Massachusetts Institute of Technology, kurz MIT. Noch während ihres Studiums schrieben sie an „Zork“, (den ursprünglichen Namen „Dungeon“ verwafen die vier „Dungeons & Dragons“ – Fans, weil sie nicht das Risiko einer Warenzeichenverletzung eingehen wollten), dessen erste Version ebenfalls ein ARPANet - Renner wurde. Nach Verlassen des MIT gründeten sie die Firma Intercom und stellen 1979 das Spiel in der endgültigen Fassung fertig. Was fehlte, war eine Möglichkeit, das Spiel, das bislang nur auf Mainframe-Computern lief, der Öffentlichkeit zugänglich zu machen. Die Z-Maschine war geboren, ein Emulator, der auf allen damaligen Heimcomputern lief, und den Programmcode von Zork abspielen konnte. In Personal Software Inc., später auch bekannt unter Visicorp, fanden sie ihren ersten Distributor und vertrieben Zork noch 1979 für den TRS-80 und 1980 für den Apple II. Damit begann die Erfolgsgeschichte von Infocom. Zork solte noch vier Nachfolger bekommen und sich über eine Millionen Mal verkaufen, bis Infocom 1986 schließlich von Activision aufgekauft wurde.

2.5.3 Textadventure-Semantik

Textadventures, wie die von Infocom, waren trotz ihrer spieltechnischen Einfachheit über einen vergleichsweise großen Zeitraum sehr gefragt. Ihre Funktionsweise blieb dabei im Großen und Ganzen unangetastet: Die Spielhandlung wurde in Textfragmente aufgebrochen, die auf bestimmte Kommandos des Spielers hin ausgelöst wurden. Die per Tastatur eingegebenen Befehle wurden von einem Subprogramm, dem sogenannten „Parser“, interpretiert. Lediglich die Anzahl möglicher Befehle und deren Komplexität wurde nach und nach erweitert.

Der Stil der Texte hatte einen hohen Wiedererkennnungswert: Die Befehle mussten in der Form „open window“ oder „go east“ abgegeben werden. Die Reaktionen blieben kurz und pragmatisch, bemühten sich aber, die Szene mit diesen knappen Worten vollständig zu beschreiben.

Als Beispiel habe ich hier den Spielbeginn von Zork, wie ich es zum ersten Mal gespielt habe, abgetippt. Die spitzen Klammern „>“ markieren die Zeilen, die ich selbst eingegeben habe.

```
West of House
You are standing in an open field west of a boarded house, with
a boarded front door.
There is a small mailbox here.

>look at mailbox
The small mailbox is closed.

>open mailbox
Opening the small mailbox reveals a leaflet.

>eat leaflet
I don’t think the leaflet would agree with you.

>read leaflet
<Taken>
“WELCOME TO ZORK!

ZORK is a game of adventure, danger, and low cunning. In it you
will explore some of the most amazing territory ever seen by
mortals. No computer should be without one!”

>_
```

Wie sich zeigt, haben Marc Blank und Co mit meiner Dummheit gerechnet. Es war ziemlich offensichtlich, was ich mit dem Faltblatt tun sollte, das ich in dem Briefkasten fand. Natürlich interessiert mich als neugieiger Spieler auch, was auf der Nachricht steht. Vielleicht ist es ja ein Hinweis zur Lösung des Spiels? Trotzdem wäre die Geschichte wenig interaktiv, wenn ich mich nicht anders entscheiden könnte, indem ich zum Beispiel versuche, das Faltblatt zu essen. Tatsächlich bekomme ich eine passende

Reaktion. Dass ich es trotz aller Sinnlosigkeit ausprobieren durfte, versöhnt mich mit der etwas schnoddrigen Antwort. Ich wusste ja bereits, dass das keine gute Idee war.

Dieses Augenzwinkern, mit denen der Parser auf die bisweilen absurden aber doch denkbaren Befehle des Spielers reagiert, sind ein Stilmittel, das sich quer durch die Spielgattung zieht. Als Spieler ist man enttäuscht, wenn man ständig als Reaktion nur ein (allerdings typisches) „Das klappt so nicht." bekommt. Man fühlt sich alleingelassen und verliert die Lust, unkonventionelle Ideen auszuprobieren. Wenn sich ein Spiel jedoch nur durch konventionelle Befehle lösen ließe, dann wäre es ein ziemlich langweiliges Spiel. Im Optimalfall erzählen gerade die Aktionen, die nicht zur Lösung des Spiels beitragen, die Geschichte. Wie man das bewerkstelligen kann, zeige ich in Kapitel 3 am Beispiel von „Edna bricht aus".

2.5.4 Interactive Fiction

> *"We unleash the world's most powerful graphics technology. You'll never see Infocom's graphics on any computer screen. [...] We draw our graphics from the limitless imagery of your imagination - a technology so powerful, it makes any picture that's ever come out of a screen look like graffiti by comparison."* (Infocom 1983)

Infocom sah sich und sein Produkt als krasse Alternative zum ihrer Meinung nach dementierten Action-Spiel. Sie führten den Begriff „Interactive Fiction" als neuen Oberbegriff der Spielgattung ein - ein Vorgang, der von Spiele-Theoretikern wie Espen Aarseth als narzistischer Akt ohne jede Relevanz kathegorisiert und weder vom Markt noch von der Fachpresse angenommen wurde. Der Spiele-Theoretiker Jesper Juul formuliert dies auf seiner Internetseite (Internetquelle 0004) folgendermaßen:

> *„[...], the term is basically used to claim literary qualities for a game. [...]Interactive fiction has from the very beginning been defined in opposition to other types of computer games, but later on many games have been promoted as more true "interactive fictions" than other games with the same label. In actuality, the products labelled interactive fiction have not developed much on a structural level; they haven't become more complex or dynamic. The primary development has rather been a move from text-based games to games based on graphics. Interactive fiction is then two things: A utopian idea and a genre continually claiming to have created this utopia."*

Trotz Infocoms Höhenflügen verloren rein textbasierte Adventures - in ihrer Glanzzeit die Kassenschlager der Computerspieleindustrie - mit der Einführung von Grafik in die Spielgattung für den Markt jegliche Bedeutung. Das Spielprinzip und Infocoms Philosophie blieben jedoch durch eine aktive Community lebendig, die sich unter dem von Infocom geprägten Begriff „Interactive Fiction“ (IF) in „USENET-Groups“ organisierte, um interaktive Romane mit literarischem Anspruch zu schreiben und nicht-kommerziell auszutauschen. „rec.arts.int-fiction“ und „rec.games.int-fiction” sind noch immer aktiv. Jährlich findet seit 1995 ausserdem die „Interactive Fiction Competition“ statt. Wie der Name nahelegt, handelt es sich hierbei um einen Wettbewerb für IF-Autoren. Später kamen die Wettbewerbe „Spring thing“ und „XYZZY Awards“ hinzu.

Es ist vielleicht dem guten Klang des Ausdrucks „Interactive Fiction“ zu verdanken, dass nach wie vor Uneinigkeit darüber herrscht, was der IF zuzuornen ist. In der Fachliteratur kann damit im weitesten Sinn jedes Medium tituliert weden, dass mit Engriffsmöglichkeiten durch den Rezipienten eine Geschichte erzählt. Im engeren Sinne sind damit narrative Computerspiele gemeint. Im noch engeren Sinne ist es ein Synonym für die Spielgattung „Adventure“, grenzt hierbei in seiner klassischen Wortbedeutung das reine Textadventure als einzig wahre Form von den übrigen Produkten ab, und steht letztlich auch stellvertretend für die Produkte der nonkommerziellen IF-Community, die sich erst nach der Zerschlagung von Infocom formierte.

2.5.5 Erste Grafik-Adventures

Das erste Mal wurden 1980 Grafik-Elemente in das Adventure übernommen. Das Ehepaar Ken und Roberta Williams schrieb mit „Mystery House“ einen Kassenerfolg für den Apple II, eine abenteuerliche Hausbegehung im Stil eines Agatha Christie Romans. Der Parser von „Mystery House“ war dem von „Zork“ unterlegen, und die Grafik bestand, ähnlich wie bei den ersten RPGs, aus primitiven Vektorlinien. Trotzdem stieß „Mystery House“ Infocom damit vom Adventure-Thron. Die Williams’ gründeten die Firma „On-Line Systems“, die später in „Sierra On-Line“ umbenannt wurde und im kommenden Jahrzehnt die Marschrichtung der Spielgattung angeben sollte.

Ein weiterer früher Vertreter des Grafik-Adventures war 1982 „The Hobbit“ von Beam Software / Melbourne House, nach dem gleichnamigen „Der Herr der Ringe“-Prequel von J.R.R. Tolkien. Neben der Grafik gab es in diesem Spiel noch eine unerhörte Anzahl an weiteren Neuerungen, besonders was die Eingriffsmöglichkeiten des Spielers anbelangte. An Interaktivität liegt „Der Hobbit“ noch heute weit über dem Standard eines klassischen Adventures: Man konnte Gegenstände beliebig miteinander verschachteln. Die NSCs wurden anhand ihrer unterschiedlichen Fähigkeiten vom Computer gesteuert. So bereiste ein virtueller Gandalf vollkommen selbstständig ununterbrochen die Spielewelt, nahm Gegenstände auf, legte sie woanders wieder ab und konnte sogar gefangengenommen werden, ohne dass der Spieler etwas davon bemerkte. Der Parser konnte nun auch sehr verschachtelte, komplizierte Sätze mit Interpunktion, Präpositionen, Adjektiven und Adverbien erkennen. Schließlich verfügten Charaktere und Gegenstände sogar über eine primitive Physik, die zum Beispiel ihre Flugeigenschafften beeinflusste, wenn man sie dem Gegner entgegenschleuderte.

2.5.6 Sierra On-Line

1983 wird Sierra On-Line von IBM beauftragt, ein Spiel zu konzipieren, um die grafischen Fähigkeiten ihres neuen Heim-Computers „PCjr“ zu demonstrieren. Es entstand „King’s Quest“, das erste Adventure, in dem der Spielercharakter als Figur grafisch im Bild dargestellt war und mittels des Interfaces (in diesem Fall die Pfeiltasten der Tastatur) durch die Szene bewegt werden konnte. „King’s Quest“ wurde ein großer Erfolg und auf diverse andere Heimcomputer-Systeme umgesetzt. Es gab sieben Nachfolger.

Die nächsten Spiele von Sierra entwickelten sich nicht nur in der grafischen Darstellung kontinuierlich weiter, auch die Hauptcharaktere wurden immer tiefer ausgestaltet. So entstand (beinahe schon das Zielpublikum verspottend) der Prototyp des Adventure-Helden als ein linkischer, inkompetenter aber liebenswerter Sozialphobiker. Sierras Autorenduo Scott Murphy und Mark Crowe (auch bekannt als „The Two Guys from Andromeda“) machten zum Beispiel 1986 mit der ersten Episode der insgesamt sechsteiligen „Space Quest“-Reihe die Öffentlichkeit mit ihrem Protagonisten „Roger Wilco“ bekannt, einem tolpatschigen Weltraumhausmeister. Ein anderes Beispiel ist der unbegabte Möchtegern-Casanova „Larry Laffer“ aus Al Lowes beinahe siebenteiliger

„Leisure-Suit-Larry"-Reihe (Teil vier wird übersprungen. In Teil fünf versucht sich ein verkaterter Larry verzweifelt an die Begebenheiten des verlorenen Teils zu erinnern).
Dabei bietet Sierra in seinen Adventures eine breite Palette unterschiedlicher Genres an: Science-Fiction (Space Quest), Fantasy (King's Quest, Quest for Glory), Kriminalgeschichte (Police Quest, Man Hunter) und sogar Western (Freddy Pharka Frontier Pharmacist) gehören zum Repertoire der Software-Schmiede. Weitere wichtige Titel sind die zweiteilige Laura Bow-Reihe, in der Sierra On-Line mit einer Agatha-Christie-ähnlichen Krimi-Athmosphäre zu seinen Wurzeln zurückkehrt, die dreiteilige Gabriel Knight Reihe von der amerikanischen Mystery-Romanautorin Jane Jensen und die witzigen Gobliiins-Spiele, die vollkommen ohne Dialog auskommen und über hauptsächlich mechanische Puzzle die Geschichte von gartenzwergartigen Gnomen erzählen, welche eine fröhlich bunte Fantasy-Welt bevölkern.
Neben Adventure-Spielen hat sich Sierra On-Line auch in anderen Computerspiel-Gattungen einen Namen gemacht. Nicht zuletzt waren sie es, die den jungen Richard Garriot zu Beginn seiner Ultima-Reihe unter Vertrag nahmen.

Die Erfolgsgeschichte von Sierra riss ab, als Lucasfilm Games 1987 mit „Maniac Mansion" das erste Point & Click-Adventure auf den Markt brachte. Bis dahin ließen sich Adventures noch immer nur über Tastatureingabe steuern. Sierra versuchte, mit einem eigenen Point & Click - Interface gleichzuziehen, doch der Schaden war bereits angerichtet. Eine neue Generation von Spielen übernahm den Markt.

Mittlerweile ist aus Sierra On-Line Sierra Entertainment geworden. Adventure-Spiele werden hier nur noch selten produziert.

2.5.7 Lucasfilm Games

Schon 1982 beauftragte George Lucas, geistiger Vater der „Krieg der Sterne"-Filme, eine kleine Gruppe von Programmierern, an der Entwicklung der jungen Computerspielbranche mitzuwirken. Im Geschäftsbericht von Lucasfilm LTD wurde 1983 ihr Auftrag folgendermaßen ausformuliert:

> *"The immediate goal is to find the best way to bring the 'Lucasfilm touch' to video games and help advance the state of the video game art."*

Ein Büro wurde in höchst kreativer Nachbarschaft angemietet: Auf der einen Seite wurde Lucasfilm Games von George Lucas' hauseigener Filmeffekt-Division „Industrial Light and Magic" (ILM) flankiert, die heute für die digitalen Effekte in fast jedem größeren Hollywood-Blockbuster verantwortlich ist. Auf der anderen Seite arbeitete eine Gruppe von Computerspezialisten, die später unter dem Namen „Pixar" digitale Kinogeschichte mit 3D-Animationsfilmen wie „Toy Story" oder „Monster AG" schreiben sollte.

Neben ihren Verdiensten um programmtechnische Neuerungen in Spielen, wie interaktive Musik (bei „Ballblazer" 1984 reagierte die Musik auf die jeweilige Spielsituation, 1991 wurde dieses Prinzip unter dem Titel „iMuse" patentiert), perspektivische Skalierung von NSC in einer prä-3D Umgebung („The Eidolon", 1985) oder dem ersten Echtzeit-Strategie-Spiel („Strike Fleet", 1987), war Lucasfilm Games hauptsächlich für die Entwicklung seiner aberwitzigen Adventure-Spiele bekannt.

Das erste Lucasfilm-Adventure kam 1986 auf den Markt. „Labyrinth" orientierte sich nur sehr grob an der von George Lucas produzierten Filmvorlage. Das Spiel blieb trotz seiner revolutionären Steuerung ziemlich unbekannt. Die bis dahin übliche Texteingabe wurde durch ein Interface ersetzt, in dem man über die Maus die entsprechenden Befehle aus einer vertikal angeordneten Liste auswählen konnte.

Um ein vielfaches aufsehenerregender war 1987 die Veröffentlichung von „Maniac Mansion". Das „Labyrinth"-Interface war zum SCUMM-Story System weiterentwickelt worden (SCUMM = Script Creation Utility for Maniac Mansion). Jetzt hatte der Spieler die Möglichkeit, den SC über 15 permanent auf dem Bildschirm sichtbare Befehle zu steuern. Desweiteren war das Spiel mit der Möglichkeit ausgestattet, drei aus sechs verschiedenen Hauptcharakteren auszuwählen. Je nachdem, wen man ausgewählt hatte, veränderte sich der richtige Lösungsweg.
Der größte Trumpf war allerdings der skurrile Humor, mit dem die Geschichte erzählt wurde. Um das Spiel zu gewinnen, musste man radioaktiv verseuchte Pflanzen zum rülpsen bringen und dem hüpfenden Tentakelmonster des bösen Doktor Fred Edison zu einer Gesangskarriere verhelfen. Die Spannung blieb dadurch aufrecht erhalten, dass man auf Schritt und Tritt befürchten musste, von den Bewohnern des Hauses gefangen genommen zu werden. Dr. Fred wanderte von Zeit zu Zeit eigenständig im Haus umher und zögerte nicht, den Spielercharakter in den Kerker zu werfen. Eine weitere Neuerung

war, dass wichtige Nebenereignisse in kurzen Zwischenschnitten, sogenannten „Cutszenen“, erzählt wurden. Dieses Stilmittel bleibt bis heute ein elementares, dramaturgisches Werkzeug für Computerspielhandlung aller Spielgattungen.

Abb. 2.5.: Bei „Maniac Mansion“ (1987) war etwa ein Drittel der Bildschirmfläche für das SCUMM-Interface reserviert.

„Maniac Mansion“ diente schliesslich sogar als Vorbild für eine gleichnamige TV-Serie, die drei Staffeln und 66 Folgen umspannte und in Deutschland unter dem Namen „Das Tollhaus“ (1991) auf dem Sender Tele 5 ausgestrahlt wurde.

Von den nachfolgenden Spielen des Maniac-Mansion-Entwicklerteams um Ron Gilbert, Aric Wilmunder, David Grossman und später Tim Schafer wurde fast jedes Einzelne ein durchschlagender Erfolg, und ergatterte hohe Auszeichnungen bei den Fachzeitschriften. Jeder neue Titel übertraf den vorhergehenden an Spielwitz und Originalität.

Aus Gründen, die noch zu erörtern sind, nahm die inhaltliche Qualität und der Charme der Spiele ab Mitte der 90er Jahre wieder ab. Das Interesse der Computerspielkonsumenten ging verloren. Andere Spielgattungen übernahmen die Führungsposition am Spielemarkt. So lässt sich sagen, dass die Titel der Point & Click - Ära den tatsächlichen Höhepunkt der Spielgattung markierten.

Nicht zuletzt deswegen orientierte ich mich bei der Erstellung von „Edna bricht aus“ sehr stark an den Abenteuerspielen dieser Periode. Einige Produkte, die mich besonders inspiriert haben, sind „The Secret of Monkey Island“ (Ron Gilbert, 1990), „Maniac Mansion 2: Day of the Tentacle“ (Tim Schafer, 1993), „Sam & Max - Hit the Road“

(1993) und „Grim Fandango“ (Tim Schafer, 1998). Die beiden Erstgenannten benutzten wie „Maniac Mansion“ das SCUMM-Interface. Spieltechnisch gesehen gab es in dieser Phase also kaum Neuerungen. Lediglich die audiovisuellen Möglichkeiten zur Darstellung wurden kontinuierlich besser, bis schliesslich Spiele wie „Full Throttle“ (1995) oder „Monkey Island 3: The Curse of Monkey Island“ (1997) optisch kaum noch von einem Zeichentrickfilm zu unterscheiden waren. Der Soundtrack wurde hier von einer professionellen Band eingespielt. Für die mittlerweile zum Standard gehörende Sprachausgabe liehen prominente Schauspieler ihre Stimmen.

Abb. 2.6.: „Full Throttle“ (1995) war ein interaktiver Zeichentrickfilm.

Mit „Grim Fandango“ wagte die Firma, die sich mittlerweile in Lucas Arts umbenannt hatte, 1998 den Schritt in die dritte Dimension. Vor vorgerenderten 3D–Hintergründen steuerte man den Polygon-Charakter Manny Calavera nicht mehr indirekt per Befehl und Mausklick, sondern direkt über die Tastatur. Mit den Pfeiltasten konnte man die Figur drehen oder vor- und zurückgehen lassen. Weitere Tasten öffneten das Inventar und führten die Standardbefehle aus („p“ für „pick up“, „u“ für „use“ usw.). Die schwierig zu erlernende Steuerung verärgerte viele Fans der Spielgattung. „Grim Fandango“ wurde trotz der mit viel Poesie erzählten Geschichte um einen mexikanischen Sensenmann ein Flop.

Im Jahr 2000 fand die Ära des Adventures mit „Monkey Island 4: Escape from Monkey Island“ ein gleichermaßen trauriges wie enttäuschendes Ende.

Technisch brilliant und vom Spielumfang her an die alten Teile der Monkey-Island-Serie anknüpfend, wirkte die Geschichte willkürlich und lieblos aus Elementen der Vorgänger zusammengestückelt. Dazu kam, dass Guybrush Threepwood, dem sympatischen Möchtegern-Pirat und Protagonisten der Reihe, der 3D-Look nicht gut zu Gesicht stand. Die Charaktere wirkten leblos, redeten aneinander vorbei und plazierten sich unglücklich in schlecht gewählten Kameraperspektiven auf dem Monitor. Einer der Gründe dafür wird sein, dass mit Tim Schafer, nach Ron Gilbert, Aric Wilmunder und David Grossman, der letzte Entwickler aus Point & Click –Tagen wütend über die Firmenpolitik bei Lucas Arts gekündigt hatte. George Lucas hatte persönlich angeordnet, den Firmenschwerpunkt weg vom Adventure, hin zu Action- und Star-Wars Merchandise-Spielen zu verschieben.

Zwei bereits angekündigte weitere Titel, „Full Throttle 2“ und „Sam and Max 2“, wurden trotz fortgeschrittener Programmierung niemals beendet.

Tim Schafer arbeitet heute für die Firma „Two Fines Production“. Mit „Psychonauts“ kam 2005 sein irrwitziges Debut ausserhalb von Lucas Arts auf den Makt – eine bunte Mischung aus Adventure, Action und Jump’n’Run.

David Grossman gründete mit einer Reihe weiterer Ex-Lucas Arts Mitarbeiter die Firma „telltale games“, die es sich, wie es der Name vermuten lässt, zum Ziel gemacht hat, den narrativen Charakter ihrer Computerspiele in den Vordergrund zu stellen. Mit der Lizenz zu Jeff Smiths Comicbuchreihe „Bone“ schaffte sich telltale games eine Grundlage für ein innovatives Konzept, das „episodische Adventure“. Kürzere Handlungsstränge und damit schnellere Herstellzeiten und günstigere Preise sollen das wiederspiegeln, was ihrer Meinung nach schon immer redundant im Adventure vorhanden war: Eine Tendenz hin zum Comic mehr denn zur Literatur, mehr zur Cartoonserie denn zum Film. So sind seit 2005 bereits zwei Bone-Episoden entstanden. Mittlerweile gelang es telltale games sogar, die begehrte „Sam and Max“-Lizenz zu erwerben. Die erste Episode kommt im Herbst und ist wahrscheinlich – wie „Bone“ – nur Online zu beziehen.

Ron Gilbert verließ Lucas Arts bereits 1995. Seitdem entwirft er für verschiedene Firmen Computerspiele für Kinder. In einem Interview mit dem österreichischen Radiosender FM4 sagte er 2004:

„Jeder mag Stories, wir erzählen uns schon seit tausenden von Jahren Geschichten. Heutzutage haben Spiele praktisch keine Geschichten, mal davon abgesehen, welche sie behaupten zu haben. Sie haben Szenarios. Ein einfacher Plot, aber sie haben keine echten Geschichten. Eine Geschichte ist eine komplexe Sache, eine Reise, die dem Zuhörer/Zuschauer/Spieler ermöglicht, etwas über sich selbst heraus zu finden. Die meisten Storys in Spielen sind einfach nur dünne Entschuldigungen für die Action, für das Töten, das später stattfinden wird. Die "Story" wird in einer langweiligen 5 Minuten Cutscene am Anfang gezeigt.

[...] Geschichten können für uns so viel mehr bedeuten, wenn wir erstmal mit ihnen interagieren können. Wir müssen halt noch rausfinden, wie man sie richtig erzählt.“

2.5.8 Point and Click

In der ersten Version des SCUMM Story Systems hatte der Spieler 15 permanent auf dem Bildschirm sichtbare Befehle zur Verfügung, um den SC zu steuern. Diese waren:

PUSH	OPEN	WALK TO	NEW KID	TURN ON
PULL	CLOSE	PICK UP	UNLOCK	TURN OFF
GIVE	READ	WHAT IS	USE	FIX

Sammelte er einen Gegenstand auf, so erschien dieser als zusätzlicher Begriff in einem dafür reservierten Bereich, dem Inventar. Vollständige Befehle an den SC wurden formuliert, indem man nacheinander auf einen Befehl, einen Inventargegenstand oder einen interaktiven Bildbereich zeigte und die Maustaste drückte (Daher auch der Name „Point & Click – Adventure“). Die auf diese Weise zusammengesetzten Befehle waren in einer Kommandozeile sichtbar, die man zum Abschluss ein letztes Mal als Bestätigung anklickte.

Das Prinzip blieb lange Zeit unverändert und auch Konkurrenzfirmen wie Sierra On-Line wechselten auf Point & Click – Interfaces. Im Grunde war diese grafische Darstellung der vorher üblichen Texteingabe bereits eine Reduzierung der Intraktionsmöglichkeiten. Selbst wenn die alten Parser selten mehr als die angegebenen 15 Befehle auflösen konnten, so war es doch dem Spieler freigestellt, auch völlig andere Kommandos einzutippen und die bisweilen lustige zufallsgenerierte Reaktion des Computers über sich ergehen zu lassen. Die Anzahl der Auswahlmöglichkeiten

verringerte sich jedoch fast von Spiel zu Spiel weiter. “The Secret of Monkey Island” und “Indiana Jones and the Last Crusade” waren auf 12 Befehle reduziert. „WHAT IS“ entfiel, weil die interaktiven Raumobjekte nun bereits beim Mouseover ihre Bezeichnung in der Kommandozeile anzeigten. „NEW KID“ wurde nicht benötigt, weil die SCs in diesen Spielen alleine unterwegs waren. „FIX“ und „UNLOCK“ entfielen vollständig. „READ“ wurde zum allgemeiner formulierten „LOOK AT“. Auch der Bestätigungs-Klick auf die Kommandozeile wurde in dieser Überarbeitung eingespart. Übrig blieben also:

OPEN	WALK TO	USE
CLOSE	PICK UP	LOOK AT
PUSH	TALK TO	TURN ON
PULL	GIVE	TURN OFF

“Maniac Mansion 2: Day of the Tentacle”, “Monkey Island 2: LeChuck’s Revenge” und “Indiana Jones 4: The Fate of Atlantis” kamen mit nur 9 Verben aus. “TURN ON” und “TURN OFF” entfielen, “WALK TO” war nun standardmäßig ausgewählt und musste deshalb nicht extra über einen Knopf verfügbar sein. Das Resultat sah so aus:

GIVE	PICK UP	USE
OPEN	LOOK AT	PUSH
CLOSE	TALK TO	PULL

So sparte man Platz und konnte die Inventargegenstände zum ersten mal grafisch als kleine Icons im Interface darstellen.
Schliesslich wechselte man wie die Konkurrenz auf ein vollständig grafisches Interface. Bei „Full Throttle“ und „Monkey Island 3: The Curse of Monkey Island“ waren die verbliebenden Befehle zu Icons zusammengefasst. Jeweils ein Inteface-Symbol für „Auge“, „Hand“ und „Mund“ ersetzten die gewohnten Begriffe „Sieh an“, „Benutze“ oder „Nimm“ und „Rede mit“. „Full Throttle“ hatte ausserdem einen Stiefel, weil Hauptdarsteller Ben immerhin ein Motorrad-Rocker war, und als solcher schon mal um sich trat. Das Interface war nun nicht mehr resident auf der unteren Bilddschirmkante zu sehen, sondern „hinter dem Mauszeiger versteckt“. Hielt man die linke Maustaste gedrückt, so öffnete es sich und man konnte den entsprechenden Befehl auswählen, indem man mit dem Mauszeiger darüber zog und die Taste losließ.

Abb. 2.7.: a) Die „Interface-Münze“ aus „Monkey Island 3“ (1997). b) Das Interface aus „Full Throttle“ (1995)

Bei „The Dig“ (1995) blieb letztlich nur noch der „Intelligente Mauszeiger“ übrig, der sich zu jedem Objekt einen passenden Befehl aussuchte und somit dem Spieler durch Mangel an Kombinationsmöglichkeiten fast schon von tiefgreifenderen Beiträgen zum Spielgeschehen ausschloss.

Die Entwicklung der großen Konkurrenz Sierra On-Line schlug derweil die gegenteilige Richtung ein. Ihr erstes Point-and-Click-Interface nach dem Textadventure versteckte die Befehle ebenfalls im Mauszeiger: Mir der rechten Maustaste liessen sich die Befehle nacheinander „durchklicken“. Eine direkte Anwahl war später durch eine versteckte Iconleiste am oberen Bildrand oder durch Tastaturkürzel möglich. In ihren letzten Adventure-Spielen fanden die Komponenten, ähnlich wie beim alten SCUMM-System eine premanente Darstellung auf dem Bildschirm.

2.5.9 Weitere Titel der Point and Click-Ära

Natürlich waren es nicht nur Sierra On-Line und Lucas Arts, die in den späten 80ern und 90ern Adventurespiele produzierten. Die meisten Mitbewerber kopierten jedoch lediglich die beliebten Vorreiter und hatten darum für die Entwicklung der Spielgattung keine weitere Relevanz. Aufgrund der hohen Nachfrage wurden aber auch diese Produkte zu Kassenerfolgen.

Zu ihnen zählt die „Simon the Sorcerer“-Reihe von Adventure Soft, die „The Legend of Kyrandia“-Reihe von Westwood, die „Discworld“-Reihe von Psygnosis sowie unzählige Einzeltitel.

2.5.10 Fan-Adventures

Nachdem sowohl Sierra On-Line als auch Lucas Arts ankündigten, keine Adventurespiele mehr zu produzieren, rettete sich die Fangemeinde, ähnlich wie beim Textadventure die IF-Community, in den Untergrund des Internets. Mithilfe von Tools wie AGS (Adventure Game Studio) stricken Hobbyprogrammierer dort an ihren Amateur-Adventures. Über die Internetseite *http://www.adventuregamestudio.co.uk/* lassen sich die besten Resultate frei herunterladen. News über weitere Independent-Fanprojekte werden auf *http://www.adventuregamers.com/underground/index.php* gesammelt.

2.5.11 Myst

1993 schlugen Rand und Robin C. Miller von der amerikanischen Firma „Broderbund" eine ganz neue Richtung für die Spielgattung ein: Ihr Titel „Myst" spielte in einer vorgerenderten 3D-Welt, durch die sich der SC ähnlich wie bei den frühen RPGs Feld für Feld vorwärts bewegt. Sofern man „Myst" überhaupt zur Gattung „Adventure" zählen kann (aufgrund des fehlenden Protagonisten herrscht über diesen Punkt Uneinigkeit in der Fachwelt), so wäre es damit eines der ersten 3D-Adventures überhaupt. Vorgerenderte Kamerafahrten zwischen den Einzelbildern, die in Quick Times MOV-Format hinterlegt sind, erzeugen die Illusion, sich durch eine komplette 3D-Welt zu bewegen. Der SC ist bei Myst nicht länger als Charakter im Bild enthalten, sondern man sieht den Spielverlauf aus seiner Perspektive. Ein „intelligenter Mauszeiger" entscheidet, auf welche Weise ein angeklicktes Objekt benutzt wird.
Seine außergewöhnliche Mischung aus 3D-Grafiken und stimmungsvollen Rätseln bescherten dem Spiel einen riesigen Erfolg. „Myst" wurde mit über einer Millionen Kopien das bis dahin am meisten verkaufte Computerspiel. Es bekam vier Nachfolger (Riven, Exile, Revelation, End of Ages) und ein Prequel (Uru – Ages beyond Myst). Andere Hesteller kopierten das Konzept, so zum Beispiel „Cyro" mit ihrer „Atlantis"-Reihe.

Kritiker sahen in Myst trotz seines großen Erfolges widersprüchlicher Weise die ersten Anzeichen für das Aussterben des Adventures. Sie argumentierten, dass die hier eingenommene Erzählperspektive (unter Anderem das Fehlen eines Protagonisten) fort

vom erzählerischen Charakter der Spielgattung hin zu den Vorgehensweisen der Konkurrenz aus dem Action-Sektor führe. Verfechter hingegen sahen in den Myst-Spielen eine Chance, das Adventure in das 3D-Zeitalter zu retten.

2.5.12 Die dritte Dimension

Die schiere Möglichkeit, fiktive Welten über den Computer in drei Dimensionen abzubilden, in denen man sich als Spieler frei bewegen kann, wandelte sich in kürzester Zeit zu einer Grundvoraussetzung für ein marktfähiges Spiel. Selbst Zeichentrick-Adaptionen, deren Vorlage zweidimensionaler nicht hätte sein können, mussten in ihrer Computerspielumsetzung dreidimensional sein, was zu teilweise absurden Ergebnissen führte.

Abb. 2.8.: Selbst zu der 2D-Legetechnik-Trickserie „South Park“ gibt es ein 3D-Spiel.

Auch vor dem Adventure machte diese Entwicklung nicht halt. Ein Hersteller nach dem anderen hörte auf, zweidimensionale Spiele zu produzieren. Fast jede noch laufende Abenteuerserie bekam einen 3D-Nachfolger.

Tatsächlich waren diese 3D-Adventures jedoch (mit einigen wenigen Ausnahmen) Schummelpakete: Der Spieler konnte sich in dieser neuen Generation keinesfalls frei in einer virtuellen Umgebung bewegen und umsehen, es blieb vielmehr bei statischen Hintergründen, die nun statt gezeichneten Grafiken modellierte und vorgerenderte 3D-Gemälde waren. Lediglich die im Spiel vorkommenden Figuren benutzten die 3D-Engines, um Drehungen, Lichteffekte und Skalierungen berechnen zu können, die vorher nur mühselig und mit viel Aufwand umzusetzen gewesen wären.

Der deutlich sichtbare Nachteil hierbei ist, dass das Handwerk des 3D-Designers noch in den Kinderschuhen steckt. Zuvor hatte man sich über die Jahre hinweg von grobpixeligen Figuren auf 16 farbigen Hintergründen zu professionellen Cartoonanimationen auf farbenfrohen Aquarellen gesteigert. Mitte der 90er Jahre waren Hard- und Software schließlich gut genug, dass man bei der Gestaltung der Spiele verlustfrei auf die Vorlagen von Illustratoren und Animatoren zurückgreifen konnte, die ihr tradiertes Handwerk nun eins zu eins auf den Computer übertragen konnten. Das Know-How war also schon lange vorhanden, nur die Technik war noch nicht so weit. Der Beruf des 3D-Designers hingegen ist praktisch von null auf parallel zur Entwicklung der Technischen Möglichkeiten entstanden, die allerersten Anfänge fanden zeitgleich und Tür an Tür statt.

Abb. 2.9.: a) Szene aus „Monkey Island 3“ in 2D mit aquarell-coloriertem Hintergrund. b) Szene aus „Monkey Island 4“ mit Polygon-Charakteren und vorgerenderten Hintergründen.

Bis heute ist das Prinzip des 3D-Adventures unverändert und führt auf dem Spielemarkt eine Art Schattendasein. Und wenn der gelegentlich erscheinende Titel mal wieder schamlos von sich behauptet, die neue Ära des Adventures einzuleiten, ist mittlerweile dem hoffnungsvollsten Anhänger klar, dass dahinter nichts weiter steckt als eine halbherzige Werbestrategie.

2.5.13 Aussterben einer Spielgattung

> *"Is Adventure Dead?" must be the most incendiary title of the many lectures, panel discussions, roundtables and presentations here at the 1999 Game Developers Conference.*
>
> *Al Lowe, März 1999*

Über den schwindenden Erfolg des Adventures und die Abwendung der Hersteller von ihrem ehemaligen Goldkind wird noch heute viel diskutiert. Die Experten sind sich uneinig darüber, ob die Spielgattung endgültig tot ist, oder noch einmal eine Renaissence erleben wird. Sie streiten ebenfalls darüber, wer oder was die Auslöser für diesen Prozess sind. Schließlich wird sich die Frage gestellt, was man besser machen müsste, damit auch weiterhin Abenteuerspiele und interaktive Geschichten ihren Platz auf dem Computerspielemarkt haben. Ich hoffe, mit meinem Konzept zu „Edna bricht aus“, welches das Thema von Kapitel **3** ist, einige Anregungen in dieser Richtung liefern zu können. Abschließend werde ich in meinem Fazit noch einmal auf diesen Punkt zurückkommen und - in Einbezug meiner praktischen Erfahrung bei der Erstellung von „Edna bricht aus“ - meine eigene Einschätzung abliefern.

3 Das dramaturgische Konzept

Hier beginnt der praktische Teil dieser Diplomarbeit. Im Kommenden werde ich die Vorgehensweise bei der Erstellung meines eigenen Entwurfs einer interaktiven Erzählung, dem 2D - Point and Click – Grafikadventure „Edna bricht aus“, erläutern. Mein Ziel ist es, das Medium „Computerspiel“ zu benutzen, um eine spannende Geschichte zu erzählen. Diese Geschichte sollte erstens die Qualitäten einer Erzählung aufweisen und zweitens die Interaktivität des Mediums als dramaturgisches Element ausnutzen. Der Versuch ist dann geglückt, wenn es mir gelingt, eine Erzählung zu kreieren, deren dramaturgischer Effekt ohne die interaktiven Elemente nicht zustande gekommen wäre.

3.1 Die Idee

Laut Espen Aarseth ist die Grundlage jedes Spiels die Simulation (Aarseth, 2004). In einem Spiel lassen sich Handlungsweisen und Strategien testen, ohne dass man bei einem Versagen physische oder soziale Negativfolgen befürchten muss. So kann man in einem Flugzeugssimulationsspiel testen, ob es einem gelingt, einen Düsenjet auf einem Flugzeugträger zu landen. Man braucht keine Angst zu haben, in einem Flugzeugwrack auf dem Meeresgrund zu enden. Es ist „nur ein Spiel“.
In einer Erzählung wird ein ähnlicher Effekt erzielt: Eine beispielhafte Handlung, die im tatsächlichen Leben schwierig, gefahrvoll oder nicht ratsam wäre, wird hier vom Autor für uns gedanklich durchgespielt. Wie eine Crashtest-Puppe schickt er seinen Protagonisten auf die Testbühne seiner Geschichte. Anhand dessen Schicksal können wir Erfahrungen gewinnen, ohne uns selbst in Gefahr zu begeben. Der erzielte Lerneffekt ist umso größer, je mehr wir uns mit dieser Crashtest-Puppe identifizieren. So warnt uns die Geschichte von Ikarus, der von seinem erfindungsreichen Vater Daedalus ein paar Flügel gebaut bekommt, vor allzu übermütigen Höhenflügen. Die Parallelen zum Flugsimulator sind evident: Auch in diesem Fall lernen wir aus Ikarus Absturz, ohne selbst Schaden davonzutragen.

Daedalus gilt in der griechsichen Mythologie übrigens als der Erbauer des Labyrinths auf Kreta. Er ist sozusagen der Urvater aller Spieledesigner.

Die Kunst der Erzählung liegt nicht darin, dass man dem Publikum die Geschichte wieder und wieder vorträgt, bis Ikarus endlich soviel Übung hat, dass er es doch schafft. Diese Qualität ist offenbar dem Spiel vorbehalten. In einem Spiel wird bei einem negativen Ausgang dazu eingeladen, es noch einmal zu probieren und die Strategie anzupassen. Bei der Erzählung wird dazu eingeladen, über die eine vorgestellte Strategie (die in diesem Fall zum Misserfolg führte) zu reflektieren, damit man in einer ähnlichen Situation nicht denselben Fehler begeht.

Computerspiele sind leistungsorientiert. Immer geht es darum, ein bestimmtes Spielziel zu erfüllen. Das Spielziel lässt sich dabei vom Spieler nicht beeinflussen. Eine Reflektion über die Natur des Ziels und die Mittel, die von der Spielewelt angeboten werden, um dieses Ziel zu erreichen, findet nicht statt. Dies war mein Ansatz für „Edna bricht aus“. Warum muss sich der Spieler nicht dafür rechtfertigen, was er in der Spielewelt anstellt? Der Spieler kommt nicht vor das Kriegsgericht, wenn er bei „Tom Clancy’s: Rainbow Six“ (1998) unverhältnismäßige Brutalität an den Tag legt. Super Mario bekommt keinen Ärger mit dem Tierschutzbund, weil er auf Schildkröten springt. Die Botschaft, die sich hier quer durch alle Spielgattungen zieht, lautet: Die Mittel sind egal, Hauptsache man gewinnt. Was wäre also, wenn man am Ende des Spiels dem Spieler doch noch eine Quittung präsentiert und ihm damit den Spiegel vorhält? Man könnte eine Handlung entwerfen, in der sich der Spieler ohne es zu merken selbst in einen nicht auflösbaren Konflikt hinein maneuvriert. Am Ende wird der Spieler dann vor eine tatsächliche Wahl gestellt, nämlich eine, in der es nicht eine richtige und eine falsche Entscheidung gibt und in der man nicht durch erneutes Ausprobieren auf die richtige Lösung stossen kann. Dies schien mir ein adäquates Konzept für meine interaktive Erzählung.

Als Thema wählte ich hierbei den von Siegmund Freud beschriebenen Konflikt zwischen „Es“ und „Über-ich“. Was für eine grausame Entscheidung muss es sein, wenn der Spieler als bewusstes „Ich“ (vertreten durch die Protagonistin Edna) zwischen seinen Trieb-Aspekten (vorrangig hier der Freiheitstrieb symbolisiert von Edans imaginärem Freund Harvey) und seinen Vernunft- und Moral-Aspekten (Der vermeintliche Antagonist „Doktor Marcel) wählen muss? Er kann sich entweder zwischen frei und schuldig oder eingesperrt und unschuldig entscheiden. Keine der Kombinationen wird dem Spieler gefallen. Er ist gezwungen, sein ganzes bisheriges Vorgehen in Frage zu stellen.

3.2 Wer bin ich?

Edna befindet sich zu Beginn der Geschichte in einer Einzelzelle und hat keine Erinnerung daran, wie sie hinein gekommen ist. Das entspricht genau der Situation des Spielers, der das Programm gerade erst gestartet hat und sich in dieser neuen Situation zurecht finden muss. Dass sie gegen ihren Willen und ohne eine Erklärung gefangen gehalten wird, gibt ihr das Recht, alles in ihrer Macht stehende zu unternehmen, um wieder frei zu kommen. Damit präsentiert sich die Welt der Regeln und Pflichten (Über-Ich) von Beginn an als Gegenspieler. Der Erwartungshaltung an Edna, in ihrer Zelle zu bleiben, kann unmöglich entsprochen werden.

Stattdessen wird Edna von nun an nur noch auf ihre Trieb-Impulse hören, die durch das Stofftier Harvey symbolisiert sind. Harvey ist das „Es“, das freie Kind in Edna. In ihm versammeln sich Ednas Freiheitstrieb, ihre Kreativität, ihr Amüsiertrieb aber auch ihre Destrudo (Zerstörtrieb). Als ständiger Ansprechpartner begleitet er Edna auf Schritt und Tritt, wie Doktor Watson es bei Sherlock Holmes tat oder Sancho Panza bei Don Qixote. Bei Bedarf kann sich Edna mit ihm über wichtige Ereignisse austauschen. Er hat aber auch Züge eines Mentors: Er kann Edna in die Vergangenheit „telemorphen“ und stattet sie auf diese Weise mit neuen Fähigkeiten aus.

Edna wird sich nicht weigern, auf Kommandos des Spielers zu reagieren, sofern sie in ihrer Macht stehen. Dabei ist es unerheblich, ob ein Kommando Sinn ergibt oder nicht: Sie spricht mit Gegenständen, bemalt Wände und kennt keine Zurückhaltung im Umgang mit anderen Personen der Spielewelt. Das macht sie zu einem liebenswerten und komischen Charakter. Als schwächliches Fräulein im Nachthemd, das umringt ist von patriarchalischen Macho-Charakteren lässt sie keine Gelegenheit aus, ihrem Trotz und ihrer Respektlosigkeit Ausdruck zu verleihen.

In den wenigen Besuchen in ihrer Vergangenheit lernen wir Ednas Hintergrundgeschichte kennen und damit auch den Ursprung ihres Freiheitsdrangs. Sie wird mal von ihrem Vater im Keller, mal von ihrem Lehrer im Wandschrank eingesperrt. Die aufgezwungene Freundschaft mit dem nervigen Alfred tut ihr Übriges. Trotzdem werden die Rückblenden bunt und fröhlich präsentiert. Diese Erinnerungen sind ihr Zuhause. Dorthin will sie zurück.

Schließlich muss Edna auch noch erfahren, dass ihr Vater wegen Mordes zu Tode verurteilt wurde, wahrscheinlich zu Unrecht. Damit kündigt sich schon früh ein zweiter Handlungsstrang an, ein Indiz zur Lösung des Rätsels um ihre gelöschte Erinnerung, aber auch ein erster Hinweis auf den nicht auflösbaren Konflikt am Ende des dritten Aktes.

3.3 Wo bin ich?

„Edna bricht aus“ handelt also von einem psychiologischen Konflikt, dem Kampf der beiden Instanzen des Unterbewusstseins im Spannungsfeld einer verdrängten Erinnerung. Was könnte da passender sein, als ein Irrenhaus als Schauplatz der Handlung zu wählen? Die ständige Konfrontation mit Gittertüren spiegelt nicht nur Ednas Freiheitstrieb wieder. Sie motiviert auch den Spieler, weiterzukommen und bildet eine hervorragende Basis für die Adventure-Typischen Rätsel: Welcher Schlüssel passt in welches Schloss? Darüber hinaus darf man in einem Irrenhaus mit einer großen Anzahl von interessanten Charakteren rechnen.

Die Spielewelt von „Edna bricht aus“ ist so aufgebaut, dass sich Edna immer wieder von innen nach außen freikämpfen muss, so dass sie nach jeder gelungenen Etappe ein größeres Stück der Welt begehen kann, bis sie sich erneut vor einer verschlossenen Tür wiederfindet. Dabei führt sie ihr Ausbruch schließlich auch über die Grenzen der Irrenanstalt hinaus. Von einer der ersten Szenen aus, dem Dach der Irrenanstalt, kann man bereits Ednas gesamte Reise überblicken, die Spielewelt also bis zu ihrem Horizont einsehen: Die Anstalt, die Straße, der Wald, die Kirche und schließlich, kaum zu erkennen, das Haus ihres Vaters. Die Richtung ist: Von innen nach außen, von oben nach unten.

Abb. 3.1.: Szene aus „Edna bricht aus“: Das Dach der Irrenantalt.

Die Anstalt liegt auf dem Gipfel eines Berges, das Haus ihres Vaters im Tal. Ich habe die Spielewelt so gestaltet, dass die ganze Umgebung eine Art Sog in Richtung des Hauses ihres Vaters erzeugt: Das Licht, die Wolken, die Sonne, das Gefälle der Straße und die Ausrichtung der Gebäude – all das zieht Edna regelrecht zurück nach Hause.

3.4 Ednas Quest

Ednas Heldenweg gliedert sich in drei Akte. Diese Unterteilung entspricht nicht Syd Fields Drei-Akt-Schema. Die Akte werden allerdings, ähnlich wie bei Field, durch die großen Wendepunkte in der Geschichte eingeleitet.

Der erste Akt spielt in der Anstalt und handelt von Ednas Flucht. Er deckt mit über sechzig Räumen und zwei Rückblenden über die Hälfte des Spielgeschehens ab. Der Akt endet damit, dass Edna mit einigen Patienten, darunter auch dem undurchsichitgen Schlüsselmeister, einen Wagen kapern und die Anstalt verlassen kann.

Im zweiten Akt hat die Gruppe um Edna einen Unfall gebaut und wird von den Handlangern des Doktors verfolgt. Es ist jetzt Nacht und Edna schlägt sich durch ein Waldstück bis zu einer Kirche durch, wo es zu einer dramatischen Konfrontation mit dem Schlüsselmeister kommt.

Der dritte Akt spielt am verlassenen Haus von Ednas Vater. Edna findet hier schließlich die Erinnerung daran, was passiert ist, wieder und muss anschließend Doktor Marcel gegenübertreten. Er stellt sie vor die schreckliche Wahl, die das Spiel beendet.

An exponierten Stellen kann der Spieler mit Harveys Hilfe in unsere Vergangenheit „telemorphen“. Mit diesem Begriff bezeichnet Harvey den von ihm moderierten Erinnerungsprozess. In diesen „Flashbacks“ können wir hauptsächlich verlorengegangene Fähigkeiten wiedererlernen. Wir erlangen aber auch Hintergrundwissen über Ednas Vergangenheit und lernen Ednas Vater sowie den nervigen Alfred Marcel, Doktor Marcels Sohn, kennen. Der Spieler merkt kaum, dass während sich Edna in der Gegenwart an den Punkt zurückarbeitet, an dem sie ihr Gedächtnis verlor, sich gleichzeitig die Geschichte in den Rückblenden langsam bis zu dem selben dramatischen Höhepunkt zuspitzt. Beide Zeitstränge enden schließlich vor Ednas Zimmer und werden dort mit der vollständigen Wiederherstellung ihrer Erinnerung zu einer linearen Geschichte zusammengeknüpft.

3.4.1 1. Akt

Das Spiel beginnt damit, dass Edna ohne Erinnerung in der engen Einzelzelle einer Irrenanstalt festsitzt. Nur ihr sprechender Stoffhase Harvey ist bei ihr. Während eines ersten Fluchtversuchs belauscht Edna ein Gespräch zwischen dem grimmigen Doktor Marcel und seinem Assistenten: Der Doktor versucht absichtlich, ihr Gedächtnis auszulöschen. Ein Grund mehr für Edna, schnellstmöglich zu verschwinden. Im Zuge ihres Ausbruchs wird sie jedoch gefangengenommen und versehentlich zu den anderen Patienten gesperrt. Dieser Rückschlag ist Segen und Fluch gleichzeitig, denn in diesem Bereich lernt Edna viele interessante Charaktere kennen. Vier davon erweisen sich als besonders nützlich: Zum Einen ist da der Alumann, ein kauziger alter Herr mit selbstgebasteltem Superheldenkostüm und einem Esoterik-Tick. Als zweites wären Hoti und Moti zu nennen, zwei sehr unterschiedliche Originale, die sich allerdings für siamesische Zwillinge halten. Schließlich trifft Edna in einer Einzelzelle auf den Schlüsselmeister, einen undurchsichtigen Typen, der einen verwegenen Fluchtplan ausgearbeitet hat.

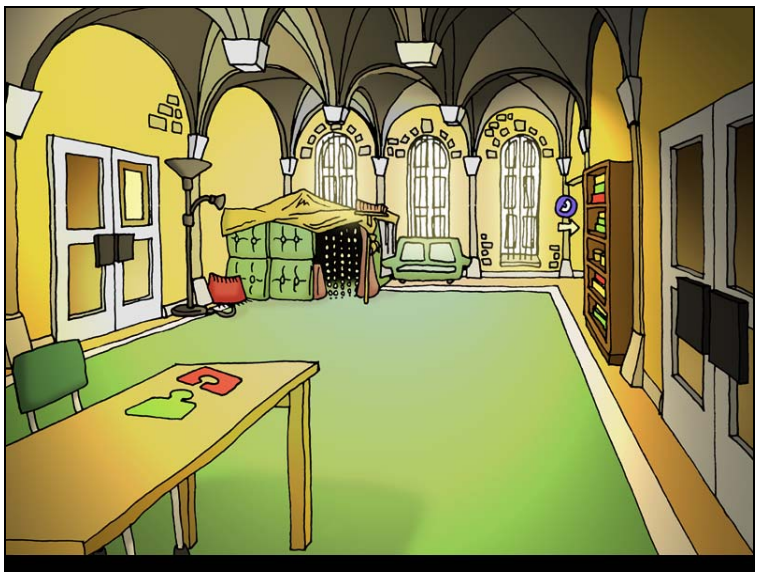

Abb. 3.2.: Szene aus „Edna bricht aus“: Der Aufenthaltsraum.

Nach seiner Anleitung kann sich Edna in mühevoller Arbeit eine Kopie des begehrten Generalschlüssels anfertigen. Gemeinsam kapern die fünf anschließend die Limousine des Doktors und entfliehen der Anstalt. Hier beginnt der zweite Teil von Ednas Reise.

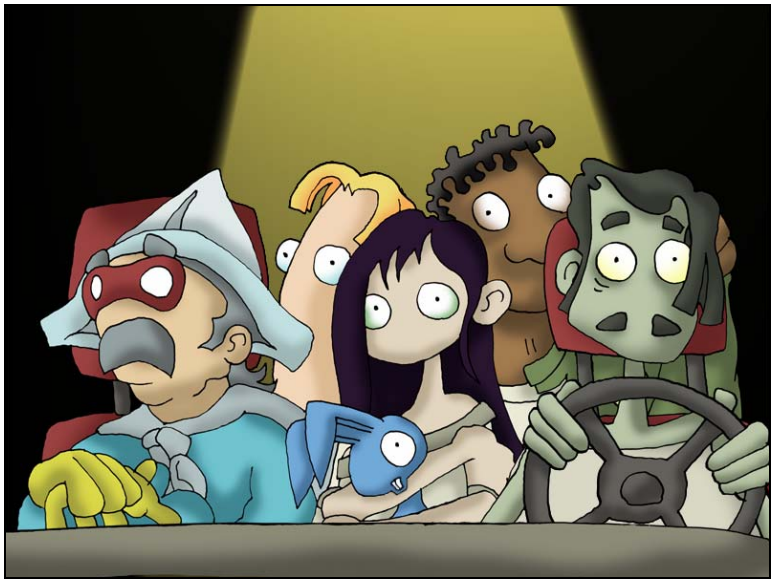

Abb. 3.3.: Szene aus „Edna bricht aus“: Die Flucht.

Im Zuge des ersten Aktes muss sich Edna zweimal in die Vergangenheit „telemorphen“. In diesen Rückblenden lernt sie ihren Vater Mattis und Alfred, den unausstehlichen Sohn von Doktor Marcel kennen. Wir erfahren auch, dass Edna bereits damals ein recht widerspenstiger Charakter war, und nicht selten zur Strafe im Keller oder in einem Schrank eingesperrt wurde. Die zweite Rückblende wird in einer Art Altarraum

ausgelöst, den Doktor Marcel für seinen Sohn eingerichtet hat. Wir erfahren also zusätzlich, dass Alfred Marcel nicht mehr am Leben ist.

Abb. 3.4.: Szene aus „Edna bricht aus“: Alfreds Altar.

Der erste Akt ist vornehmlich heiter und bunt gehalten. Umso stärker wird der Bruch, wenn der zweite Akt zuerst optisch und dann auch inhaltlich immer düsterer wird.

3.4.2 2. Akt

Die Flüchtlinge kommen nicht weit: Der Schlüsselmeister ist ein schlechter Fahrer und so landet die Limousine abseits der Straße in einem Tümpel. Als Edna erwacht, haben es sich Hoti und Moti bereits im Autowrack gemütlich gemacht. Den Alumann findet Edna nach kurzem Suchen auf einer Aussichtsplattform. Vom Schlüsselmeister fehlt jede Spur. Dafür entdeckt Edna, dass ihre Verfolger weiter unten auf der Strasse eine Blockade errichtet haben, die Unfallstelle haben sie offensichtlich übersehen. Ihre neuen Freunde weigern sich, weiter mitzukommen. Edna zieht also auf eigene Faust los.

Nachdem Edna einen Fluss und ein rostiges Tor überwunden hat, versperrt ihr das Haupttor einer Kirche den Weg. Ihr bleibt nichts Anderes übrig, als die Kirche zu betreten, um den Pastor um Hilfe zu bitten. Dadurch findet die Geschichte einen vorläufigen Höhepunkt: Edna wird mit einem selbsterzeugten Übel konfrontiert, dem Schlüsselmeister, den sie zuvor gedankenlos aus der Irrenanstalt befreit hat.

Abb. 3.5.: Szene aus „Edna bricht aus“: Die Straße bei Nacht.

Er entpuppt sich als gemeingefährlicher Psychopat, lyncht den Pastor am Glockenstrang und beschuldigt Edna, die Verantwortung dafür zu tragen: Immerhin war sie es doch, die ihn aus der Anstalt befreit hat. Damit nicht noch ein Unglück passiert, schließt er sich zusammen mit Edna in der Kirche ein.

Abb. 3.6.: Szene aus „Edna bricht aus“: Die Kirche.

Wenn Edna sich befreien will, muss sie sich etwas einfallen lassen, um nicht auch gleichzeitig den Schlüsselmeitser auf die Welt loszulassen. Ihre Anstrengungen enden damit, dass ihr Widersacher von der Empore stürzt. Edna hat den Schlüssel zum Kirchentor. Aber zu welchem Preis?

3.4.3 3. Akt

Edna erreicht schließlich das Haus ihres Vaters. Hier ist alles verwahrlost und kaputt. Die Räume wirken wie eine alptraumhafte Version der zuvor in Rückblenden besuchten Bilder aus der Vergangenheit.

Abb. 3.7.: Szene aus „Edna bricht aus“: Vor Ednas Zimmer.

Edna gelingt es, bis zu ihrem Zimmer vorzustoßen, wo sie in ihrem Tagebuch einen Hinweis darauf vermutet, was damals geschehen ist. Um das Buch schneller finden zu können, bittet sie Harvey, sie ein letztes Mal in die Vergangenheit zu „telemorphen“.

3.4.4 Showdown

In diesem Flashback ist Edna in ihrem Zimmer eingesperrt. Das Tagebuch ist schnell gefunden. Edna erkennt, dass Harvey sie genau an den Tag gebracht hat, an dem ihr Vater den Mord begangen haben soll. Sie muss schnell zu ihm. Um das zu bewerkstelligen, sperrt sie mit einem Trick zunächst Alfred in ihrem Zimmer ein, der zuvor vor ihrem Zimmer Wache geschoben hatte. Harvey und Sie teilen sich auf, um Mattis schneller finden zu können.

Der Spieler steuert jetzt Harvey von Raum zu Raum durch das Haus. Harvey sieht eine Schreckensvision nach der nächsten: Mattis kocht Kinder auf dem Herd. Mattis backt Kinder im Ofen. Mattis zersägt Kinder im Garten. Doch jedes Mal wenn Edna

dazukommt, war es nur falscher Alarm. Schliesslich findet Edna ihren Vater friedlich auf der Terrasse sitzend. Sie ist natürlich erleichtet, dass er kein Mörder ist. Mattis hingegen reagiert ungehalten und schickt Edna zurück auf ihr Zimmer. Kaum öffnet sie die Tür springt ihr Alfred entgegen. Alfred ist wütend. Er schnappt sich Harvey und will ihn kaputt machen. Edna kann das nicht zulassen. Sie schubst Alfred die Treppe hinab.

Zurück in der Gegenwart ist Edna fassungslos. Sie weiß jetzt wieder alles. Sie war diejenige, die den Mord beging und Alfred im Affekt tötete. Mattis hatte die Schuld freiwillig auf sich genommen, um sie zu schützen.

In dem Moment dieser Erkenntnis kommt Doktor Marcel die Treppe hinauf. Er erklärt Edna die letzten Teile des Puzzles: Er hatte Mattis versprechen müssen sie aufzunehmen und sie zu heilen. Der Doktor sah aber nur die eine Möglichkeit, dies zu bewerkstelligen – indem er ihre ganze Erinnerung und damit ihren aufsässigen Charakter auslöschte.

Nun endlich weiß er, was zu tun ist. Edna muss sich ihres Stoffhasen entledigen. Harvey ist Ednas letzter Anker zur Vergangenheit. Wenn Harvey fort ist, kann Doktor Marcel endlich Ednas ganzen Charakter mit all ihren verrückten Einfällen auslöschen.

Harvey beschwört uns, nicht auf den Doktor zu hören. Der Doktor ist der Einzige, der von Ednas Schuld weiss. Sie soll lieber den Doktor die Treppe hinunter stoßen. Dann sind sie endlich frei.

Edna richtet sich hilfesuchend an den Spieler. „Was soll ich nur tun?“

Es folgt eine einfache Auswahlliste:

```
Höre auf Doktor Marcel
Höre auf Harvey
```

3.5 Motivation

Bei einem umfangreichen Spiel wie „Edna bricht aus“ ist es wichtig, den Spieler zu motivieren, das Spiel nicht frühzeitig abzubrechen. Ein Mittel, um dies zu erreichen, ist, ihm in jeder Spielsituation das nächste Ziel genau vor Augen zu führen, auch wenn es noch etwas weiter entfernt ist. Es darf zu keinem Zeitpunkt vorkommen, dass der Spieler ziellos umherirrt, weil er an das „große Ziel“ noch nicht herankommt und das „kleine Ziel“ nicht kennt.

Bei „Edna bricht aus“ war das einfach. Die „großen Ziele“ sind meistens verschlossene Türen, Gitter oder bewachte Ausgänge. Sobald sich der Spieler erst einmal orientiert hat, weiß er, wo ein möglicher Ausgang ist, den er „geöffnet“ haben möchte. Meistens gibt es sogar mehrere davon. Meine Hoffnung ist, dass der Spieler diese verschlossenen Gitter genau wie Edna als Herausforderung sieht.

Ist ein Etappenziel erreicht, so gilt es, die Aufmerksamkeit des Spielers auf das nächstgrößere Ziel zu lenken. Ein gutes Beispiel hierfür ist der Augenblick, an dem es Edna gelingt, die Hintertür mit dem Generalschlüssel zu öffnen. Tritt sie nun nach draußen, so gibt es eine kurze Cutszene. Doktor Marcel kommt von seinen Bersorgungen zurück und stellt fest, dass in der Anstalt die Hölle los war. Die Wächter berichten ihm, dass sich Edna befreien konnte. Doktor Marcels Reaktion ist ein wenig ernüchternd für den Spieler:

```
      STIESEL
Herr Doktor! Eine Katastrophe! Edna ist ausgebrochen!

      DOKTOR MARCEL
Stiesel. Wie oft sind uns schon Patienten aus ihren Zellen
entwischt?

      STIESEL
Sie meinen, am Tag? So drei im Durchschnitt.

      DOKTOR MARCEL
Und wie viele davon sind über die Mauern entkommen?

      STIESEL
Naja...Keiner.

      DOKTOR MARCEL
Keiner. Richtig.
Der einzige Weg rein oder raus ist durch das Haupttor.
Und selbst wenn sie einen Weg findet, das Tor zu öffnen ... ohne
Transportmittel kommt sie keinen Kilometer weit.
```

Dieser kurze Dialog bringt den stolzen Spieler, der vielleicht sogar schon dachte, das Spiel gelöst zu haben, wieder auf den Boden der Tatsachen zurück. Er hat tatsächlich nur einen Etappensieg erreicht. Gleichzeitig wird ihm aber das nächste Hindernis vorgestellt: Die Mauer. Um diese Erkenntnis und Ednas Marschrichtung noch einmal zu festigen, gibt es auch an der Hintertür einen kurzen Dialog zwischen Harvey und Edna.

EDNA

Ich bin endlich frei!

HARVEY

Jippi.
Dann lass uns in den Zoo.

EDNA

Das geht nicht, Harv. Ich muss die Beweise finden, die meinen Vater entlasten.

HARVEY

Dann hoffe ich, die Informationen sind innerhalb dieser hohen Mauern da.

EDNA

Was?

EDNA sieht sich nach allen Seiten um. Das Anstaltsgelände ist von Mauern umgeben.

EDNA

Mist.

In diesen knappen Sätzen haben wir dei Dinge erreicht: Wir haben 1) unser Erfolgserlebnis, wir haben 2) unsere neue Richtung, und wir haben 3) sogar noch einmal das höhere Ziel, die Suche nach den Beweisen, betont.

3.6 Schlüsselsituationen

Bei der eben behandelten Situation handelt es sich also um einen Etappenwechsel. Edna bekommt eine neue Richtung und ein neues Etappenziel. Darüber hinaus ist ihr nun ein weiterer Bereich, nämlich das Anstaltsgelände, zugänglich. Tatsächlich ist jedes erfolgreich gelöste Rätsel ein kleiner Etappensieg, denn es erschließen sich nun neue Möglichkeiten. So wird auch die vorherige Unterscheidung zwischen „kleinen Zielen“ und „großen Zielen“ deutlich. „Kleine Ziele“ erschließen nur wenige neue Möglichkeiten, vielleicht sogar nur eine. „Große Ziele“ erschließen viele neue

Möglichkeiten. Da sich bei „großen Zielen“ viele Elemente und manchmal auch die Spielrichtung ändert, handelt es sich hierbei meist um dramaturgisch wichtige Punkte.

Diese Schlüsselsituationen sind in sofern wichtig, weil wir an einem solchen Punkt einschätzen können, wo sich der Charakter befindet und was er bisher alles erreicht hat. Man kann eine Art Zwischenbillanz ziehen.
Ein Beispiel: Um die Anstalt zu verlassen braucht Edna den Generalschlüssel (in diesem Fall ist die Bezeichnung „Schlüsselsituation“ also doppeldeutig). Von den folgenden Inventarobjekten können wir uns demnach sicher sein, dass Edna sie in allen Räumen jenseits der Hintertür nicht mehr im Inventar hat, weil sie diese zur Erstellung eines Generalschlüssels direkt oder indirekt benötigt hatte:

Lexikon	Hat Edna unter eine Türklinke gesteckt
Tonerde	Hat Edna zur „Tonmasse“ verknetet
Becher	Da hat Edna „Kaffee“ reingefüllt
Kaffee	Hat Edna dem Bienenmann gegeben
Fliege	Hat Edna ins Ohrenschmalz gesteckt
Formular	Wurde zu „unterschriebenes Formular“
Ohrenschmalz	Wurde mit „Fliege“ zu „Fliege in Ohrenschmalz“
Stuhlbein	Wurde zu „kaputtes Stuhlbein“
Fliege in Ohrenschmalz	Hat Edna Professor Nock gegeben
KleiderbÃ¼gel Nr. 1	Hat Edna dem Alumann gegeben
KleiderbÃ¼gel Nr. 3	Hat Edna dem Alumann gegeben
Medaille	Hat Edna in den Topf gelegt >„Medaille in Topf“
Schaufel	Hat Edna Almo gegeben
Stinkdrink	Hat Edna ausgetrunken
Chips	Hat Edna über dem Bettlaken zerbröselt
Bohnendip	Hat Edna an Peter gegeben
dampfender Topf	Hat Edna in die Form gegossen >„Topf“
Medaille in Topf	Hat Edna eingeschmolzen. >„dampfender Topf“
Form	Da wurde Gold reingegossen >„dampfende Form“
Abdruck	Hat Edna in der Heizung gebrannt >„Form“
Tonmasse	Damit hat Edna den „Abdruck“ erstellt
Cocktailkarte	Wurde zu „Modifizierte Cocktailkarte“
dampfende Form	Wurde zu „abgekülte Form“
abgekÃ¼hlte Form	Wurde zu „Generalschlüssel“

Nur einen Raum zuvor hätte sie fast alle diese Items noch haben können.

3.7 Rätselklassen

Den richtigen Schlüssel für eine Tür zu finden ist nur eine von vielen denkbaren Aufgaben, die man dem Spieler stellen kann. In Adventurespielen heißen Aufgaben, die vom Spieler bewältigt werden müssen „Rätsel“ oder „Puzzle“. Typische Rätselarten sind:

Die verschlossene Tür

Wie bereits beschrieben muss der Spieler den passenden Schlüssel für die Tür finden. Diese Tür kann alles sein, was den Weg versperrt: Ein Felsen, der weggesprengt werden muss, ein Wächter, den man bestechen kann. Im zweiten Akt von „Edna bricht aus“ versperrt zum Beispiel ein Fluss den Weg. Der Schlüssel ist das Abschleppseil, das Schlüsselloch ist das Brückengeländer. Also muss bisweilen auch das Schlüsselloch gefunden werden. Wichtig ist nur dies: Die Tür ist uns bekannt, aber der Schlüssel fehlt.

Der verborgene Schatz

Der Spieler braucht einen Gegenstand, der vergraben, leicht zu übersehen, von etwas überdeckt oder zwischen vielen anderen Gegenständen versteckt ist. Das kann auch ein Ausgang wie der Lüftungssachacht in Ednas Zelle sein.

Der Tauschhandel

Ein NSC hat einen Gegenstand, will ihn aber nur im Tausch rausgeben. Edna tauscht in „Eba“ zum Beispiel die „Fliege in Ohrenschmalz“ bei Professor Nock gegen einen „Kleiderbügel“. Was der NSC haben möchte verrät er uns meistens im Gespräch. Auch Gegentände wie der Kaffeeautomat können einen Tauschhandel mit uns eingehen. Der Kaffeeautomat fordert sogar zwei Objekte von uns: Den leeren Becher und die Münze. Im Austausch bekommen wir Kaffee.

Das Labyrinth

Hat ein Bereich des Spiels einen Ausgang, der zwar frei zugänglich, aber durch den unübersichtlich verzweigten Pfad dorthin schwer zu erreichen ist, so spricht man von einem Labyrinth.

Soziale Interaktion

Auch die Dialoge haben bisweilen Labyrinthcharakter, nämlich dann, wenn es die Aufgabe des Spielers ist, aus dem Geäst an Auswahlmöglichkeiten die eine zielführende herauszufinden. Solche Dialog-Labyrinthe sollen in der Regel nicht durch willkürliches herumprobieren gelöst werden, sondern dadurch, dass man die Eigenheiten der Charaktere analysiert und nach den so gewonnenen Erkenntnissen versucht, die richtigen Gesprächsthemen herauszufinden, die den NSC dazu veranlassen, auf die gewünschte Weise zu reagieren.

Kombinationsrätsel

Ein Kombinationsrätsel liegt dann vor, wenn es nötig ist, zwei oder mehr Gegenstände zusammenwirken zu lassen, um einen Effekt zu erzielen. Ein hochgradiges Kombinationsrätsel findet sich im zweiten Teil des ersten Aktes am Speiseplan. Hier kann man fünf Zettel beliebig auf fünf Plätze einer Liste verteilen, um den Patienten in der Kantine verschiedene Speisen zuzuordnen. Nur in einer Kombination ist es uns möglich, eine Essensschlacht zu provozieren.

Mechanische Rätsel

Der Spieler wird mit einem Mechanismus konfrontiert, dessen Funktionsweise er zunächst verstehen muss, um damit einen gewünschten Effekt zu erzielen. Solche Mechanismen bestimmen die Spielewelt von „Myst“. In „Edna bricht aus“ sind sie verhältnismäßig selten. Die Armaturen des Lieferwagens im zweiten Akt sind ein seltenes Beispiel – auch wenn es sich hier um einen Mechanismus handelt, der jedem Autofahrer vertraut ist.

Logikpuzzle

Bei einem Logikpuzzle handelt es sich um ein abstraktes Problem, dass durch logische Überlegung zu lösen ist. Edna bekommt ein solches Rätsel im Zuge einer Gruppentherapie im ersten Akt geboten.

Um in einem Adventure ein Ziel zu erreichen ist es oftmals nötig, mehrere solcher Rätsel in beliebiger Reihenfolge zu lösen.

3.8 Hinweise

In manchen Phasen des Spiels kann Edna eine recht große Anzahl von Räumen bewandern. Hat sie im ersten Akt zum Beispiel Zugang zum Wäschelift erlangt, Hoti und Moti „befreit“ und deren Bettlaken verdreckt, so taucht das Laken im Keller in der Wäscherei auf. Jetzt kann Edna über das Wäscheliftsystem das ganze Haus erreichen, denn vom Keller aus kommt sie über das Treppenhaus bis zurück unters Dach und in Dr. Marcels Büro.

Besonders jetzt ist es wichtig, Hinweise zu streuen. Wo muss Edna als nächstes hin? Was ist jetzt ihre Aufgabe? Eine Möglichkeit, dem Spieler einen Hinweis zu geben besteht in der Illustration. Der Illustrator ist in der Lage, Objekte so zu zeichnen, dass sie geradezu nach Aufmerksamkeit schreien. Ein Beispiel dafür ist das Schloss an der Hintertür. Es ist so groß und haptisch gezeichnet, dass man es unmöglich übersehen kann. Das große Schlüsselloch darin lädt geradezu dazu ein, einen Schlüssel darin umzudrehen. Wir brauchen also einen Schlüssel für diese Tür.

Abb. 3.8.: Szene aus „Edna bricht aus“: Die Hintertür.

Weitere Hinweise können in jeder Art von Interaktionsskript hinterlegt sein. Es wäre zum Beispiel möglich, Edna jedes mal, wenn sie versucht einen Gegenstand zu nehmen, der sich nicht aufsammeln lässt, folgenden Satz sagen zu lassen:

```
EDNA
Diesen Gegenstand hätte ich nicht halb so gerne wie einen
Schlüssel für die Hintertür.
```

Diese Möglichkeit ist allerdings nicht sehr praktikabel. Damit Edna nicht bereits diese Reaktion von sich gibt, wenn sie die Hintertür noch gar nicht gesehen hat, müsste man beim Betreten des Raumes mit der Hintertür ein Skript auslösen, das allen Raumobjekten, die diesen versteckten Hinweis haben sollen, die oben genannte Reaktion über einen entsprechenden Skriptbefehl zuweist. Wenn sie später den Generalschlüssel erlangt, müsste man diese Skripte alle wieder über ein ähnliches Skript zurücktragen.

Die dritte und naheliegendste Methode, um Hinweise zu streuen, sind Dialoge.

3.9 Dialogführung

Bei interaktiven Dialogen mit vorgefassten Auswahloptionen handelt es sich bereits um kleine, verzweigte Narrationen. Es ist eine metalineare Handlung mit den Charakteristika einer Baum-Struktur.

An jedem Knotenpunkt eines Dialogs spaltet sich die Handlung in beliebig viele Zweige auf. Bei „Edna bricht aus“ ist die Zahl der möglichen Antworten pro Auswahl auf zehn begrenzt. Da aber bei Dialogen mit der Eba-Engine jede Auswahl nicht zwangsläufig direkt zum nächsten Knoten führt, sondern ein ganzes Skript auslöst, welches die unterschiedlichsten Effekte im Spiel erzielen kann, ist es auch möglich, dass die Auswahl einer Option zum Beispiel drei weitere Antworten an verschiedenen Knotenpunkten der Baumstruktur dieses oder eines anderen Dialogs „freischaltet“ und dann erst den nächsten Knotenpunkt anläuft. Die dadurch entstehenden möglichen Zusammenhänge lassen sich kaum noch sinnvoll in einem Diagramm abbilden.

Der im Folgenden vorgestellte Dialog verfügt über fünf Knotenpunkte auf drei Ebenen. Im Diagramm werden nur die Wege dargestellt, die zu anderen Knoten, nicht diejenigen, die zurück zum ersten Knoten führen oder das Gespräch beenden. Eine gestrichelte Linie bedeutet, dass die Option nicht von Anfang an verfügbar ist.

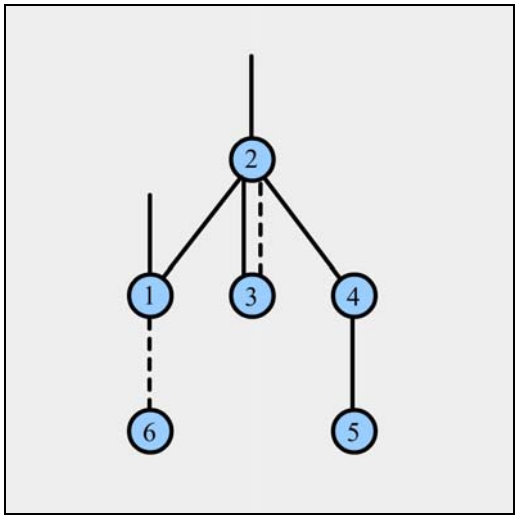

Abb. 3.7.: Vereinfachtes Baum-Diagramm zu einem Dialog aus „Edna bricht aus“.

Es handelt sich um das Gespräch mit dem Wächter namens „Blase“, den Edna im ersten Akt im Monitorraum trifft. „Blase“ sitzt vor einer riesigen Monitorwand. Hinter ihm

steht ein leerer Wasserspender und ein Papierkorb mit benutzten Pappbechern. Am Gürtel des Wächters hängt der begehrte Generalschlüssel.

Der Dialog kann auf zwei verschiedene Weisen beginnen, an Knoten eins oder an Knoten zwei. Betritt Edna zum ersten Mal den Raum, so will der Wächter wissen, wer da hinter ihm steht. Der Spieler hat in der folgenden Auswahlliste die Möglichkeit, ihm die Wahrheit zu sagen oder ihn anzulügen. Die blass gedruckten Optionen sind noch nicht aktiv. Sie sind erst später verfügbar.

```
            BLASE
      Bist Du das, Neuer?


Auswahl 1
1)    Ja, ich bin der Neue. Wer sonst.
2)    War ein Scherz. Ich bin natürlich doch der Neue.
3)    Um ehrlich zu sein bin ich eine entlaufene Irre.
4)    Ich bin niemand. Ignorier mich einfach.
5)    Schon mal was von "Thor, Gott des Donners" gehört?
6)    Ich bin ein Geräuschimitator, der sich auf Wasser spezialisiert
      hat.
7)    Ich bin schon weg.
```

In jedem Fall wird der Wächter Edna für seinen neuen Kollegen halten, selbst wenn sie ihm die Wahrheit sagt. Er beendet nun das Gespräch damit, dass er Edna bittet, ihn nicht länger zu stören.

```
            EDNA
      Um ehrlich zu sein bin ich eine entlaufene Irre.

            BLASE
      Entlaufene Irre. Jaja. Hör mal, Neuer: Wenn Du nur zum
      herumblödeln gekommen bist, kannst Du auch genauso wieder gehen.
```

Gleichzeitig wird für die spätere Benutzung dieser Auswahlliste Option 1) deaktiviert und Option 2) aktiviert.
Missachtet der Spieler die Bitte des Wächters und beginnt das Gespräch über den Befehl „Rede mit Wächter“ erneut, so landet man an einem anderen Teil des Dialogs, nämlich an Knotenpunkt 2.

```
            EDNA
      Ich störe ja nur ungern, aber ...


            BLASE
      Fass Dich kurz.


Auswahl 2
1)    Ich bin nicht der Neue.
2)    Ist das da der Generalschlüssel?
3)    Zurück zum Generalschlüssel...
4)    Was guckst Du da?
5)    Hast Du etwa das ganze Wasser getrunken?
6)    Ich lass Dich dann mal wieder allein.
```

Als erste Option wird dem Spieler hier angeboten, die Aussage über Ednas Identität zu revidieren, was uns zurück zu Knoten 1 bringt. Die zweite Option bringt uns ebenfalls auf die nächste Gesprächseben, nämlich zu Knoten 3. Hier kann der Spieler versuchen, dem Wächter den Generalschlüssel „abzuschwatzen".

```
            EDNA
      Ist das da der GeneralschlÃ¼ssel?


            BLASE
      Und ob.


            EDNA
      Und mit dem bekommt man alle TÃ¼ren hier auf?


            BLASE
      Alle bis auf eine, ja.


Auswahl 3
1)    Kann ich ihn haben? Ich habe meinen verlegt.
2)    Kann ich ihn haben? Hulgor schickt mich, ihn zu holen.
3)    Kann ich ihn haben? Dr. Marcel sagt, ich soll ihn bekommen.
4)    Kann ich ihn haben? Ich muss ins Lager, das leere Wasser
      auffÃ¼llen.
5)    Kann ich ihn haben? Ich komme sonst nicht hier raus.
6)    Welche TÃ¼r bekommt man denn damit nicht auf?
7)    Egal. Ich will ihn gar nicht.
```

Bereits durch das Lesen der möglichen Optionen erfährt man, wie wichtig dieser Schlüssel ist. Probiert man sie der Reihe nach durch, so erhält man den Hinweis, dass dieses Exemplar das einzige ist, das sich in unserer Reichweite befindet. Option 4) bietet zusätzlich noch einen Hinweis auf die andere grosse Thematik in diesem Dialog,

die volle Blase des Wächters. Option 6) bietet einen versteckten Hinweis auf ein ganz anderes Rätsel: Doktor Marcel hat einen geheimen Raum im zweiten Stock, der Schlüssel hierzu befindet sich irgendwo in seinem Büro. Dies ist ein Beispiel dafür, wie man den Spieler mit versteckten Hinweisen an kommende Aufgaben erinnern kann. Option 7) führt zurück zur Auswahlliste 2. Hier hat sich jetzt Option 2) deaktiviert. Die Antwort auf diese Frage hat Edna bereits erhalten. Um trotzdem zum Knotenpunkt 3 zurückkehren zu können ist jetzt Option 3) aktiviert worden, die diese Funktion übernimmt. Die veränderte Liste sieht so aus:

```
Auswahl 2
1)    Ich bin nicht der Neue.
2)    Ist das da der Generalschlüssel?
3)    Zurück zum Generalschlüssel...
4)    Was guckst Du da?
5)    Hast Du etwa das ganze Wasser getrunken?
6)    Ich lass Dich dann mal wieder allein.
```

Die Reaktion auf Option 4) ändert sich je nachdem, ob der Wächter gerade zur Monitorwand oder zum Fernseher gedreht ist. Der Fokus seiner Aufmerksamkeit kann vom Spieler beeinflusst werden, indem er die Fernsehantenne auf dem Dach verdreht. Dies ist also ein gutes Beispiel dafür, dass sich die Dialogführung auch von ausserhalb des Gesprächs beeinflussen lässt.

Option 5) führt an einen weiteren Knotenpunkt. Die durch diese Option erreichbare Auswahlliste 4 beschäftigt sich damit, dass der Wächter den ganzen Wasserspender alleine leergetrunken hat und dringend auf die Toilette muss. Hier deutet sich eine ganz neue Möglichkeit an, das Rätsel um den Generalschlüssel zu lösen. Wenn der Wächter erst einmal auf der Toilette ist, ist es vielleicht leichter, an das begehrte Objekt heranzukommen. Um den Spieler darauf zu stossen, dass die volle Blase des Wächters für ihn von Bedeutung ist, wird auf diesen Ast des Dialogs durch eine weitere Vezweigung ein Fokus gesetzt.

```
      EDNA
Hast Du etwa das ganze Wasser getrunken?

      BLASE
Erinner mich nicht daran.
Ich müsste eigentlich längst auf Klo. Aber ich soll meinen
Posten nicht verlassen. Als wenn bei den Irren je was passieren
würde! Gottseidank bin ich was das anbetrifft sehr
strapazierfähig. Man nennt mich nicht umsonst "Blase".
```

```
Auswahl 4
1)    Geh doch einfach. Ich pass solange auf.
2)    Wow! Das GAAANZE Wasser?
3)    So lange zurÃ¼ckzuhalten ist ungesund!
4)    Aprospos Wasser...
5)    Ich geb auf. Deiner Blase bin ich nicht gewachsen.
```

Mit dieser Auswahl wird der Spieler geradezu eingeladen, den NSC weiter zu provozieren.

```
      EDNA
Wow! Das GAAANZE Wasser?

      BLASE
Erinner mich nicht daran. BITTE.
```

Hier wird der Dialog schließlich auch noch dazu benutzt, unsere Protagonistin Edna zu Charakterisieren: Der Wächter bittet sie inständig, nicht mehr über das Thema „Wasser“ zu reden. Er muss wirklich dringend. Edna ist jedoch eine sehr aufsässige Person. Der Spieler ist eingeladen, sich in diese Rolle hineinzufühlen, indem er Edna unerbittlich fortfahren lässt und Option 4) wählt:

```
      EDNA
Aprospos Wasser ...
```

```
Auswahl 5
1)    Ich war in den Ferien am Pazifik.
2)    Wusstest Du, dass Sie auf dem Mars Spuren von Wasser gefunden
      haben?
3)    Ich hab gehört, es soll Regen geben.
4)    Wenn ich nach Hause komme, geh ich erstmal baden.
5)    Mein Cousin ist bei der Feuerwehr.
6)    Hast Du Waterworld mit Kevin Costner gesehen?
7)    Ich geb auf. Deiner Blase bin ich nicht gewachsen.
```

Wählt man zum Beispiel Option 2) eröffnet sich folgender Dialog:

```
      EDNA
Wusstest Du, dass Sie auf dem Mars Spuren von Wasser gefunden
haben?

      BLASE
Hör endlich auf über Wasser zu reden!

      EDNA
Furchen von Rinnsalen, die jahrzehntelang das Gestein umspült
haben.
```

```
      BLASE
Stop it!

      EDNA
Ganze Bäche soll es da gegeben haben. Reissende Ströme.
Die hektoliterweise Wassermassen durch enge Schluchten gepresst
haben.

      BLASE
Gleich ist es soweit. Gleich bring ich dich um!
```

Auch wenn der Spieler mit diesen angebotenen Optionen den Wächter an den Rand eines Nervenzusammenbruchs treibt, ist er schließlich doch gezwungen Option 7) zu wählen. Damit kommt er zurück zu Knoten 2. Als verstecktes Gimmick öffnet sich nun aber auch Option 6) in Knoten 1. Kommt der Spieler also noch einmal auf Ednas Identität zu sprechen, so kann er durch Anwahl dieser Option zum letzten Knotenpunkt 6 gelangen und erhält damit eine weitere Gelegenheit, dem Wächter auf den Geist zu gehen.

```
6)    Ich bin ein Geräuschimitator, der sich auf Wasser spezialisiert
      hat.


Auswahl 6

1)    Gluck. Gluck. Gluck. Gluck. Gluck. Gluck.
2)    Psssssssssssssssssssssssssssssssssss...
3)    Lunk. Lunk. Lunk. Lunk. Lunk. Lunk. Lunk.
4)    Blubblubbelblubbblubbblubbblubbeldiblubb.
5)    *Blob*blob*blob*blob*blob*blob*blob*blob.
6)    Wuuuscccccchh!
7)    Die Vorstellung ist beendet.
```

Die Letzte Option beendet das Gespräch.

3.10 Die Interaktivitätsillusion

Ein Adventurespiel wie „Edna bricht aus“ ist nicht in der Lage, zur Laufzeit des Programms neue Reaktionen zu generieren. Es ist nicht kreativ. Jede mögliche Aktion, die sich der Spieler einfallen lässt, hat eine Adresse in der darunterliegenden Datenstruktur. Bei „Edna bricht aus“ ist diese Adresse die ID der Skript-Tabelle in der Datenbank.

Versucht der Spieler über das Interface mit der Spielewelt zu interagieren, so entsteht nichts Neues. Es werden maximal bereits vorhandene Zustände umgeschaltet.

Wenn der Spieler beispielsweise Edna den Befehl „Nimm Lexikon“ ausführen lässt, deaktiviert sich das Raumobjekt „Lexikon“ und das vorher inaktive Inventarobjekt „Lexikon“ wird aktiviert. Beide Objekte wie auch das Skript, das diesen Prozess ausführt, sind von Beginn des Spiels an vorhanden und warten nur darauf aktiviert, deaktiviert oder ausgeführt zu werden.

Der Skripter hat dafür Sorge zu tragen, dass so viele Interaktionen wie möglich „persönlich“ behandelt werden, also nicht durch Standardreaktionen wie die Textausgabe „Das Klappt so nicht.“. Man spricht auch davon, dass Ereignisse „abgefangen“ werden. Hinterlassen unwichtige oder falsche Aktionen einen sichtbaren Effekt in der Spielewelt (aktivieren oder deaktivieren sich also dadurch sichtbare Objekte), so scheint es dem Spieler, als hätte er gestalterisch auf die Welt eingewirkt.

In „Edna bricht aus“ kann der Spieler an vielen Orten solche Spuren hinterlassen, ohne dass die Veränderungen einen Nutzen für den Spielfluss haben. Als Beispiel ist es Edna möglich, in ihrer Einzelzelle neunzehn verschiedene Polster mit dem abgebrochenen Stuhlbein zu zerschlitzen, obwohl nur eines davon der Lösung des Rätsels dient. Edna weiss von sich aus nicht, bei welchem Polster diese Behandlung sinnvoll ist, darum wäre es unlogisch, wenn es ihr nur bei einem möglich wäre. Beim Beobachten der ersten Testspieler fiel mir auf, dass sie auch dann die restlichen Polster zerstörten, wenn ihnen bereits klar war, welches Polster das richtige sein musste. Der Spieler ist in der Regel glücklich, wenn er feststell, dass seine Aktionen eine Veränderung auf dem Bildschirm hervorrufen. Es entsteht dadurch bei ihm die Illusion, dass es sich um eine Welt mit physikalischen Gesetzen handelt, mit der er frei interagieren kann.

Diese Interaktivitätsillusion findet sich im Großen auch bei der Bewältigung des Quests wieder. Je mehr Sackgassen es im Labyrinth gibt, umso mehr hat der Spieler den Eindruck, dass der Weg, den er hindurch gefunden hat, seine eigene Kreation ist, seine eigene schöpferische Leistung.

4 Umsetzung

4.1 Programmierung

4.1.1 Anforderungen

„Edna bricht aus“ ist ein klassisches Point-And-Click-Adventure. Das bedeutet, dass der Spieler alleine durch Auswählen einzelner Bildbereiche per Mausklick Ereignisse auslösen können soll. Auf diese Weise steuert er einen Avatar (in diesem Fall ist es die Hauptfigur Edna), durch eine Welt, die sich aus einzelnen, Bildschirmfüllenden „Räumen“ (auch „Szenen“ oder „Screens“) zusammensetzt.

Während des Spiels teilt sich der Bildschirm eines Point-And-Click-Adventures klassischer Weise in zwei Bereiche: Im Hauptteil des Bildes sehen wir den „Raum“ oder die „Szene“. Hier befindet sich für gewöhnlich auch die Hauptfigur (SC = Spielercharakter), alle Nebencharaktere (NSC = Nicht Spieler Charakter) und sämtliche Gegenstände, mit denen Edna interagieren kann. Jedem Raum muss genau eine Hintergrundgrafik und eine Hintergrundmusik zugeordnet werden können. Am unteren Bildschirmrand befindet sich das Interface (GUI = Grafisches User Interface).

„Edna bricht aus“ benutzt insgesamt 7 verschiedene GUIs, die je nach ihrer Funktion im Spiel benannt sind. Es gibt das Edna-GUI, das Harvey-GUI, das Edna_jung-GUI, das Skript-on-Click-GUI, das Drag-on-Click-Skript-on-Drop-GUI, das Zengarten-GUI und das Start-GUI. Beschreibt man alle Funktionen, die das fertige Spiel beinhalten soll in all ihren Erscheinungen und Mechaniken, so erhält man eine genaue Aufstellung der Aufgaben, die die Programmierung zu lösen hat. Darum werde ich an dieser Stelle die Funktionsweise des Edna-GUIs in allen Einzelheiten erläutern. Zusammen mit den Funktionen der anderen GUIs ergibt sich hieraus die Anforderungsliste an die Programmierung.

4.1.1.1 Das Edna-GUI

Das Edna-GUI ist, wie auch das Edna_Jung-GUI, ein Klon des SCUMM-Story-System von Lucasfilm Games, den heutigen LucasArts. Bei diesem System steuert der Spieler

die Hauptfigur (in diesem Fall Edna), indem er eine Anweisung aus einzelnen Satzfragmenten zusammensetzt. Der Anweisungstext wird in der „Kommandozeile“ sichtbar gemacht. Ist eine Anweisung vollständig, so reagiert das Programm, indem es Edna die gewünschte Aktion ausführen lässt.

Dementsprechend teilt sich das Edna-GUI in drei Bereiche: Links unten befinden sich die vier Verben. Dabei handelt es sich um Befehle, mit denen wir Edna steuern können. Zur Auswahl stehen „Ansehen“, „Nehmen“, „Benutzen“ und „Reden“. Es soll Edna ermöglicht werden, mit jedem beliebigen NSC oder Gegenstand (ausser Ausgängen) in einem dieser vier Modi zu interagieren. Bei Mouseover aktivieren sich die Schaltflächen und der Kommandotext wird in der Kommandozeile dargestellt. Klickt man auf eine Schaltfläche, zum Beispiel „Benutzen“, wird sie als gedrückt dargestellt. Die Textausgabe in der Kommandozeile ist nun resident und bleibt erhalten, auch wenn der Mauszeiger die Schaltfläche wieder verlässt.

Abb. 4.1.: Bildschirmaufteilung von „Edna bricht aus“.

Die Kommandozeile ist der zweite Bereich des Edna-GUI und befindet sich direkt über den Befehlen. Es handelt sich um eine linksbündige Textzeile zur Ausgabe von zusammengesetzten Kommandotexten. Der dritte Bereich ist das Inventar. Es befindet

sich unten rechts, neben den Befehlsknöpfen. Unter dem Inventar-Knopf versteckt sich ein 5 x 7 Felder grosses Raster. In diesem Bereich werden Gegenstände abgelegt, die Edna im Verlauf des Spieles aufsammelt („Items“). Das Inventar öffnet sich bei einem Linksklick auf die Inventar-Schaltfläche. Klickt man einmal auf das „Schloss“-Symbol in der unteren rechten Ecke, so öffnet sich das Inventar bereits bei einem Mouseover. Ein weiterer Klick auf das Schloss stellt die alte Funktionalität wieder her. Hat Edna mehr als 35 Gegenstände aufgesammelt, so lässt sich der 5x7 Felder große Bereich durch Pfeiltasten zeilenweise nach oben und nach unten verschieben. Zu Beginn des Spiels besitzt Edna noch keine Items. Nur ihr Stofftier Harvey liegt im Inventar.

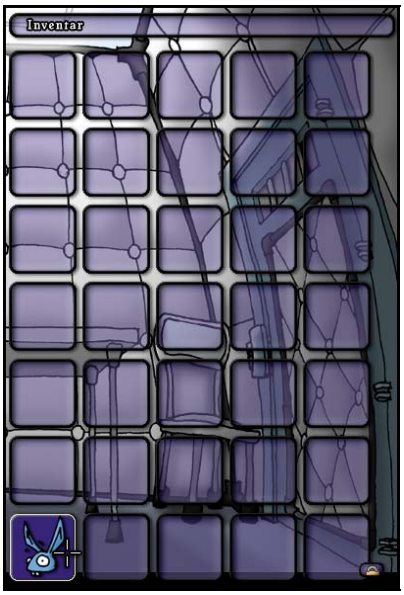

Abb. 4.2.: Das Inventar.

Ein Klick auf Harvey kombiniert ihn nun mit dem zuvor ausgewählten Befehl. „Benutze Harvey“ ist in diesem Fall noch keine vollständige Anweisung, denn Inventargegenstände sind dafür da, dass man sie mit Gegenständen in der Szene benutzt. Dementsprechend lautet der Text in der Kommandozeile auch „Benutze Harvey mit“. Verlassen wir mit dem Mauszeiger das Inventar, so fährt es wieder zurück in seinen Urzustand. Der Text in der Kommandozeile bleibt jedoch bestehen.

Klickt man jetzt auf einen Gegenstand, zum Beispiel den Tisch, erscheint dessen Bezeichnung ebenfalls in der Kommandozeile hinter dem ausgewählten Befehl. „Benutze Harvey mit Tisch“ ist ein vollständiger Befehl, den das System als Anweisung erkennt. Dieser Anweisung ist eine Aktion zugeordnet, die nun ausgeführt wird.

Mit dieser Methodik lassen sich nun alle Gegenstände im Raum ansehen. Einige davon kann Edna nehmen, dann landen sie im Inventar an der nächsten freien Position. Edna

kann ausserdem mit Objekten reden, was hauptsächlich dann Sinn macht, wenn es sich bei den Objekten um NSCs handelt. Gegenstände aus dem Inventar lassen sich mit beliebigen anderen Gegenständen benutzen. Raumgegenstände lassen sich auch ohne weitere Kombination benutzen (was zum Beispiel bei Objekten wie Lichtschaltern wichtig ist). „Edna bricht aus“ beinhaltet mehr als 2000 Raumobjekte. Über die Hälfte davon sind „Raumobjektinteraktionen“ (ROI), von denen jedem Einzelnen vier verschiedene Skripte für die direkte Benutzung über die Befehle zugeordnet sind („Ansehen-Skript“, „Nehmen-Skript“, „Benutzen-Skript“, „Reden-Mit-Skript“). Darüber hinaus lassen sich fast alle ROIs mit einer Vielzahl der insgesamt 77 Inventargegenstände benutzen, die Edna im Verlauf des Spiels aufsammeln kann. Ohne ausgefeilte Datenstruktur wäre eine solche Fülle an abzufangenden Ereignissen nicht zu handhaben. Darum setzt „Edna bricht aus“ auf einer HSQL-Datenbank auf, in der alle Gegenstände mit ihren Parametern und Referenzen zu anderen Objekten festgehalten sind. Die Datenbank und ihre Struktur werden in Abschnitt **4.1.4** behandelt.

Ist ein Befehl komplett, geht Edna zunächst zu dem entsprechenden Raumobjekt, bevor sie die Anweisung behandelt. Wenn nichts weiter ausgewählt wurde interpretiert das System einen Klick als „Gehe zu“-Befehl. Auch in dem Fall versucht Edna also, den angeklickten Punkt zu erreichen. Dazu ist es natürlich nötig, dass eine Entscheidung zwischen begehbarem und nicht begehbarem Bereich getroffen wird. Diese Bereiche sind je nach Raum unterschiedlich. Darum ist zusätzlich zur Hintergrundgrafik für jeden Raum ein boolesches Array in Form eines konvertierten Schwarz-weiss-Gifs hinterlegt, das anzeigt, welche Pixel für Edna begehbar sind und welche nicht (WAM = „Walkable Area Map“).

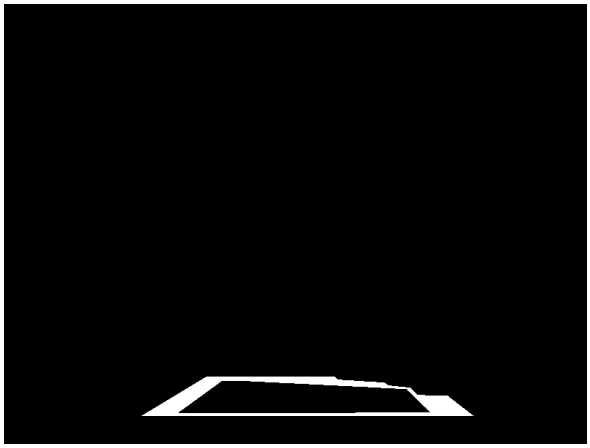

Abb. 4.3.: Die „Walkable Area Map“ (WAM) zu Ednas Zelle.

Da mitunter nicht begehbare Bereiche inmitten dieser WAM auftauchen können, wenn zum Beispiel ein Tisch im Wege steht, muss Ednas Bewegung einem Algorithmus folgen, der sie nacheinander die günstigsten Punkte ansteuern lässt, um das Objekt zu umrunden.

Eine weitere Schwierigkeit im Zusammenhang mit der Bewegung im Raum stellt die Perspektive dar. Jedem Raum sind deswegen Parameter zugeordnet, die es erlauben, Edna je nachdem wie weit sie sich in der Tiefe des Bildes befindet zu skalieren und ihre Geschwindigkeit anzupassen. Neben der Geh-Animation sind ihr als Hauptcharakter noch drei weitere Standard-Animationen zugeordnet. Es gibt eine Animation für den Fall, dass sie wartet, eine für den Fall, dass sie spricht und eine für den Fall, dass Harvey spricht, den sie immer dabei hat. Edna kennt vier Himmelsrichtungen. Jeder der oben genannten Aktionsmodi („walk“, „talk“, „wait“, „think“) braucht also vier Animationen, eine für jede Himmelsrichtung. Ausserdem gibt es noch eine unbegrenzte Anzahl an Zusatzanimationen, die im Bedarfsfall dem Playersprite zugeordnet werden kann. So lässt der Aktionsmodus 5 Edna nach einem Gegenstand greifen und im Aktionsmodus 6 bückt sie sich, um etwas vom Boden aufzuheben. Die Zusatzaktionsmodi müssen natürlich ebenfalls die Himmelsrichtungen kennnen.

Hat Edna einen Befehl bekommen und ist am Zielpunkt angelangt, so muss eine beliebig lange Liste von Anweisungen ausgeführt werden. Solche Anweisungslisten nennt man „Skript“. Sie sind meist in einer einfachen Skriptsprache formuliert. Auch für „Edna bricht aus“ haben wir eine solche Skriptsprache entwickelt. Der EBA-Skriptinterpreter kennt 39 Kommandos, mit denen sich alle Ereignisse behandeln lassen, die zur Darstellung einer selbstablaufenden Handlung und zum Skriptgesteuerten modifizieren von Spielzuständen benötigt werden. So kann man per Skript Edna sprechen lassen („say(Hallo Harvey!)“), eine Spezialanimation ausführen („animatesc(05,0500)“) oder einen Gegenstand deaktivieren („inactivate(10160401)“). Selbst einfache If-Abfragen sind mit dem EBA-Skriptinterpreter möglich. Eine vollständige Auflistung der Skriptbefehle sowie eine detaillierte Beschreibung der hinterlegten Funktionen findet sich im Anhang auf der CD. In Abschnitt **4.2** wird der gesamte Vorgang des „Skriptens“ anhand eines ausführlichen Beispiels veranschaulicht. Das Skripten beinhaltet sowohl das Anlegen der Objekte in der Datenbank, als auch das Schreiben der Skripte.

Ein Druck auf die [.]-Taste beendet eine Animation oder eine Textausgabe. So lassen sich bereits bekannte Reaktionen abkürzen.

Wird Edna in einem Dialog vor eine Wahl gestellt, so verschwinden die GUI-Komponenten. An ihre Stelle tritt eine Liste aus möglichen Dialogoptionen (CL = „Choicelist“). Hier stehen dem Spieler bis zu zehn mögliche Fragen oder Antworten zur Verfügung, um das Gespräch zu steuern. Der Dialogbildschirm passt sich in der Höhe der Anzahl der Auswahlmöglichkeiten an. Fährt man mit der Maus über eine Dialogoption, so wird diese in einer anderen Farbe dargestellt. Ein Linksklick auf die Auswahl deaktiviert die Choicelist, aktiviert die GUI-Komponenten und startet ein Skript.

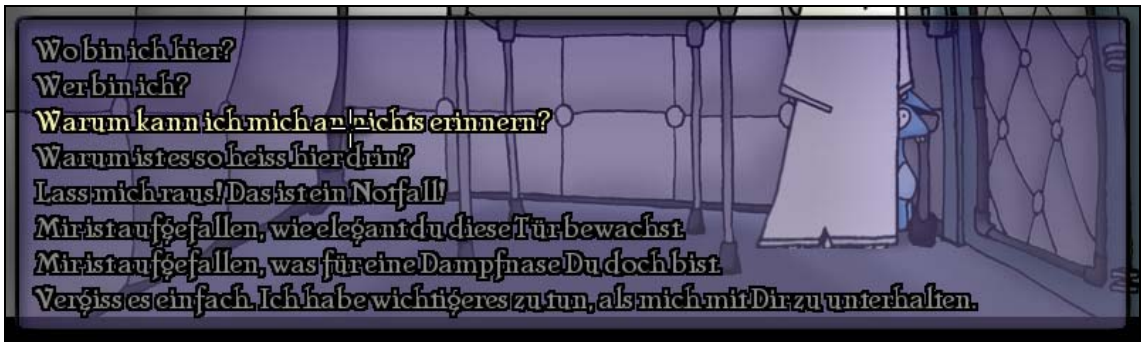

Abb. 4.4.: Eine „Choicelist“ mit acht Optionen.

Ein Druck auf die [Esc]-Taste öffnet das Hauptmenü. Hier lässt sich über entsprechende Schaltflächen ein neues Spiel laden, das laufende Spiel abspeichern oder das Spiel beenden. Es stehen 9 Speicherplätze zur Verfügung, die im rechten Bildbereich angezeigt werden. Zu jedem Speicherstand wird eine Bilddatei abgespeichert, die hier angezeigt wird, um den Speicherstand zu identifizieren.

Abb. 4.5.: Das Hauptmenü.

Ein Klick auf diesen Bereich führt dazu, dass unten rechts zwei weitere Knöpfe angezeigt werden: „Okay“ und „Cancel“. Je nachdem, ob man sich im „Laden“- oder „Speichern“-Modus befindet, läd oder speichert der Druck auf „okay“ das ausgewählte Spiel, während „Cancel“ den Vorgang abbricht.

Befindet man sich mit der Maus über einem Ausgang, so wandelt sich der Mauscursor in eine Hand. Bei einem Klick auf das Objekt wird nun ein Raumwechsel ausgeführt, sobald Edna dort angekommen ist. Es kann allerdings auch optional ein Skript hinterlegt sein, das über den Raumwechsel hinweg bestehen bleiben muss. Die Veränderungen im Raum und in Ednas Inventar müssen bei einem Raumwechsel bestehen bleiben. Im Zielraum kann auch ein ganz anderes GUI aktiv sein. Einige GUIs sind sehr speziell und nur in wenigen Räumen gültig. Das Edna_jung-GUI verhält sich analog zum Edna-GUI. Nur das Harvey-GUI soll hier noch kurz erläutert werden:

4.1.1.2 Das Harvey-GUI

Das Harvey-GUI unterscheidet sich sehr vom Edna-GUI und wird nur in den Rückblenden benutzt. Harvey ist nur eine Halluzination, also kann er nichts anfassen oder benutzen. Dafür kann er Edna behilflich sein, indem er Informationen an Orten einholt, die für sie nicht zugänglich sind. Diese Informationen zieht sich Harvey in Form von „Topics“ per Drag und Drop aus den Objekten auf seine „Topicleiste“, einen Bereich aus 14 Feldern am unteren Bildrand. Zieht er ein Topic auf die junge Edna, die sich mit ihm in einem Raum befindet, so wird ein Skript ausgeführt. Harvey spricht nun mit Edna über seine Beobachtung. Mit der rechten Maustast kann Harvey ein weiteres Skript auf eine Raumobjektinteraktion oder ein Topic in seiner Leiste ausführen.

Abb. 4.6.: Das Harvey-GUI.

4.1.2 Wahl der Waffen

AGS

Für das Demo, das im Rahmen eines Hochschulprojektes bereits im Oktober 2005 fertiggestellt wurde, hatten wir ein Development-Kit namens „Adventure Game Studio“ (AGS) benutzt, das bereits eine Vielzahl von Funktionen anbietet, die nötig sind, um ein klassisches Adventure-Spiel zu produzieren. Leider sind die Möglichkeiten, das Interface nach eigenen Wünschen zu gestalten, beschränkt. Auch die Skript-Engine von AGS bietet nicht alle Funktionen, die zu einer vollständigen Umsetzung meiner Geschichte nötig gewesen wären.
Die Datenmenge, die „Edna bricht aus“ mittlerweile beherbergt, wäre in AGS unter keinen Umständen noch handhabbar gewesen wäre.

JAVA

Es war mein Ziel „Edna bricht aus“ unter marktgerechten Bedingungen zu erstellen. Das hieß auch, dass wir keine Software, Schriftart und Klassenbibliothek verwenden durften, die zur Distribution eine Lizenz erfordert hätten. Um bei der Programmierung nicht von null auf beginnen zu müssen, galt es also eine Klassenbibliothek zu finden, die über eine freie Lizenz verfügte.

Weil sowohl ich als auch Felix Engel und Stefan Hütter, die beide massgeblich an der Programmierung beteiligt waren, bislang hauptsächlich auf Java programmiert hatten, suchten wir zunächst eine offene Klassenbibliothek für Computerspiele in Java. Wir wurden fündig in GTGE – der Golden T Game Engine von Golden T Studios, die im Rahmen des Creative Commons Lizenzvertrags unter Bedingung der Namensnennung der Hersteller zur kommerziellen Nutzung weiterverwendet werden darf.

GTGE

GTGE bietet eine Reihe von Klassen und Methoden zur Erstellung von Computerspielen unter Java, darunter Kollisionsabfragen, animierte Sprites, Timersteuerung, Audiorenderer, einen GameFontManager und natürlich eine Runtime Engine.

Mit der Wahl, GTGE zu verwenden, kauften wir uns die Bedingung ein, für das Laden des Spieles die GTGE-Vogaben zu verwenden. GTGE wurde konzipiert, um einfache

Arcade-Games zu programmieren, nicht aber um die komplexe Spielewelt eines Adventure-Games abzubilden. Die Klassen und Methoden der Engine konzentrieren sich hauptsächlich darauf, Kollisionsabfragen von Sprites in Ereignisse umzusetzen.

Um GTGE für unsere Zwecke dienstbar zu machen, mussten wir eine Reihe von Modifikationen und eine Vielzahl von Erweiterungen vornehmen. Durch diese Änderungen wurde aus der von GTGE bereitgestellten GameEngine die EbaGameEngine. Das Konzept dieser Engine und die Aufteilung der oben beschriebenen Aufgaben in die neu entstandenen 27 Programmklassen werden in Kapitel **4.1.3** geschildert. Die größte Abweichung vom Konzept der GTGE-Engine ist die Anbindung an eine HSQL-Datenbank.

Die Datenbank

Bereits in einem frühen Stadium der Konzeption wurde deutlich, dass der Umfang des Spiels Ausmaße annehmen würde, der eine harte Integration der Spielelemente in den Programmcode ausschloss. Eine andere Lösung musste gefunden werden, die eine dynamische Erweiterung des Spielinhalts zuliess.

Mehr noch: Programm und Inhalt sollten nach Möglichkeit vollständig voneinander getrennt werden, so dass im optimalen Fall das selbe Programm für die Erstellung einer Vielzahl verschiedener Spiele verwendet werden könnte.

Felix Engel fand das Mittel zur Lösung: Eine transportable, lizenzfreie HSQL-Datenbank. Diese Datenbank funktioniert über einfachste CSV-Dateien, in Klartext geschriebene Textdateien, die die Relationen der Datenbank darstellen. Jede Tabelle der Datenbank ist dabei durch eine CSV-Tabelle repräsentiert. Der Inhalt der Datenbank lässt sich somit ändern, indem die einzelnen Einträge direkt in den CSV-Dateien umgeändert werden oder ganze Zeilen gelöscht werden.

Zusammen mit der Skript-Engine bildet diese Datenbank das Tätigkeitsfeld des Skripters. Durch Befüllen der Datenbank entsteht der Spielinhalt. Ihr Herzstück ist die Tabelle „Skript“, in der kurze Anweisungslisten zur Darstellung von selbstablaufenden Handlungen angelegt sind. Diese Anweisungslisten sind über eindeutige Bezeichner mit den Objekten verbunden, die dieses Ereignis später auslösen sollen.

Das Befüllen der Datenbank ist die Hauptaufgabe des Skripters, der sich mit Fug und Recht ausführender Dramaturg des Computerspiels nennen darf. Darum wird später in diesem Kapitel der Aufbau der Datenbank sehr genau beschrieben und an einem Beispiel verdeutlich. Eine vollständige Dokumentation aller Tabellen und Felder findet sich im Anhang auf der CD.

Die EBA-Skriptengine

Die wichtigste Tabelle in der Datenbankstruktur ist die Skript-Tabelle. Hier sind Anweisungslisten hinterlegt, die von einem Objekt der Klasse „EbaSkriptInterpreter“ zeilenweise abgearbeitet werden. Pro Skriptzeile wird eine Anweisung ausgelesen, die eine Animation, einen Raumwechsel, die Änderung eines Parameters in der Datenbank oder andere Ereignisse auslöst.

Um solche Anweisungen eindeutig zu benennen, wird eine einfache Skriptsprache benötigt. Die Skriptsprache, die wir für EBA entwickelt haben, kennt 39 Skriptbefehle, um den Spielfluss zu steuern.

In der Skript-Tabelle bestimmt der Skripter also alle Reaktionen und Ereignisse, die nicht pauschal durch die Engine vorgegeben sind. Jedesmal, wenn Edna etwas sagt, einen Gegenstand aufnimmt oder eine Animation abgespielt wird, ist ein Skript dafür verantwortlich. Die Skripttabelle ist nicht nur das Herzstück des Spiels, sondern für den Dramaturgen auch die Waffe der Wahl, um letztlich die Geschichte zu erzählen. Edna bricht aus hat annähernd 50.000 Zeilen Skript.

Eine genaue Erläuterung der 39 Skriptbefehle, die ihm dafür zur Verfügung stehen, finden sich im Anhang auf der CD. Ein anschauliches Beispiel, wie sie einzusetzen sind, folgt in Abschnitt **4.2**.

4.1.3 Der „Super-Raum“ und seine Klassen

4.1.3.1 Die GTGE-Logik

Die Logik hinter GTGE ist schnell erklärt: Die Main-Methode erzeugt ein Objekt vom Typ GameLoader. Diesem weden über eine Setup-Methode die Parameter übergeben, die nötig sind, um eine Spielfläche einzurichten. Dazu zählen erstens die Bildschirmauflösung, zweitens ein boolscher Wert, der darüber entscheidet, ob das Spiel im Vollbildmodus oder in einem Fenster ausgeführt wird und drittens ein weiterer boolscher Wert, der zwischen zwei grundsätzlichen Methoden zur Speicherpufferung entscheidet. Ausserdem wird dem GameLoader ein Objekt vom Typ GameEngine übergeben, das noch in der Main Methode erzeugt wird.

```
public class Eba {

	public static void main(String[] args) {

		final boolean fullScreenMode = true;
		final boolean bufferStrategy = false;

		GameLoader myGameLoader = new GameLoader();
		myGameLoader.setup(new EbaGameEngine(),
			new Dimension(800, 600), fullScreenMode,
		bufferStrategy);
		myGameLoader.start();
	}
}
```

Diese GameEngine beinhaltet die Methode getGame(int gameID), die nun vom GameLoader mit dem Parameter gameID=0 aufgerufen wird, um das (erste) Spiel zu starten. In dieser Methode wird ein weiteres Objekt erzeugt, das GameObject, welches das laufende Spiel symbolisiert und dem lediglich die GameEngine als Parameter übergeben wird. Über den Aufruf der Methode getGame(int gameID) kann man mittels abweichender Parameter gameID unterschiedliche Spiele starten, nachdem der GameLoader erst einmal das initiale Spiel mit der Nummer „0“ automatisch begonnen hat. Diese Option soll es ermöglichen, in einem Arcade Game verschiedene Orte des Spiels (Den Startbildschirm, das eigentliche Spiel und die Highscore-Tabelle) voneinander zu trennen. Die Mechaniken sind hier so unterschiedlich, dass es sich anbietet, verschiedene GameObject-Klassen zu programmieren, die natürlich alle von GameObject erben müssen.

```
public class EbaGameEngine extends GameEngine {

      public EbaGameEngine() {
            { distribute = true; }
      }

      public GameObject getGame(int gameID) {

            switch (gameID) {
                  case 0:
                  myGameObject =
                        (GameObject)(new MenuObject(this));
                  break;
                  case 1:
                  myGameObject =
                        (GameObject)(new EbaGameObject(this));
                  break;
                  case 2:
                  myGameObject=
                        (GameObject)(new HighScoreObject(this));
            }
            return myEbaGameObject;
      }
}
```

Wie wir gleich sehen werden, verfolgt die EbaGameEngine hier eine etwas andere Philosophie und zweckentfremdet die getGame-Methode für seine eigenen Belange. Zunächst aber soll noch der Aufbau des GameObjects kurz erklärt werden.
Das GameObject ist in drei Bereiche unterteilt: Initialisierung, Aktualisierung und Darstellung. Diese drei Aufgaben werden in den Methoden initResources(), update() und render() bearbeitet.

Die Initialisierung findet in initResources() statt. Hier werden alle Objekte angelegt, die später vom Spiel verwendet werden. Das kann zum Beispiel ein abstraktes Klassenobjekt wie der SkriptInterpreter sein. Vor allem werden aber Sprites initialisiert, die hier auch gleichzeitig in SpriteGroups gebündelt werden. Die Spritegroups werden schliesslich dem Spielfeld hinzugefügt.
Alle gegenständlichen Objekte werden in Eba als Sprites, AnimatedSprites oder über eine Klasse, die von einem dieser beiden Typen erbt angelegt und (im Allgemeinen) zu

einer der folgenden SpriteGroups addiert: BackgroundSprites (Objekte, die sich generell auf der Hintergrundebene befinden. Edna kann nicht hinter ein Objekt aus der BackgroundSpritegroup treten), ObjectSprites (Alle Objekte, die sich entweder vor oder hinter dem SC befinden können und auch der SC selbst finden sich in dieser SpriteGroup wieder), GuiSprites (die Elemente des User Interfaces befinden sich grundsätzlich vor Edna und allen Raumobjekten), TextSprites (Hier befinden sich die Textsprites der visuellen Sprachausgabe) und schliesslich MenuSprites (Das Hauptmenü verdeckt mit seinen Komponenten alle anderen Darstellungen). Auch bei der Choicelist handelt es sich um eine SpriteGroup.

Die Aktualisierung hat ihren Platz in der Methode update(). Hier findet die Ereignisbehandlung statt. Wenn eine Taste gedrückt oder die Maus bewegt wird, so hat das möglicherweise eine Veränderung auf dem Spielfeld zur Folge: Fährt man mit der Maus über einen Gegenstand, so wird zum Beispiel sein Mouseovertext in der Kommandozeile angezeigt. Klickt man mit der linken Maustaste auf einen leeren Punkt, so beginnt der PlayerSprite damit, sich auf den nähsten begehbaren Punkt der WalkableAreaMap zu begeben. Die vielen Unterscheidungen, die hier zu treffen sind, finden sich in einer höchst verzweigen Struktur von if-Abfragen wieder, die den Hauptteil der Update-Methode des GameObjects ausmacht.

Die Render-Methode sorgt schliesslich dafür, dass das gesamte Spielfeld gerendert und somit auf dem Bildschirm dargestellt wird.

Während die initResources-Methode also nur einmal bei der Erstellung des Game-Objects durchlaufen wird, werden die anderen beiden Methoden ständig wieder ausgeführt. Dabei wird zuerst das komplette update abgearbeitet. Gerendert wird erst, wenn alle Veränderungen am Ende des „Update-Zyklus“ eingefangen sind.

```
public class EbaGameObject extends GameObject {

	public EbaGameObject(EbaGameEngine myGameEngine) {
		super(myGameEngine);
	}

	public void initResources() {
		//TODO: Initialisierung
	}
```

```
        public void update(long elapsedTime) {
                //TODO: Aktualisierung
        }

        public void render(Graphics2D g) {
                //TODO: Darstellung
        }
}
```

4.1.3.2 Der Super-Raum

Bei jedem Spiel werden also zunächst alle verwendeten Grafiken angelegt, im Laufe des Spieles werden dann nur noch ihre Eigenschaften (zum Beispiel Koordinaten und Sichtbarkeit) verändert. Das mag für ein Arcade-Spiel praktikabel sein, aber nicht für ein Spiel in dem es über 2000 verschiedene Raumobjekte gibt, von denen aber maximal 30 gleichzeitig in einer Szene aktiv sein können. Wir trafen eine folgenreiche Entscheidung: Für jeden Raum sollte bei Eba eine neue Instanz des GameObjects erstellt und die alte verworfen werden. Dabei sollte sich bei jedem Raum neu entscheiden, welche grundsätzlichen Spiel-Mechanismen zur Ereignisbehandlung, also welches GUI hier zu verwenden ist. Dementsprechend leiteten wir von der bereits modifizierten Klasse EbaGameObject für jedes GUI eine Kindklasse ab. Anstatt also über die GameEngine ein GameObject aufzurufen, wird hier, je nach Raumnummer, eine von sieben möglichen GUI-Klassen instantiiert, die von EbaGameObject erbt. Zu Beginn des Spiels, bei der automatischen Anwahl der Raumnummer „0“, ist dies die Klasse EbaGuiStart, die die Mechanismen des Introscreens umfasst.

Bei jedem Raumwechsel wird also der gesamte Spielinhalt aus dem aktuellen Speicher gelöscht und durch den Inhalt des neuen Raums ersetzt. Der Spielinhalt wird hierbei dynamisch aus einer Datenbank geladen, die einzelnen Komponenten können über entsprechende Einträge in der Datenbankstruktur eindeutig dem Raum, in dem sie auftauchen sollen, zugeordnet werden. Ändert sich im Laufe des Spiels der Zustand eines Objekts, so geschieht das sowohl in seiner Abbildung im aktuellen Raum als auch in der Datenbank. Die Änderungen bleiben deshalb auch bei einem Raumwechsel erhalten. Wenn man also diesem einen Raum in Unterscheidung der möglichen GUIs alle Spielmechanismen beibrächte, alle nötigen Datenbankzugriffe, Skriptaufrufe und Animationsmechanismen, so wäre mit Programmierung dieses einen „Super-Raums“ die Programmierung des ganzen Spieles abgeschlossen. Der gleiche Code könnte benutzt werden, um eine Spielewelt von zwei oder von 200 Räumen zu gestalten.

Voraussetzung dafür ist natürlich, dass sich alle variablen Eigenschaften der Objekte in der Datenbank befinden. Bevor wir uns aber der Datenstruktur widmen, folgt zunächst eine kurze Vorstellung der 27 Programmklassen und ihrer Funktionen.

4.1.3.3 Siebenundzwanzig Klassen

Am Anfang des Kapitels wurden bereits die Klassen vorgestellt, die von GTGE benötigt werden, um überhaupt erst einmal ein Spiel zum Laufen zu bringen.

Eba *Hauptprogrammklasse mit Mainmethode*
EbaGameEngine
EbaGameObject

Ausserdem gibt es sieben GUI-Klassen, die als Kindklassen des EbaGameObject an seiner Stelle von der GameEngine instantiiert werden, um den jeweiligen Raum mit seinen Interface-Funktionen abzubilden.

EbaGuiStart
EbaGuiEdna
EbaGuiEdnaJung
EbaGuiHarvey
EbaGuiDragSkript
EbaGuiSkriptOnKlick
EbaGuiZen

Die übrigen 17 Klassen und ihre Aufgaben werde ich an dieser Stelle in aller Kürze vorstellen.

EbaSpriteGroup

EbaSpriteGroup erweitert die von GTGE angebotene Klasse SpriteGroup um die Methoden „getSprite(int id)“ und „getSprite(String dataID)“, die es erlauben, die entsprechende Spritegroup nach einem bestimmten Objekt zu durchforsten.

EbaTextSprite

EbaTextSprite erweitert den VolatileSprite, so dass man eine beliebige Zeichenkette als Sprite anzeigen kann. Schriftart, Dauer und Koordinaten werden zusammen mit der Zeichenkette übergeben. Der TextSprite bekommt grundsätzlich eine 3 Pixel breite Umrandung in einer zweiten Farbe (bei Eba ist es in jedem Fall schwarz), damit man die Buchstaben unabhängig von der Hintergrundfarbe lesen kann.

EbaCommandPromptSprite

Sorgt dafür, dass Textsprites in der Kommandozeile angezeigt werden. Der CommandPromptSprite funktioniert ähnlich wie der Text Sprite. Da die Kommandozeile aber nicht nach einer Weile verschwinden darf, erbt diese Klasse von Sprite anstatt von VolatileSprite. Über die Methode „change“ lässt sich der EbaCommandPromptSprite aktualisieren.

EbaChoiceList

Erbt von EbaSpriteGroup. Mit dieser sehr speziellen Spritegroup werden die interaktiven Auswahloptionen eines Dialoges dargestellt. Die Klasse sorgt aber auch für die Ausführung des hinterlegten Skriptes, wenn eine Option ausgewählt wurde, bringt also seine eigene Ereignisbehandlung mit.

EbaInteractionSprite

EbaInteractionSprites bilden jene Objekte auf dem Spielfeld ab, die in der Datenbank als „Raumobjektinteraktion“ zu finden sind. Alle Eigenschaften dieser Objekte haben hier ein entsprechendes Datenfeld und können über Getter- und Setter-Methoden gesetzt und gelesen werden.

EbaObjectSprite

EbaObjectSprites bilden jene Objekte auf dem Spielfeld ab, die in der Datenbank als „Raumobjektdarstellung“ zu finden sind. Da ein Raumobjekt entweder eine Raumobjektinteraktion oder eine Raumobjektdarstellung, aber genauso gut auch beides sein kann, mussten wir uns, um das Problem der Mehrfachvererbung zu vermeiden, auf eine Vererbungshierarchie einigen. Darum erbt der EbaObjectSprite vom EbaInteractionSprite.

EbaExitSprite

Auch der EbaExitSprite erbt vom EbaInteractionSprite. Bei Ausgängen, die über dieses Objekt abgebildet werden, handelt es sich um spezielle Raumobjektinteraktionen, die über einige zusätzliche Parameter wie zum Beispiel den Zielraum und die Walk-In-Koordinaten für den Raumwechsel verfügen.

EbaPlayerSprite

Der EbaPlayerSprite bildet den Spielercharakter mit all seinen Funktionen und Animationen ab. Hier wird entschieden, bei welcher Aktion welche Animation abzuspielen ist und wann der Aktionmodus wieder gewechselt werden kann. Es wird sowohl die Länge seiner zugehörigen Bildschirmtextausgabe als auch die Skalierungsgröße berechnet. Wird ein Player Sprite initialisiert, so baut er sich selbstständig und dynamisch anhand seines in der Datenbank festgehaltenen Characteranimationset einen Array aus BufferedImages zusammen, in dem alle Einzelbilder jeder Animation hintereinander weg gespeichert werden. Gleichzeitig merkt er sich über einen Array aus Referenzzahlen, die Anfangs- und Endframes dieser Bildfolgen. So ist es möglich, verschieden lange Animationen für analoge Aktionsmodi bei unterschiedlichen SC abzuspeichern.
Unser Standard-SC Edna besteht in den meisten Räumen aus 82 Einzelbildern, die sich in 26 Bildfolgen unterteilen.

Der Playersprite ist auch für die Kommunikation mit der WalkableAreaMap verantwortlich.

EbsNscSprite

Der NscSprite verhält sich analog zum EbaPlayerSprite und ist nur deshalb keine Kindklasse von ihm, weil die Grafiken, aus denen er sich seine Animationen zusammenlädt, an einer ganz anderen Stelle in der Datenbank zu finden sind.

EbaInterfaceHasBase

Ein Interface (im programmiertechnischen Sinne), das den Klassen EbaObjectSprite, EbaPlayerSprite und EbaNscSprite Datenfelder für einige Eigenschaften betreffend ihrer Grundlinie reserviert.

EbaItemSprite

EbaItemSprites bilden jene Objekte auf dem Spielfeld ab, die in der Datenbank als „Inventarobjekt“ zu finden sind. Alle Eigenschaften dieser Objekte haben hier ein entsprechendes Datenfeld und können über Getter- und Setter-Methoden gesetzt und gelesen werden.

EbaTopicSprite

EbaTopicSprites bilden jene Objekte auf dem Spielfeld ab, die in der Datenbank als „Topic“ zu finden sind. Alle Eigenschaften dieser Objekte haben hier ein entsprechendes Datenfeld und können über Getter- und Setter-Methoden gesetzt und gelesen werden.

EbaHauptmenue

Das EbaHauptmenue erbt ähnlich wie die Choicelist von SpriteGroup. Ihre Einzelkomponenten werden bereits beim Raumstart geladen und warten nur darauf, sichtbar geschaltet zu werden. Im Hauptmenü finden die Laden- und Speichern-Funktionen aus der Klasse EbaFileIO ihre grafische Darstellung.

EbaFileIO

EbaFileIO bietet Methoden an, um die Datenbanktabellen zur Spielsicherung und Spielwiederherstellung zu kopieren.

EbaScriptInterpreter

Der EbaSkriptInterpreter liest, sobald über das Interface ein vollständiger Befehl ausgelöst wurde, die Datenbanktabelle „Skript“ aus und führt Zeile für Zeile die unter der übrgebenen SkriptID hinterlegte Anweisungsliste aus. Das Ausfüllen dieser Tabelle ist Aufgabe des Skripters, die 39 Befehle der in dieser Klasse programmierten, primitiven Skriptsprache sind sein Werkzeug. Eine vollständige Liste befindet sich im Anghang auf der CD.

EbaWalkableAreaMap

In dieser Klasse findet sich der komplxe Wegfindungsalgorithmus, der es dem SC erlaubt, anhand eines booleschen Arrays den bestmöglichen Weg zwischen zwei Punkten zu finden. Im Weg stehende Hindernisse werden umrundet.

EbaWalkableAreaMapMaker

Der EbaWalkableAreaMapMaker ist ein eigenständiges Programm, das zur Laufzeit des Spiels nicht benötigt wird. Mit seiner Hilfe lassen sich zweifarbige Bilddateien im Gif-Format in WAM-Files umrechnen, die den booleschen Array enthalten, der zur Laufzeit des Spiels von der Klasse EbaWalkableAreaMap benötigt wird.

4.1.4 Die Datenstruktur

„Edna bricht aus“ ist so konzipiert, dass Programm und Content voneinander getrennt sind. Während sich die Programmierung in den 27 Javaklassen der Engine befindet, verteilt sich der Content auf 20 Tabellen einer einfachen HSQL-Datenbank. Bei diesen Tabellen handelt es sich um CSV-Dateien, wie sie auch von Microsoft Excel zur Tabellenkalkulation verwendet werden. Mit einem einfachen Texteditor geöffnet, erkennt man, dass hier die einzelnen Felder der Tabelle mit Semikoli abgetrennt in einer Zeile stehen. Ein Zeilenumbruch bedeutet auch eine neue Zeile in der Tabelle. In diese Tabellenfelder einfachsten Formats werden die spielbestimmenden Parameter eingetragen.

Im laufenden Spiel wird sich der Inhalt dieser Tabellenfelder natürlich ändern. Deshalb ist es nötig, das laufende Spiel vom ursprünglichen Auslieferungszustand zu trennen: Zu Beginnn des Spiels werden alle 20 Tabellen aus ihrem Stammverzeichnis „EbaStart/“ in das temporäre Verzeichnis „EbaData/“ kopiert. Das laufende Spiel greift nun nur noch auf die Tabellen in „EbaData/“ zu. Nur hier werden Veränderungen durch das Spiel vorgenommen. Die Tabellen in „EbaStart/“ bleiben stets unverändert.

So gibt es zum Beispiel ein Feld „Aktiv“ in der Tabelle „Raumobjekt“, das anzeigt, ob sich ein Raumobjekt im Raum befindet oder nicht. Ist der Wert in diesem Feld auf „true“ gesetzt, so bedeutet es, dass dieses Objekt im Raum aktiv ist, „false“ bedeutet das Gegenteil. Ist dem Raumobjekt eine „Raumobjektdarstellung“ zugeordnet, wird das Objekt angezeigt, sobald das Feld auf „true“ steht. Ist ihm eine „RaumobjektInteraktion“ zugeordnet, kann das Objekt nun von Edna benutzt werden. Das Feld „Raumobjekt.Aktiv“ ist ein gutes Beipiel für einen Parameter, der sich im Laufe des Spieles verändert. Nimmt Edna einen Gegenstand, so verschwindet seine

Darstellung und seine Interaktivität im Raum. Der Gegenstand taucht stattdessen im Inventar auf. Auch wenn Edna den Raum verlässt und wieder betritt, bleibt der Gegenstand verschwunden. Dies ist nur möglich, weil der Parameter „Aktiv“ in der Datenbank für das laufende Spiel verändert wurde. Beginnt man das Spiel hingegen von neuem, muss der Gegenstand wieder an Ort und Stelle sein.

Da es ziemlich unwahrscheinlich ist, dass man „Edna bricht aus“ in nur einer einzigen Sitzung durchspielt, ist es nötig, den veränderten Zustand bei Bedarf abspeichern und zu einem späteren Zeitpunkt wieder laden zu können. Die Funktion „Speichern“ des Hauptmenüs tut wenig Anderes, als die Tabellen des laufenden Spiels aus „EbaData/“ in einen von neun Savegame-Ordnern mit den Namen „EbaSaveGame1/“ bis „EbaSaveGame9/“ zu kopieren. Zusätzlich wird ein Screenshot der momentanen Spielsituation abgespeichert und eine Handvoll Parameter, die nicht im übrigen Datenbankmodell erfasst werden (Der aktuelle Raum, Ednas Koordinaten sowie ihre Blickrichtung). Die SaveGame-Parameter stehen in einer Tabelle „SaveGame“. Diese hat eine Sonderstellung im Datenbank Modell, weil ihr Inhalt als einziger nicht durch die Lade- und Speichervorgänge mitkopiert wird. Lädt man einen Raum über das Hauptmenü, so wird das laufende Spiel mit den Tabellen des gewünschten Spielstands überschrieben. Edna wird auf die im Savegame hinterlegten Koordinaten positioniert und anhand der abgespeicherten Blickrichtung ausgerichtet. Damit ist die Spielsituation vom Zeitpunkt des Abspeicherns vollständig rekonstruiert worden. Das Spiel ist geladen.

Mit der EBA-GameEngine lassen sich also beliebig viele Adventure-Spiele entwickeln, ohne auch nur eine einzige Codezeile neu zu programmieren. Lediglich dann, wenn man ein User-Interface wünscht, das sich in der Funktion von den sieben angebotenen Interfaces unterscheidet, ist es von Nöten, eine neue GUI-Klasse zu entwickeln, die in ihrer Funktionsweise analog zu den bisherigen GUIs funktioniert. Grafische Änderungen des Interfaces können es nötig machen, dass die Position von Schaltflächen neu bestimmt werden muss. Auch hierfür gibt es bislang kein Feld in der Datenbank. Die entsprechenden Änderungen müssen direkt in der GUI-Klasse vorgenommen werden. Weitere Spieleparameter, die bislang „hart“ in den Spielcode integriert sind, obwohl sie sich als gestalterisches Element von Spiel zu Spiel unterscheiden dürften, sind die Liste möglicher Schriftarten und die Anzeigedauer des Bildschirmtextes.

Alle weiteren Spieleparameter, die abhängig vom Inhalt des Spieles sind, nicht von seiner Funktionsweise, befinden sich in der Datenbank. In diesem Kapitel werde ich die Tabellen der Datenbank im Einzelnen beschreiben. Im Anhang findet sich zusätzlich eine tabellarische Erläuterung der einzelnen Felder. Eine genaue Aufstellung der möglichen Skriptbefehle, die in der wichtigsten Datenbank-Tabelle, der „Skript"-Tabelle eingetragen werden dürfen, findet sich ebnefalls im Anhang. In Abschnitt **4.2** folgt später ein Beispiel, an dem das Ausfüllen der Datenbank veranschaulicht wird.

Nach der Lektüre nur dieser beiden Kapitel und des Anhangs sollte es also möglich sein, eigene Adventure-Spiele mit der EBA-GameEngine zu entwickeln.

RAUM

Wie in Abschnitt 3 gezeigt wurde, ist der Raum als kleinste wiederholbare Einheit die Grundlage des Datenmodells. Beim Start des Spieles werden nur jene Objekte geladen, die sich im aktuellen Raum befinden. Bei einem Raumwechsel werden die Objekte des aktuellen Raums verworfen. Die Objekte des Zielraums treten an ihre Stelle.

WALKABLEAREAMAP

Jeder Raum verfügt über genau eine Walkable-Area-Map. Es handelt sich hierbei um einen booleschen Array von 800x600 Feldern, entsprechend der Bildschirmauflösung des Spiels. Für jeden Bildpunkt ist hier durch eine boolesche Variable verzeichnet, ob der PlayerCharacter auf diesem Pixel stehen darf oder nicht. Während der Entstehung des Spiels schrieb Stefan Hütter ein kleines eigenständiges Programm, das in der Lage ist, zweifarbige Gif-Dateien in WAM-Files umzurechnen, die den fertigen booleschen Array enthalten. So lassen sich schnell und unkompliziert die benötigten WAMs aus schwarz/weiss maskierten Hintergrundbildern erstellen.

GUI

In der GUI-Tabelle sind die variablen Parameter eingetragen, die nur für ein spezielles GUI gültig sind. Da im Spielfluss ein GUI stellvertretend für einen Spielercharakter steht (Es gibt das EdnaGUI, das HarveyGUI und das Edna_JungGUI), werden an dieser Stelle auch die charakterspezifischen Schriftarten für die Textausgabe angelegt.

TIMER

Jedem Raum kann ein Timer zugeordnet sein. Ist er aktiv, so wird in dem enstsprechenden Raum nach Ablauf der hier eingetragenen Dauer (in Milisekunden) das angegebene Skript ausgeführt.

SKRIPT

Die wichtigste Datei des Spiels ist die Tabelle „Skript“. Hier finden sich die abgefangenen Reaktionen auf alle denkbaren Ereignisse, die durch Kombinationen aus Befehl und Raumobjekt, Item und Raumobjekt, Befehl und Item oder Item und Item, aber auch durch Topics (nur Harvey-GUI), Timer oder Choicelist-Auswahl ausgelöst wurden. Ist für ein Ereignis kein Skript hinterlegt, so reagiert Edna mit Skript null: „say(Das klappt so nicht.)“. Ist für ein Ereignis zwar eine SkriptID angelegt worden, die entsprechende ID lässt sich in der Skript-Tabelle aber nicht finden, so reagiert Edna mit „say(Da weiss ich einfach nicht, was ich machen soll.“). Läuft im aktiven Raum ein Timer ab, ohne dass ein Skript hinterlegt ist, so reagiert Edna mit „say(Irgendetwas ist gerade passiert. Ich weiss aber nicht was.)“. Beim Skripten haben wir uns alle Mühe gegeben, das kein einziges dieser Ereignisse jemals eintritt. Im Gegenteil haben wir versucht, so viele Ereignisse wie möglich „persönlich“ zu behandeln. Auf jede Kombination aus Befehl und Raumobjekt sollte Edna eine Reaktion parat haben, die genau auf diese Situation passt, auch wenn nur ein Bruchteil der möglichen Befehle sinnvoll sind. Das selbe gilt im eingeschränkten Maße für Kombinationen aus Item und Raumobjekt. Wenn hier mal kein maßgeschneidertes Skript hinterlegt ist, dann wird zumindest ein Standardskript ausgeführt, dass universell für das benutzte Item gültig ist.

Skripte bestehen nicht selten auch aus mehreren Anweisungen, die hintereinander ausgeführt werden. Wenn im Spiel eine selbstablaufende Ereignisfolge stattfindet, in der die Charaktere eine kurze Handlung ausführen oder einen Dialog sprechen, dann nennt man das „Cutszene“. Auch hierfür ist die Skript-Engine verantwortlich. Ein Skript, das eine Cutszene beinhaltet, kann mitunter über hundert Zeilen lang werden.

Ein Skript hat neben seiner ID also auch eine Zeilennummer. Ein Skript wird immer mit Zeile eins gestartet. Ist eine Zeile abgearbeitet, so überprüft der Skriptinterpreter, ob die Skriptzeile mit der nächsthöhreren Zeilennummer existiert. Ist dies nicht der Fall, so gilt

das Skript als beendet. Sind Skriptzeilen also nicht konsequent aufsteigend durchnumeriert, so werden die abgesetzten Teile des Skriptes einfach ignoriert.

Insgesamt erstreckt sich die Skript-Tabelle von „Edna bricht aus“ über etwa 50.000 Zeilen. Eine vollständige Auflistung der einzelnen Skriptbefehle ist im Anhang auf der CD zu finden.

CHOICELIST

Befindet sich Edna in einem Gespräch, so darf der Spieler mitunter eine passende Antwort aus einer Auswahl vorgefertigter Gesprächsthemen aussuchen. Diesen Auswahlbereich habe ich „Choicelist“ getauft. Jeder Auswahlmöglichkeit ist neben einer Auswahlnummer und dem Auswahltext auch ein boolesches Aktiv-Flag und eine SkriptID zugeodnet. Innerhalb einer Choicelist dürfen maximal 10 Auswahlzeilen gleichzeitig aktiv sein, mehr können nicht angezeigt werden. Anders als beim Skript müssen die Auswahlnummern nicht unbedingt aufeinander folgen. Die einzelnen Zeilen werden lediglich in der Reihenfolge dieser Nummern sortiert. Auch dient die Auswahlnummer zur eindeutigen Identifizierung der Gesprächsoption, wenn man per Skript eine Choicelist-Auswahlmöglichkeit aktivieen oder deaktivieren möchte. Dadurch ist es möglich, dasselbe Gespräch durch Ereignisse von ausserhalb mit neuen Gesprächsthemen zu erweitern. So trifft Edna im Aufenthaltsraum der Irrenanstalt sowohl auf den „Alumann“ als auch auf „Droggelbecher“. Droggelbecher versperrt ihr den Zutritt zur Kissenburg. Edna kann sich beim Alumann über dieses empörende Verhalten auslassen. Es würde aber wenig Sinn ergeben, diese Gesprächsoption schon von Anfang an zu aktivieren. Bevor Edna nicht probiert hat, die Kissenburg zu betreten, kann sie auch nicht darüber reflektieren, wie unfair es ist, dass sie nicht hinein darf.

Man kann sich leicht vorstellen, dass es recht aufwendig ist, die Dialoge so zu gestalten, dass sie über einen langen Zeitraum des Spieles Gültigkeit behalten, und noch schwerer, sie zu pflegen, dass bei entscheidenden Veränderungen in der Spieleumgebung, alle Dialogsoptionen und Reaktionen nach wie vor Sinn ergeben (Wenn Edna es einmal geschafft hat, die Kissenburg zu betreten, dann muss sich die Gesprächsführung mit dem Alumann wieder ändern. Wir wissen aber nicht mit Sicherheit, ob sich der Spieler überhaupt jemals die Mühe gemacht hat, darüber zu reden).

RAUMOBJEKT

Alle Objekte, mit denen der Player Character im Laufe des Spieles interagieren kann, oder die dargestellt werden müssen und die nicht dem GUI, dem Hauptmenü, dem Playersprite oder einer Choicelist zuzuordnen sind, sind Raumobjekte. Dazu zählen Raumobjekte mit RaumobjektInteraktion (ROI), Raumobjekte mit RaumobjektDarstellung (ROD), Ausgänge und NSCs. Ein einzelnes Raumobjekt kann dabei sowohl eine ROI als auch eine ROD haben und gleichzeitig noch ein NSC sein. Beliebige Kombinationen sind möglich. Alle Unterkathegorien müssen aber einem Raumobjekt zugeordnet sein. Hier sind nämlich wichtige Parameter, wie die Positionierung im Bild, der Pfad zur Bilddatei und das Aktiv-Flag eingetragen.

RAUMOBJEKTINTERAKTION

Jedes Raumobjekt, mit dem Edna interagieren können soll, ist ein RauobjektInteraktion. Das gilt auch für Ausgänge und NSC. Unter Anderm sind hier der Walk-To-Point und die vier Interaktionsskripte für „Ansehen“, „Benutzen“, „Nehmen“ und „Reden“ hinterlegt.

AUSGANG

Ein RaumobjektInteraktion kann ein Ausgang sein. In dem Fall verwandelt sich der Cursor beim Mouseover in den in der GUI-Tabelle referenzierten Ausgangs-Cursor. Bei Klick auf den Ausgang geht Edna zum WalkToPoint der zugehörigen Raumobjektinteraktion und führt dort für gewöhnlich einen Raumwechsel durch. Ist allerdings beim ROI ein Benutzen-Skript hinterlegt, so wechselt sie den Raum nicht. Stattdessen wird das Skript abgearbeitet (das seinerseits natürlich trotzdem einen Raumwechsel beinhalten darf). Dieses Skript nennt sich dann „Raumwechselskript“.

Da, wie oben beschrieben, bei einem Raumwechsel alle Objekte bis auf die GameEngine vom Programm verworfen und (wenn sie im neuen Raum benötigt werden) neu geladen werden müssen, schreiben sich die Ausgangsparameter vor dem Raumwechsel in Variablen der EbaGameEngine. Auch die Information, ob gerade ein Skript ausgeführt wird, und wenn ja, in welcher Zeile es sich gerade befindet, werden nun in der GameEngine vermerkt. Mit diesen Parametern wird dann der neue Raum gestartet.

RAUMOBJEKTDARSTELLUNG

Es gibt drei Gründe, aus denen ein Raumobjekt eine Raumobjektdarstellung benötigt:

Das Raumobjekt ist nicht Teil der Hintergrundgrafik, soll aber im Bild auftauchen.
Das Raumobjekt ist zwar Teil der Hintergrundgrafik, es soll aber möglich sein, dass der PlayerCharacter hinter dem Objekt verschwindet.
Dem Raumobjekt ist über die Tabelle „Bildfolge“ eine Animation zugeordnet.

Für den zweiten Fall müssen Kriterien gefunden werden, nach denen entschieden wird, ob der Spielercharakter vor oder hinter dem Objekt steht. Für die meisten Objekte könnte man sagen, dass es ausreicht, die Y-Koordinate des Fußpunktes zu kennen. Edna befindet sich nur dann vor dem Tisch in ihrer Zelle, wenn ihre Y-Koordinate sich oberhalb der Linie befindet, die zwischen den Punkten gespannt ist, an denen die vorderen Tischbeine den Boden berühren. So eine gedachte Linie heißt „Baseline“. Die Baseline des Tischs verläuft Horizontal auf der Y-Koordinate des Fusspunktes, des unteren mitleren Punktes der Bilddatei, die den freigestellten Tisch zeigt. Doch es gibt noch ganz andere Fälle.
Bei einer von der Zimmerdecke herabhängenden Lampe zum Beispiel befindet sich der Fußpunkt keinesfalls am unteren mittleren Ende des Bildbereichs. Fällt man das Lot nach den Gesetzen des perspektivischen Zeichnens auf die untere Ebene, findet man den Fusspunkt dort, wo das Lot die Ebene schneidet. Auch hier reicht eine Waagerechte durch den so ermittelten Punkt, um zu entscheiden, ob der PlayerCharacter vor oder hinter dem Objekt steht.
Ganz anders verhält es sich allerdings, wenn das Objekt selbst in der perspektivischen Flucht liegt, den PlayerCharacter aber nicht verdeckt. Ein gutes Beispiel hierfür ist die Gittertür im Flur vor dem Aufenthaltsraum in „Edna bricht aus“. Wenn Edna das erste Mal diesen Ort passiert befindet sie sich hinter dem Gitter, wird also in jedem Fall von dem Objekt verdeckt, unabhängig von ihrer Y-Koordinate. Alsbald wird sie jedoch von einem Wächter aufgegriffen und in den geschlossen Bereich der Anstalt gesperrt. Nun befindet sie sich in jedem Fall vor dem Gitter, obwohl sie sich auf der selben Tiefe im Bild, auf den selben Y-Koordinaten bewegen kann.
Nach dem Lösen einer Großzahl von Rätseln kann Edna sich schließlich eine Kopie des Generalschlüssels anfertigen. Mit diesem Objekt ist es ihr möglich, die Gittertür zu öffnen. Wie wird nun entschieden, auf welcher Seite des Gitters sich Edna befinden

muss? Es lässt sich keine Y-Koordinate finden, von der sich pauschal sagen liesse, dass Edna ab diesem Punkt vor oder hinter dem Gitter ist.

Darum ist es in „Edna bricht aus“ möglich, einer RaumobjektDarstellung vier Koordinaten mitzugeben, die eine beliebige Grundlinie im Raum aufspannen. Die von Edna als Referenz benutzt wird, ob sie sich vor oder hinter dem Objekt befindet.

Abb. 4.7.: Die „Baseline“ entscheidet, auf welcher Seite eines Objektes sich der SC befindet.

BILDFOLGE

Bei einer RaumobjektDarstellung kann es sich auch um ein animiertes Objekt handeln. In diesem Fall ist ihnen im Feld „BildfolgeID“ eine Bildfolge zugeordnet. Auch NSCs sind animierte Objekte. Sie beziehen ihre Bildfolgen allerdings aus ihrem CharacterAnimationSet, nicht aus ihrer Raumobjektdarstellung. In beiden Fällen kann eine solche Bildfolge entweder in einer Schleife ablaufen, oder darauf warten, dass sie per Skriptbefehl einmalig abgespielt wird.

Einer Bildfolge können beliebig viele Animationsbilder zugeordnet sein. Welche Bilder das sind, und wie oft und in welcher Reihenfolge sie abgespielt werden, ist in der Tabelle „Animationsbild“ festgehalten. In der Tabelle Bildfolgen befinden sich lediglich die übergeordneten Einstellungen: Die Anzeigedauer eines Animationsframes und ob es sich um eine Schleife handelt.

ANIMATIONSBILD

Eine Bildfolge besteht aus einer Serie von Animationsbildern. Diese sind in der Tabelle „Animationsbild“ aufgelistet. Sie werden in der Reihenfolge ihrer ID abgespielt.

Verfügen sie über eine „AbweichendeAnzeigedauer“ die größer ist als eins, so wird das entsprechende Bild um den hier eingetragenen ganzzahligen Faktor länger angezeigt.

Die Breite und Höhe einer Animation richtet sich nach den Dimensionen ihres ersten Bildes. Darum sollte die Animation auch möglichst aus gleichgroßen Einzelbildern bestehen, zumindest sollte das erste Bild der Animation die maximale Breite und Höhe aufweisen.

CHARACTERANIMATIONSET

Ein Characteranimationset (CAS) ist eine Sammlung von Bildfolgen, die zu einem bestimmten Charakter gehören. Für jeden Charakter, egal ob SC oder NSC, gibt es mindestens einen Eintrag in der Tabelle „Characteranimationset“. Ein vollständiges CAS besteht allerdings aus mehreren Einträgen, einem pro möglichen Aktionsmodus. Für NSCs gibt es die drei Standard-Aktionsmodi „Warten“, „Reden“, „Gehen“. Für SCs gibt es noch einen weiteren Aktionsmodus: „Denken“. Nicht für alle Aktionsmodi muss tatsächlich auch ein Eintrag hinterlegt sein. Es sind aber auch uneingeschränkt viele weitere Aktionsmodi denkbar, sogenannte „Zusatzaktionsmodi“.

Für jeden Aktionsmodus können vier Bildfolgen hinterlegt werden – eine für jede mögliche Blickrichtung.

AKTIONSMODUS

In dieser Tabelle sind die möglichen Aktionsmodi aufgelistet. Die ersten vier Aktionsmodi sind hierbei für die Standardaktionen reserviert. 0 = „Gehen“, 1 = „Reden“, 2 = „Denken“, 3 = „Warten“. Es sind beliebig viele weitere Aktionsmodi denkbar.

Da diese Tabelle ausser einer Liste der IDs nur eine Bezeichnung enthält, die nur der Orientierung des Skripters dient, könnte man auf die Idee kommen, dass die Tabelle redundant ist. Sie wird allerdings vom Programm benötigt, um Sprites für SCs und NSCs dynamisch aus der Datenbank zu laden. Alle Animationsbilder eines Charakters werden pro Raum in einem einzigen Array Of Buffered Images, einem dynamisch erzeugten Feld aus Bildern, hinterlegt.

NSC

Nichtspielercharaktere sind im EbaDatenmodell spezielle Raumobjekte, die über ein Charakterenimationsset in beliebig vielen Aktionsmodi verfügen. Mindestens sollten sie aber „warten“ können, da dies ihr Grundzustand ist. Die meisten NSCs können allerdings auch „reden“. Darum ist ihnen ein Font zugeordnet.

Da nicht gewährleistet ist, dass der NSC im selben Größenverhältnis gezeichnet ist wie der Spielercharakter, muss es die Möglichkeit geben, ihm abweichende Parameter für die perspektivische Darstellung mitzugeben. Ist ein NSC unbeweglich an einem Ort, so ist er häufig genau passend für den Hintergrund gezeichnet. In diesem Fall soll er die Raumperspektive überhaupt nicht beachten. Dies wird erreicht, indem die Werte YatZeroScale und YatFullScale auf den selben Wert gesetzt werden. Nun ist der NSC-Sprite von der Skalierung ausgenommen und wird immer in der Originalgröße seiner Bilddatei angezeigt.

NSCs können über Skriptbefehle analog zum PlayerSprite gesteuert werden. Sie bekommen sogar ihre eigene Instanz der WalkableAreaMap. So können sie sich genau wie der PlayerCharacter durch den Raum bewegen. Dadurch kann es vorkommen, dass sie genau wie der PlayerCharacter die Baseline einer RaumobjektDarstellung passieren. In diesem Fall ist es sehr hinderlich, dass sie selbst ebenfals über eine RaumobjektDarstellung mit einer fest eingeskripteten Baseline verfügen. Werden die vier BaselineParameter der RaumobjektDarstellung auf –1 gesetzt, so ignoriert der NSC, dass er über eine Baseline verfügt und benuzt seinen Fußpunkt, um zu entscheiden, ob er sich vor oder hinter einem Gegenstand befindet. Nur so ist es möglich, dass Hulgor in Doktor Marcels Büro Edna um den Tisch jagt, wenn sie ihm zu nahe kommt.

INVENTAROBJEKT

Das EdnaGUI verfügt über ein Inventar mit fünf mal sieben Feldern, das von der unteren linken Ecke zeilenweise nach oben aufsteigend befüllt wird, wenn Edna im Laufe des Spieles Gegenstände aufsammelt. Hat sie mehr als 35 Gegenstände eingesammelt, so erweitert sich das Inventar automatisch. Zwei Pfeile erscheinen, um das Inventar eine Zeile hoch oder runter zu verschieben. Bei EBA gibt es alleine 79 Inventarobjekte (INVO) für den Edna-Handlungsstrang, die auch „Items“ genannt werden. 68 davon findet sie im ersten Akt. Auch das Edna_jung-GUI kennt

Inventarobjekte. Darum wird in dieser Tabelle auch vermerkt, zu welchem GUI das Item gehört. Das Inventarobjekt verfügt über kein Aktiv-Flag. Es befindet sich im Inventar, sobald der Parameter „Inventarposition“ größer null ist. Im Übrigen funktioniert ein Inventarobjekt analog zur Raumobjektinteraktion.

BENUTZEMIT

In der Tabelle BenztzeMit ist hinterlegt, welches Skript ausgeführt werden soll, wenn ein Inventarobjekt mit einer RaumobjektInteraktion kombiniert wird. Im Initialzustand des Spieles folgen die SkriptIDs zur besseren übersicht strengen Konventionen, so dass der hier eingetragene Zusammenhang auch maschinell hergestellt werden könnte. In Einzelfällen ist jedoch bei zwei Kombinationen von vorne herein das selbe Skript hinterlegt.
Ausserdem muss es in die Tabelle alleine deswegen schon geben, weil sich die hier getroffene Zuordnung im Verlauf des Spieles durch Skripteinwirkung ändern kann.

INVENTARBENUTZEMIT

In der Tabelle InventarBenutzeMit ist hinterlegt, welches Skript ausgeführt werden soll, wenn ein Inventarobjekt mit einem anderen Inventarobjekt kombiniert wird.

TOPIC

Die „Topic“-Tabelle ist die einzige Tabelle, die im EdnaGUI keine Verwendung findet. Sie ist ausschliesslich für das HarveyGUI bestimmt. Harvey kann sich per Drag and Drop diese grafischen Stellvertreter von RaumobjektInteraktionen in sein Interface ziehen. Von dort aus nutzt er sie, um mit Edna über das zugehörige Raumobjekt zu reden. Es handelt sich also um ein als Objekt dargestelltes Gesprächsthema.

Topics können anders als Inventarobjekte beliebig auf Harveys Topicleiste plaziert werden.

SAVEGAME

Der Tabelle Savegame ist eine Sonderstellung im Datenbankmodell inne. Sie wird von den Methoden der Lade/ und Speichervorgänge oder beim Spielstart nicht mitkopiert, denn hier stehen Informationen zu den abgespeicherten SaveGameTabellen, die im Datenbankmodell selbst nicht auftauchen, nämlich in welchem Raum sich der SC

befunden hat, wo er dort gestanden und wohin er geguckt hat. Die Savgame-Tabelle hat immer genau 9 Einträge.

Die Tabelle wird bei einem Speichervorgang automatisch von der Engine ausgefüllt und muss vom Skripter nicht angefasst werden.

4.2 Die Skript-Engine

Dem Skripter stehen insgesamt 39 einfache Befehle zur Verfügnung, um beliebige Handlungsabläufe und die damit einhergehenden Änderungen in der Spieleumgebung auszulösen. Diese Anweisungen werden vom Programm Zeile für Zeile umgesetzt.

Die Anzahl der möglichen Skriptbefehle begrenzt nicht nur die Eingriffsmöglichkeiten des Spielers in die Handlung, sondern auch die Anzahl der Ausdrucksmöglichkeiten der agierenden SC und NSC. Sie sind also sowohl für die Spiellogik als auch für die Dramaturgie von eminenter Bedeutung.

Ein besonders einfaches Beispiel für ein Skript ist das Skript mit der ID „0“. Es wird standardmäßig benutzt, wenn kein Skript hinterlegt ist. Der Eintrag in der Skript-Tabelle lautet:

```
0;1;say(Das klappt so nicht);
```

Was wir hier sehen, sind vier Felder einer HSQL-Datenbanktabelle, die mit Semikoli voneinander abgetrennt sind. Bei dem ersten Parameter handelt es sich um die ID des Skripts. Es ist Skript „0“. Der zweite Parameter zeigt uns die Skriptzeile an. Skript „0“ hat nur eine Skriptzeile, dessen Zeilennummer „1“ ist. Skripte müssen immer mit Zeile 1 beginnen und mit aufsteigender Zeilenzahl lückenfrei durchnumeriert werden.

Im dritten Feld steht die Anweisung. Es handelt sich um den Skriptbefehl „say()“. In der Klammer können dem Skriptbefehl Parameter übergeben werden. Dem Befehl „say()“ wird zum Beispiel eine Zeichenkette übergeben. Diese Zeichenkette wird als Bildschirmtext in der Farbe des im GUI hinterlegten TalkFont ausgegeben., wärend die

Animation abgespielt wird, die im CharacterAnimationSet unter dem Aktionsmodus 1 („Talking“) für die Blickrichtung, in die der SC gerade blickt, hinterlegt ist.

Das vierte Feld ist leer. Hier darf der Skripter einen beliebigen Kommentar hinterlassen, um in der sehr umfangreichen Skript-Tabelle nicht die Übesicht zu verlieren.

Verkürzt lässt sich die Bedeutung der vollständigen Skriptzeile als: Edna sagt: „Das klappt so nicht“ zusammenfassen.

Wie bereits mehrfach erwähnt, findet sich eine genaue Auflistung der Skriptbefehle und Ihrer Bedeutungen im Anhang auf der CD. Mit Ihrer Hilfe und der ebenfalls im Anhang befindlichen Beschreibung der Datenbank, möchte ich als Abschluss dieses Kapitels in einem weiteren Beispiel die Arbeit des Skripters verdeutlichen. Es handelt sich um eine Anleitung zur Erstellung eines eigenen Raums.

4.2.1 Ein eigener Raum

Nach ihrer Flucht aus der Irrenanstalt erleidet Edna mit ihren Helfern einen Autounfall. Ihre Limousine ist halb in einem Tümpel versunken und die Strasse ist durch die Wagen der Verfolger blockiert. Hoti und Moti haben es sich trotzdem in dem Autowrack bequem gemacht. Von dem Schlüsselmeister fehlt jede Spur. Den Alumann hingegen finden wir schließlich auf einer Aussichtsplattform. Der schrullige Anti-Mentor hofft dort, von einem Blitz getroffen zu werden. Dies soll die Szene sein, die ich hier in vereinfachter Form beispielhaft „vorskripten“ werde.

Abb. 4.8.: So soll der eigene Raum später aussehen.

RAUM

Als erstes müssen wir den Raum in der Tabelle „Raum“ anlegen.

Die entsprechende Zeile muss lauten:

```
200401;Plattform;bg/plattform.png;;2004;0.02;0.2;-5;520;1;1;0
```

Wir sehen hier zwölf bereits ausgefüllte Felder, die mit semikoli voneinander getrennt sind.

Das erste Feld ist die **RaumID**. Es handelt sich um den vierten Raum im zweiten Akt, von dem es nur eine Version gibt. Also ist diese ID nach der im Anhang beschriebenen Konvention 200401.

Die **Bezeichnung** des Raumes ist „Plattform“. Sie dient uns nur zur Orientierung.

Als nächstes kommt der **Pfad zur Bilddatei** im png-Format (ich habe mir hier diesen kurzen Pfad ausgedacht, damit alles in eine Zeile passt).

Das leere Feld dahinter wird später einmal den **Pfad zur Audiodatei** enthalten. Zunächst müssen wir ohne auskommen.

„2004“ in Feld 5 ist nicht etwa eine Jahreszahl, sondern die **ID der WalkableAreaMap**. Unter dieser Nummer wird später die WalkableAreaMap in der entsprechenden Tabelle geführt.

Es folgen zwei Felder für die **vertikale** und **horizontale Geschwindigkeit**. Da sich Edna in diesem Raum eigentlich nur auf einer Linie bewegen kann, kommt der vertikalen Geschwindigkeit kaum Bedeutung zu. Die Horizontale Geschwindigkeit „0.2“ ist ein Erfahrungswert.

Bei den nächsten beiden Werten handelt es sich um die **Y-Koordinate des Fluchtpunktes** und die **Y-Koordinate**, an der Edna ihre **maximale Skalierung** erreicht hat. Auch diese Werte brauchen hier nur eine grobe Abschätzung, da Edna sich nur wenige Pixel in die Perspektive bewegen kann. Auf diesen wenigen Pixeln sollte sie nach Möglichkeit kaum ihre Skalierung ändern. Darum sind die Pixel weit auseinander gewählt.

Ednas **GuiID** ist 1. (Den Inhalt der entsprechenden Tabelle werde ich in diesem Beispiel aussen vor lassen. Man möge mir glauben, dass es sich bei dem GUI mit der ID 1 um das Edna-GUI handelt)

Ednas **CharakteranimationsetID** ist ebenfalls 1.

Einen Timer gibt es in diesem Raum nicht. Darum ist an der letzten Position als **TimerID** eine 0 eingetragen.

WALKABLE AREA MAP

Die Raum-Tabelle ist also ausgefüllt. Als nächstes sollten wir dafür sorgen, dass Edna auch irgendwo stehen kann. Dafür brauchen wir eine WalkableAreaMap. Dazu müssen wir in einem Schwarzweissbild den begehbaren Bereich weiss, den nicht begehbaren schwarz markieren. Das Produkt sollte in diesem Fall so aussehen:

Abb. 4.9.: Die WAM von Raum 200401.

Wie schon erwähnt kann sich Edna in diesem Bild nicht weit in die Tiefe bewegen. Mit dem WalkableAreaMapMaker können wir das so erstellte GIF in einen WAM-File konvertieren. Nun ist es uns möglich, die Tabelle auszufüllen.

```
2004;200401;bg/2004.wam
```

Hinter den einzelnen Feldern stehen folgende Inhalte: 1) WalkableAreaMapID; 2) RaumID; 3) Pfad zur WAM-Datei

Feld Nummer 1) dürfte uns noch aus der Raum-Tabelle bekannt sein. Es ist dieselbe WalkableAreaMapID, die dort bereits referenziert ist. Die beiden Felder müssen übereinstimmen, damit die WalkableAreaMap gefunden wird.

RAUMOBJEKT

Edna kann die Szene jetzt betreten und in ihr herumlaufen. Unser Raum ist allerdings noch vollkommen leer und es gibt keine Objekte, mit denen sie Interagieren könnte. Ausserdem ist Edna vor dem Geländer gezeichnet, welches sie eigentlich bedecken müsste. Was uns fehlt, sind Raumobjekte.

```
20040101;Treppe ;200401;694;583;2004/treppe.png;true
20040201;Bank   ;200401;620;354;2004/bank.png;true
20040301;Mond   ;200401;243;234;2004/mond.png;true
20040401;Anstalt;200401;186;584;2004/anstalt.png;true
20040501;WB     ;200401;583;600;2004/wb.png;true
20040601;Flasche;200401;400;253;2004/flasche.png;true
20040701;Alumann;200401;457;294;2004/alumann_wait.png;true
20040801;Blitz  ;200401;583;600;2004/wb.png;false
```

Mit acht Raumobjekten ist dieser Raum verhältnismässig leer. Die Treppe soll nachher unser Ausgang sein, Alumann ist ein NSC. „WB“ ist hier eine Abkürzung für „Walk Behind“. Damit ist eine reine Raumobjektdarstellung gemeint, deren einziger Zweck darin besteht, dass Edna von ihr verdeckt wird.

Abb. 4.10.: Ein typischer „Walk Behind“.

Die einzelnen Felder der Raumobjekt-Tabelle sind: 1) ID; 2) Bezeichnung; 3) RaumID; 4) X-Koordinate; 5) Y-Koordinate; 6) Pfad zur Bilddatei; 7) Aktiv-Flag

Die Konvention einer Raumobjekt ID ist: Vierstellig die Raumnummer, zweistellig die Objektnummer, zweistellig die Version. Alle Objekte dieses Raumes liegen nur in einer Version vor.

Ausserdem sind alle Objekte im Grundzustand des Raumes hier aktiv. Darum steht ihr Aktiv-Flag auf „true“. Am Schluss soll es Edna möglich sein, das Objekt „Flasche“ zu

stibizen. Anhand dieses Beispiels werde ich dann zeigen, wie man den Aktiv-Zustand eines Raumobekts durch einen Skriptbefehl ändert.
Um die (hier bereits eingetragenen) X- und Y-Koordinaten zu bestimmen ist ein wenig Fingerspitzengefühl gefragt. Es kommt tatsächlich bei der Ausrichtung der Objekte auf jeden Pixel an. Die Objekte orientieren sich (dank des HasBase-Interfaces) an der unteren Mitte der Bilddatei.
Die anderen Felder sind selbsterklärend.

RAUMOBJEKTDARSTELLUNG

Im Raum hat sich mit den Eintragungen noch nichts verändet. Das ist kein Wunder, denn es fehlen die Raumobjektinteraktionen und Raumobjektdarstellungen.
So sehen die Einträge in der Tabelle Raumobjektdarstellung aus.

```
200401;20040501;;0;600;800;600
200402;20040601;;0;599;800;599
200403;20040701;;0;599;800;599
200404;20040801;20040103;0;599;800;599
```

Der Reihe nach bedeuten die Felder: 1) ID; 2) RaumobjektID; 3) BildfolgeID (optional); 4) GundlinieX1; 5) Grundlinie Y1; 6) Grundlinie X2; 7) Grundlinie Y2

Nur vier von den Raumobjekten müssen dargestellt werden. Der Alumann (ID = 20040701), die Flasche (20040601), der Walk-behind (20040501), und der Blitz (20040801). Die anderen Raumobjekte wie der Mond oder die Anstalt sind bereits in der Hintergrundgrafik vorhanden, müssen also nicht extra gezeichnet werden. Auch können sie Edna nicht verdecken. Das Feld „BildfolgeID“ ist in allen drei Fällen leer. Nur der Blitz ist animiert. Die Flasche oder der Walkbehind sind nicht animiert. Der Aluman selbst ist zwar animiert, doch er bezieht als NSC diese Bildfolgen von woanders.
Die letzten vier Felder spannen jeweils eine Linie, die „Grundlinie“, auf. Sie legt fest, ob sich Edna vor dem Raumobjekt befindet oder dahinter. Ist Ednas Y-Koordnate unterhalb der Linie, so muss Edna auch vor dem Objekt gezeichnet werden. Weil das in diesem Bild nicht vorkommen kann, sind die Y-Werte der Linien so weit unten wie nur möglich. Das Geländer soll allerdings vor den anderen beiden Objekten gezeichnet werden. Darum sind seine Y-Werte auch um einen Pixel größer als die der anderen beiden.

RAUMOBJEKTINTERAKTION

Der Raum sieht jetzt bereits genau so aus, wie wir ihn haben wollen. Edna steht hinter dem Geländer auf dem jetzt auch der Alumann kniet. Neben ihm steht die Flasche (das „Elexir“?). Doch wir können mit keinem der Gegenstände interagieren. Wir müssen zuerst die Raumobjektinteraktions-Tabelle ausfüllen.

```
200401;20040101;Treppe;730;360;s;b;0;0;0;0
200402;20040201;Bank;626;360;o;a;20040201;20040211;20040221;20040231
200403;20040301;Mond;411;360;n;a;20040301;20040311;20040321;20040331
200404;20040401;Anstalt;411;360;n;a;20040401;20040411;20040421;20040431
200405;20040601;Flasche;411;360;w;n;20040601;20040611;20040621;20040631
200406;20040701;Alumann;626;360;w;r;20040701;20040711;20040721;20040731
```

Die Felder sind hier: 1) ID; 2) RaumobjektID; 3) Bezeichnung „on Mouseover“; 4) WalkToPointX; 5) WalkToPointY; 6)StandByBlickrichtung; 7)Default-Aktion; und dann leider in der nächsten Zeile: 8) AnsehenSkriptID; 9) BenutzenSkriptID; 10) NehmenSkriptID; 11) RedenMitSkriptID

4), 5) und 6) lassen uns den Punkt und ihre Blickrichtung bestimmen, an den sich Edna stellen soll, wenn sie mit dem Objekt interagiert.
7) ist die Vorauswahl für die Interaktion. „a“ steht für „Ansehen“, „b“ für „benutzen“, „n“ für „nehmen“ und „r“ für „reden“. Edna wird also standardmässig beim Anklicken mit der rechten Maustaste mit dem Alumann reden und den Mond Ansehen.
Die Felder 8), 9) 10) und 11) sind besonders wichtig. Hier sind die SkriptIDs eingetragen, über die wir unsere Anwendungslisten in der Tabelle „Skript“ finden werden. Der Treppe sind keine SkriptIDs zugeordnet. Sie soll uns später als Ausgang dienen. Für Ausgänge kann man maximal ein Skript hinterlegen und zwar in Feld 9). Dieses wird dann immer statt des Standard-Raumwechsels ausgeführt, wenn der Spieler den Ausgang anwählt und Edna an seinem WalkToPoint angelangt ist.

AUSGANG

Der Ausgang ist schnell geskriptet:

```
200401;200401;200301;712;286;w
```

Die Felder: 1)ID; 2)RaumobjektinteraktionID; 3)ZielraumID; 4)WalkInPointX; 5)WalkinPointY; 6)Charakterblickrichtung

Wird der Ausgang benutzt, so betritt Edna also den Raum 200301 an den Koordinaten (712 / 286) und guckt nach westen.

BENUTZEMIT

Auch wenn noch keine Skripte mit den eingetragenen IDs in der Skripttabelle hinterlegt sind, so können wir doch jetzt bereits mit jedem Objekt auf die ihm zugedachte Weise interagieren. Mit einer Ausnahme: In unserem Inventar liegt Harvey und wartet darauf, benutzt zu werden. Damit das Programm weiss, welches Skript bei diesen Kombinationen ausgeführt werden soll, muss die BenutzeMit-Tabelle ausgefüllt werden.

```
1001;20040201;2004020101
1001;20040301;2004030101
1001;20040401;2004040101
1001;20040601;2004060101
1001;20040701;2004070101
```

Die Felder: 1) InventarobjektID; 2) RaumobjektID; 3)SkriptID

Wir haben Glück, dass wir uns für dieses Beispiel nur um Harvey kümmern müssen. Eigentlich muss es einen entsprechenden Eintrag für alle möglichen Inventearobjekte geben, die Edna in diesem Raum besitzen kann. So gibt es aber nur jeweils eine Tabellenzeile für jede Kombination Harvey – ROI (Allerdings nicht für die Treppe. Man kann keine Gegenstände mit Ausgängen benutzen).

CHOICELIST

Eine Choicelist stellt eine Auswahl in einem Dialog dar. Sie kann nur über einen Skriptbefehl aufgerufen werden. Jeder Option ist dabei eine SkriptID zugeordnet, die bei der jeweiligen Anwahl ausgeführt wird.

```
20040101;01;true;Dringend! Es geht um Leben und Tod!;2004019001
20040101;02;true;Nicht so dringend.;2004019002
```

Die Felder: 1) ID; 2) Auswahlnummer; 3) Aktiv-Flag; 4) Auswahltext; 5) SkriptID

NSC

In unserem fertigen Raum wollen wir uns kurz mit dem Alumann unterhalten können. Möglicherweise will er ja gar nicht, dass wir ihm seine Flasche entwenden? Dafür ist es nötig, dass er ein vollwertiger NSC ist, mit einem Charakteranimationsset, mehreren Bildfolgen und den dazugehörigen Animationsbildern. Nach diesen vier Tabellen können wir dann endlich zur Königsdisziplin übergehen: Dem Ausfüllen der Skript-Tabelle. Doch zuerst der NSC.

```
20040101;20040701;200401;Alumann;NscFontLind;0;0;0;0
```

Die Felder: 1) ID; 2) RaumobjektID; 3) CharakteranimationssetID; 4)Bezeichnung; 5) Font; 6) VSpeed; 7)Hspeed; 8) YAtZeroScale; 9) YatFullScale

Um 3), das Charakteranimationsset werden wir uns als nächsten kümmern.
6) und 7) entfallen, weil sich der Alumann nicht vom Fleck bewegen wird. 8) und 9) sind beide null. Damit signalisiert man dem Programm, dass der NSC nicht anhand einer alternativ hier einzutragenden Perspektive skaliert werden soll.
5) Hier ist die Schriftart eingetragen, die der NSC beim Sprechen benutzt. Der Alumann spricht in einem beruhigenden Lindgrün. Die zur Verfügung stehenden Schriftarten sind in der GameEngine angelegt.

CHARAKTERANIMATIONSSET

Für jeden möglichen Aktionsmodus wird ein Eintrag in der Tabelle Charakteranimationsset vorgenommen. Der Alumann hat nur zwei Animationen: Eine für „sprechen“ und eine für „warten“.

```
200401;1;AlumannSpricht;20040102;0;0;0
200401;3;AlumannWartet;20040101;0;0;0
```

Die Felder: 1) ID; 2) AktionsmodusID; 3) Bezeichnung; 4) LinksBildfolgeID; 5) RechtsBildfolgeID; 6) VorneBildfolgeID; 7) HintenBildfolgeID

Bei 1) handelt es sich genau um die ID, die wir bei NSC.CHARAKTER-ANIMATIONSSETID eingetragen haben.

Bei 2) steht der Aktionsmodus. Die „1“ in der oberen Zeile steht für den Aktionsmodus „sprechen“, die „3“ steht für den Aktionsmodus „warten“.

Bei 4), 5) 6) und 7) stehen die Bildfolgen für die unterschiedlichen Himmelstrichtungen. Der Alumann schaut allerdings nur nach westen. Darum muss es in diesem Fall auch nur jeweils eine Bildfolge geben.

BILDFOLGE

Es gibt also nur zwei Bildfolgen, die wir für den Alumann in der entsprechenden Tabelle einzutragen haben. Die beiden IDs müssten wir in der vorhergehende Tabelle unter Feld 4) wiederfinden. Ausserdem gibt es noch die Blitz-Bildfolge.

```
20040101;alumann_wait_links;100;true
20040102;alumann_talk_links;200;true
20040103;Blitz;200;false
```

Die Felder: 1) ID; 2) Bezeichnung; 3) Anzeigedauer; 4) Loop

Der Parameter 3) bestimmt, wie lange in Milisekunden ein Frame der Animation stehenbleibt. Die „reden“-Animation wird also etwas langsamer abgespielt als die „warten“-Animation.
Laut 4) handelt es sich nur bei dem Blitz um keine geloopte Animationen.

ANIMATIONSBILD

Dies ist die letzte Tabelle, die wir ausfüllen müssen, bevor wir endlich mit dem richtigen Skripten anfangen können.

```
20040101;20040101;2004/alumann_wait.png;1
20040102;20040102;2004/alumann_talk1.png;1
20040103;20040102;2004/alumann_talk2.png;1
20040104;20040102;2004/alumann_talk3.png;1
20040105;20040103;2004/blitz.png;1
```

Die Felder: 1) ID; 2) BildfolgeID; 3) Pfad zur Bilddatei; 4) Abweichende Anzeigedauer

Wir erkennen, dass die Wait-Animation nicht wirklich eine Animation ist. Sie besteht nur aus einem Bild. Das liegt liegt daran, dass der Alumann eine Taucherbrille trägt, durch die man sein Blinzeln nicht sieht.

SKRIPT

Schliesslich ist es soweit. Wir können endlich die Reaktionen auf die bereits angelegten Ereignisse skripten. Erinnern wir uns kurz, an welchen Stellen wir SkriptIDs eingetragen hatten: Es gab jeweils vier mögliche Interaktionsarten bei den fünf ROIs, die keine Ausgänge waren. Ausserdem gab es für jedes dieser fünf ROIs ein Skript in der BenutzeMitTabelle, das durch die Interaktion mit unserem einzigen Inventarobjekt, Harvey, ausgelöst werden sollte. Zwei weitere Skripte sind in einer Choicelist referenziert. Es gibt also 27 Skripte, die jetzt in dieser Tabelle eingetragen werden können. Damit wir nicht ständig nachgucken müssen, zu welchem Ereignis welche SkriptID gehört, ist es klug, einheitliche Konventionen bei der Vergabe von SkriptIDs einzuhalten. Die vier normalen Interaktionsmodi sind immer achtstellig und enden auf „01“ für „ansehen“, auf „11“ für „benutzen“, auf „21“ für „nehmen“ und auf „31“ für „reden“. Die ersten sechs Stellen der ID werden wieder einmal durch Raum (vierstellig) und Objekt (zweistellig) bestimmt. Die BenutzeMit-Reaktionen sind hingegen zehnstellig. Hier findet sich zusätzlich zu Raum und Objekt auch noch ein zweistelliger Inventarobjektkennzeichner.

Da ich die Bedeutung der Felder in der Skript-Tabelle bereits im letzten Kapitel erläutert habe, verzichte ich an dieser Stelle darauf, es noch einmal zu tun. Lediglich die Funktion der Befehle werde ich im Kommentarfeld, direkt im Skript, erklären. Zur optischen Hervorhebung habe ich die Kommentare mit Pfeilen markiert.

```
20040201;1;say(Nettes PlÃ¤tzchen.);            >>>Bank ansehen
20040211;1;say(Keine Zeit, auszuruhen.);      >>>Bank benutzen
20040221;1;say(Ein wenig sperrig, oder?);     >>>Bank nehmen
20040231;1;say(Ich muss dringend mit meiner Bank reden.);>>>Bank reden
20040231;2;say(SpÃ¤ter.); >>>Diese Skript hat zwei Zeilen!

2004020101;1;say(Was hÃ¤ltst du davon, Harvey?);>>>Bank+ Harvey
2004020101;2;think(*GÃ¤hn!*); >>> mit dem Befehl think() lassen wir
2004020101;3;say(Schon gut.); >>> Harvey reden


20040301;1;say(Das ist der Mond.);              >>>Mond ansehen
20040311;1;say(Soweit reichen meine Arme nicht.);>>>Mond benutzen
20040321;1;say(Soweit reichen meine Arme nicht.);>>>Mond nehmen
20040331;1;say(WU - HUUUUU!);                   >>>Mond reden
20040331;2;say(*heul*);

2004030101;1;say(Was hÃ¤ltst du davon, Harvey?); >>>Mond+ Harvey
2004030101;2;think(*WU - HUUUUU!*);
2004030101;3;think(*heul*);
```

```
20040401;1;say(Endlich bin ich da raus.);        >>>Anstalt ansehen
20040411;1;say(Ich war lange genug da.);         >>>Anstalt benutzen
20040421;1;say(Die will ich nicht mal geschenkt.);>>>Anstalt nehmen
20040431;1;say(Zur HÃ¶lle mit dir, Hort des Grauens!);>>Anstalt reden


2004040101;1;say(Was hÃ¤ltst du davon, Harvey?);>>>Anstalt+ Harvey
2004040101;2;think(Ich bin froh, da endlich draussen zu sein.);
```

Jetzt wird es spannend. Das nächste Objekt ist die Flasche, die wir unbedingt haben wollen.

```
20040601;1;say(Die will ich unbedingt haben!);   >>>Flasche ansehen
20040611;1;say(Dafür müsste sie erst mir gehören...);>>> benutzen
20040631;1;say(Ich will dich unbedingt haben!);>>> Flasche reden

20040621;1;animatesc(05,0500); >>> animatesc: führt eine Animation aus
20040621;2;saynsc(20040701,Ã„hem.);>> saynsc: Der Alumann spricht
20040621;3;say(Oh. Entschuldigung.);
20040621;4;say(Brauchst Du die Flasche etwa noch?);
20040621;5;saynsc(20040701,Ja.);>>> wen nsc sprechen, muss man ihre
20040621;6;say(Und wann bist Du damit fertig?);>>raumobjektID angeben
20040621;7;saynsc(20040701,Niemals.);
20040621;8;changeroiskript(20040601,n,20040622);
```

Edna hat die Flasche zwar nicht bekommen, dafür haben wir aber drei neue Skriptbefehle kennen gelernt: animatesc lässt den SC eine Animation ausführen. Der erste Parameter „05“ entspricht dabei dem Aktionsmodus. Der zweite Parameter „0500“ ist die Dauer der Animation in Milisekunden. In diesem Fall macht Edna für eine halbe Sekunde eine Greifbewegung.

Der zweite neue Skriptbefehl ist saynsc. Damit kann der Skripter einen NSC wie den Alumann sprechen lassen. Man muss allerdings angeben, welcher NSC gemeint ist. Dies geschieht über die RaumobjektID.

Der Befehl changeroiskript bedeutet eine Änderung in der Datenbank. Es soll die Skriptzuweisung eines ROI geändert werden. Die übergebenen Parameter sind erstens die RaumobjektID des zu ändernden Objekts, zweitens das Kürzel des zu ändernden Interaktionsskripts (in diesem Fall „n“ für nehmen) und dritens das neue Skript, das stattdessen ausgeführt werden soll. Würden wir im laufenden Spiel in der Tabelle Raumobjektinteraktion nachgucken, müsste uns auffallen, dass der ehemals so fleissig geskriptete Eintrag unter „NehmenSkriptID“ sich nun geändert hat:

```
200405;20040601;Flasche;411;360;w;n;20040601;20040611;20040622;20040631
```

Die Bedeutung ist, dass der Alumann bei Ednas zweitem Versuch, die Flasche zu stehlen, anders reagieren soll.

```
20040622;1;animatesc(05,0500);     >>> Flasche nehmen
20040622;2;saynsc(20040701,Ã„hem.);
20040622;3;say(Oh. Entschuldigung.);
20040622;4;say(Ich dachte, Du guckst gerade nicht.);
```

Das Skript hat sich zwar geändert, doch wieder ist es Edna nicht gelungen, die Flasche zu bekommen. Aber es gibt noch ein drittes Skript:

```
20040623;1;animatesc(05,0500);
20040623;2;inactivate(20040601);
20040623;3;itemactivate(1060);
20040623;4;say(Jippi! Ich habe die Flasche!);
```

Der Skriptbefehl inactivate setzt nun das Active-Flag der Flasche (über die angegebene RaumobjektID identifiziert) auf false. Damit wird das Raumobjekt nicht mehr in der Szene angeziegt und Edna kann auch nicht mehr damit interagieren.
Der Skriptbefehl itemactivate aktiviert dafür ein Inventarobjekt, dasjenige mit der InventarobjektID „1060“. Es handelt sich um die Flasche, die ab nun im Inventar angezeigt wird.
Doch wie kommt es dazu, dass dieses dritte Skript als Reaktion auf „nimm Flasche“ erfolgt? Der Eintrag in der Datenbank ist noch immer auf das zweite Skript eingestellt. Weil wir nur diesen einen Raum und nur noch ein Raumobjekt übrig haben, liegt die Vermutung nahe, dass die Änderung durch einen weiteren changeroiskript-Befehl in Interaktion mit dem Alumann auftreten muss.

```
20040701;1;say(Das ist mein Mentor, der Alumann.);>>>Alumann ansehen
20040711;1;say(Er wartet auf einen Blitz.); >>>Alumann benutzen
20040711;2;say(Nicht auf jemanden, der ihn vom GelÃ¤nder schubst.);

20040721;1;say(Er kann die Ballance alleine halten.);>>Alumann nehmen

20040731;1;say(Hallo Alu, altes Haus!);   >>>>Alumann reden
20040622;2;say(Darf ich *bitte* Deine Flasche haben?);
20040622;3;saynsc(20040701,Hmm. Wie dringend brauchst Du sie denn?);
20040622;4;choice(20040101);
```

Ach ja - Da war ja noch etwas. Durch den Skriptbefehl „choice“ wird die Choicelist mit der angegebenen ID geöffnet. Ein kurzer Blick zurück verrät uns, dass wir in der angegebenen Choiceliste zwei Optionen haben:

```
20040101;01;true;Dringend! Es geht um Leben und Tod!;2004019001
20040101;02;true;Nicht so dringend.;2004019002
```

Hier sind die beiden möglichen Skripte:

```
2004019001;1;say(Dringend! Es geht um Leben und Tod!;);
2004019001;2;saynsc(20040701,Nichts auf der Welt ist so dringend.);
2004019001;3;activate(20040103);
2004019001;4;animate(20040103);
2004019001;5;inactivate(20040103);
2004019001;6;animatescp(08,8000);
2004019001;7;saynsc(20040701,Oh. Da hab ich mich wohl getäuscht.);
2004019001;8;fadeout(black,4000);
2004019001;9;paramexit(800000,400,300,w);
2004019001;10;fadein(black,0010);
2004019001;11;saysound(400,300,GAME OVER);
```

```
2004019002;1;say(Nicht so dringend.);
2004019002;2;saynsc(20040701,Deine Ausgeglichenheit ehrt dich.);
2004019002;3;saynsc(20040701,Du sollst die Flasche haben.);
2004019002;4;saynsc(20040701,Nimm sie ruhig. Sie gehört Dir.);
2004019002;5;changeroiskript(20040601,n,20040623);
```

Was das untere Skript tut, lässt sich mit Anwendung der oben gewonnenen Erkenntnisse leicht entschlüsseln. Um das obere Skript zu verstehen, benötigt man hingegen etwas Phantasie und Kombinationsvermögen.

Abb. 4.11.: animate(20040103).

4.3 Grafik-Design

4.3.1 Anforderungen aus der Dramaturgie

Der Spieler ist in einem Adventure nicht gezwungen, die Geschichte in einer vorgegebenen Reihenfolge oder in einem vorgegebenen Tempo durchzuspielen. Oft kann er sich sogar aussuchen, ob er sich mit den Figuren des Spiels unterhält oder nicht. Wenn er sich dagegen entscheidet, so enthält das Spiel nur noch die kurzen, monologen Reaktionen des SC. Die Geschichte mit all ihrem Druck und ihren Konflikten muss nun über das Bild erzählt werden. Hier wird die Spannung und die Athmosphäre des Spiels also in entscheidendem Maße mit generiert. Die Illustratoren müssen folglich eng mit dem Dramaturgen zusammenarbeiten. Der Hintergrundillustrator ist Bühnenbildner, der Animator ist Schauspieler.

Die Hintergründe sollten so gezeichnet sein, dass die relevanten Objekte und die Ausgänge gut zu erkennen sind. Wichtige Bereiche müssen optisch hervorgehoben werden. Gegenstände mit denen man interagieren kann sollten haptisch wirken, ohne zu sehr aus dem Hintergrund hervorzustechen. Ausserdem müssen die einzelnen Bilder eines Spielabschnitts eine homogene Stimmung erzeugen. Besonders wichtig ist hierbei die „Spielrichtung“. Es folgen einige kleine Beispiele:

Das Spiel startet in Ednas Einzelzelle. Dieser Ort ist besonders eng und bedrückend gezeichnet, die Perspektive zwängt sich über Ednas Kopf zusammen, als würde man durch ein Fisheye-Objektiv blicken. Der Bildausschnitt erinnert bewusst an die Müllschachtszene in „Krieg der Sterne“, in der sich als klaustrophobischer Super-GAU die Wände auf die Helden zubewegen. Die Spielrichtung ist somit klar: Edna muss hier raus.

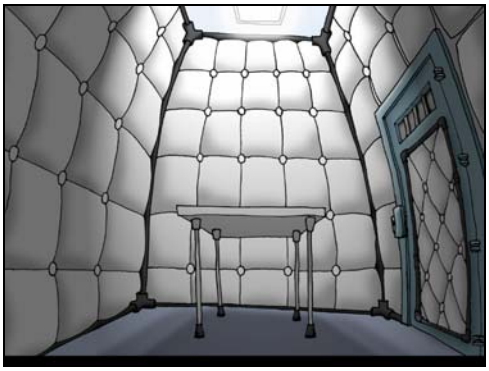

Abb. 4.12.: Ednas Zelle.

Gelingt es Edna aus der Zelle auszubrechen, gelangt sie zunächst in einen Lüftungsschacht. Kriecht sie dort ganz bis zum Ende, kommt sie an ein Gitter, das ihr den Weg versperrt. Das golden einströmende Licht, die fernen Berge und der tiefliegende Horizont im Vergleich zu der dunklen, miefigen Athmosphäre des Luftschachts geben ihr den nächsten Auftrag: Sie muss raus an die frische Luft.

Abb. 4.13.: Das Ende des Lüftungsschachts.

Schafft es Edna schließlich, bis auf das Dach zu gelangen, kann sie ein erstes Mal frisch durchatmen und sich in einem größeren Zusammenhang orientieren. Ein Blick nach unten gibt ihr einen Überblick über den kommenden Weg. Die Anstalt ist dabei fast wie ein Pfeil im Bild arrangiert, die Wolken zeigen wie Finger auf einen Punkt hinter dem Tor. Edna muss runter und raus.

Abb. 4.14.: Die Vorderansicht der Irrenanstalt.

Auf diese Weise wird Edna immer weiter vorangetrieben. Der Hintergrund spiegelt ihren Freiheitsdrang wider.

Dem Animator kommt ebenfalls eine wichtige Aufgabe zu. Er charakterisiert die Figuren des Spiels. Ednas Erscheinungsbild bestimmt zu einem wesentlichen Teil mit, ob der Spieler sich mit ihr identifiziert oder nicht.

Edna sieht ein wenig ungesund aus, ein wenig unbeholfen und wirr, aber entschlossen. Dass sie nur einen weissen Kittel trägt, macht sie verletzlich. Und dass sie ihren Stoffhasen Harvey mit sich herumträgt macht sie irgendwie liebenswert. All das sind Attribute, die den Spieler dazu ermuntern sollen, ihr zu helfen.

Ihr Gesichtsausdruck ist so gewählt, dass er zu möglichst vielen Stimmungen im Spiel passt. Schauspieler reden hier vom „neutralen Gesicht“. Es gibt zahlreiche Studien, in denen einem Publikum derselbe Film von einem neutral spielenden Schauspieler gezeigt wurde. Je nachdem, was für Musik im Hintergrund gespielt, oder an was für andere Szenen er geschnitten wurde, schworen die Zuschauer, dass das selbe Gesicht mal fröhlich, mal traurig, mal wütend oder entsetzt blickte. Da es technisch sehr aufwendig geworden wäre, je nach Stimmung einen anderen Gesichtsausdruck bei Edna auszulösen, der sich dann auf alle Aktionsmodi (gehen, stehen, reden) auswirkt, musste Edna einen solchen Gesichtsausdruck erhalten.

4.3.2 Anforderung aus der Programmierung

Eine ganz andere Art von Auftrag bekommt der Illustrator aus Richtung der Programmierung. Die Räume müssen so gezeichnet werden, dass Ednas Kopf nicht zu nahe am Fluchtpunkt liegt. Der Grund besteht darin, dass Ednas Skalierung sich nach ihrer oberen linken Ecke ausrichtet. Bewegt sie sich in der Perspektive, muss sich ihre Größe ändern. Bleibt bei dieser Neuberechnung ihr oberer linker Punkt auf der selben Y-Koordinate, so kommt es zu seltsamen grafischen Effekten. Edna skaliert sich nur sprungweise.

4.3.3 2d vs. 3d

Wie in Kapitel 2 gezeigt wurde, hat das 3D-Adventure seinen Vorläufer mittlerweile vollständig vom Markt gedrängt. Selbst vor der Kinoleinwand macht diese Entwicklung nicht halt: Disney kündigte 2004 bereits an, dass „Home on the Range“ der letzte 2D-

Disney Film werden würde. Tatsächlich haben 3D–Welten einen grossen Reiz und gelten mittlerweile als Bedingung dafür, ein Computerspiel erfolgreich am Markt zu platzieren. Lichteffekte und freiwählbare Perspektiven bieten neue, spektakuläre Möglichkeiten der Darstellung.

Auf der anderen Seite sind die Anforderungen an die Systemkapazitäten hoch. Um Gewebe, Fell oder Rasen so realistisch darzustellen, wie es in den bekannten Pixar Animationsfilmen der Fall ist, müssen Computer bisweilen Tagelang rechnen. Für ein Spiel, dass die grafische Darstellung in Sekundenbruchteilen ändern können soll, müssen hier noch große Kompromisse gemacht werden. In der Realität besteht die 3D-Welt eines Computerspiels aus Objekten und Figuren, die aus Polygonen zusammengesetzt sind und auf deren einzelnen Oberflächen Texturen dargestellt werden. Das ist optisch nicht unbedingt ansprechender als eine klassische Zeichentrickgrafik. Die Hintergründe sind bei einem 3D-Adventure meist vorgerendert und können daher auch bessere Grafiken bieten als in RPGs oder Ego-Shootern. Das führt allerdings oft dazu, dass sich die Figuren und benutzbaren Objekte nicht in den Hintergrund einfügen.

Die Entscheidung „Edna bricht aus“ im 2D-Look zu gestalten, hat mit all diesen Überlegungen eigentlich wenig zu tun. Es war letztlich meine persönliche Präferenz, die hier den Ausschlag gab. Es ist nebenbei höchst fraglich, ob das erstellen von etwa 150 3D-Hintergründen und 32 Charakteren in der erforderlichen Zeit machbar gewesen wäre.

4.3.4 Comic look vs. Real look

In den späten 80ern und frühen 90ern war aufgrund der geringen Auflösung der Unterschied zwischen Comicgrafik und Realgrafik im Adventure nur gradueller Natur. Die Hintergründe wurden damals noch nicht eingescannt, sondern direkt am Computer erstellt. Je besser die optischen Möglichkeiten wurden, desto mehr kristallisierte sich heraus, dass das Erreichen einer fotorealistischen Grafik in weiter Ferne lag, wohingegen Comicgrafiken immer näher an ihre Printvorlagen herankamen. Die wenigen Versuche, Animationen aus Film- oder Fotoschnipseln zusammenzubasteln sahen allesamt unbeholfen und wenig professionell aus. Die gelungenste Variante war

wohl das Adventure „Toonstruck“(1996), in dem Christopher Lloyd (bekannt als Doc Brown in „Zurück in die Zukunft“) einen Comicautoren spielte, der in seine eigene Comicwelt transportiert wird.

„Edna bricht aus“ ist im Comic look gezeichnet. Die Diskrepanz zwischen den harmlos wirkenden Cartoonfiguren und dem brutalen Ende ist durchaus beabsichtigt.

Es existierte niemals ein Plan, Edna bricht aus optisch nicht als einen interaktiven Comic zu gestalten. Sicherlich wäre es aber interessant, mit den heutigen Möglichkeiten einen Versuch in diese Richtung zu starten.

4.3.5 Die Benutzeroberfläche

Neben der Animation und der Hintergrundillustration ist die Gestaltung der Benutzeroberfläche eine dritte Aufgabe der Layout-Abteilung. Das Interface ist die meiste Zeit des Spieles sichtbar und beeinflusst so das „Look-and-Feel“ des ganzen Spiels.

Eine erste Entscheidung war, die Schaltflächen für die wichtigen vier Aktionen nicht durch Symbole, sondern textuell zu kennzeichnen. „Edna bricht aus“ ist trotz seiner lustigen Grafik kein Spiel für Kinder. Darum bringen schwarzgetrübte Halbtransparenzen ein wenig Ernsthaftigkeit in die Spielewelt mit ein.

4.3.6 Kompromisse

Zum Zeitpunkt der Abgabe sind nicht alle Animationen fertig gestellt. Edna müsste an einigen Stellen zu sehen sein, wie sie Treppen steigt, Leitern hochklettert oder sich vom Wäscheliftsystem fallen lässt. Solche Animationen sind einer der zeitaufwendigsten Arbeitsschritte. In einer späteren Version werden sie nachgereicht werden.

5 Fazit

5.1 Wo hört das Spiel auf?

Wenn man Espen Aarseths Ansicht (vgl. Kapitel **2.2.8**) folgt, dann handelt es sich bei dem dieser Arbeit beiliegenden Computerprogramm nicht um ein Spiel. Es ist „top-down“. Die Handlung ist vorgeplant, wenn auch nicht in jeder denkbaren Situation. Verschiedene Strategien, um das Programm zu Ende zu bringen, stehen nur dann zur Wahl, wenn der Autor verschiedene Lösungswege anbietet. Ob man als Spieler das Programm zum Ende führt, hängt davon ab, ob man den einen richtigen Weg durch das Labyrinth aus Räumen und Rätseln findet.

Es handelt sich bei „Edna bricht aus“ definitiv um eine Erzählung, die sich weit mehr an Strukturen wie „Drama“, „Roman“ oder „Film“ orientiert als an Strukturen wie „Halma“, „Hockey“ oder „Dart“. Das Programm ist dabei nicht weniger interaktiv als ein Spiel. Allein im ersten Raum, Ednas Zelle, gibt es 25 Objekte, die sich auf jeweils vier veschiedene Arten benutzen lassen. Zusammen mit Harvey im Inventar ergeben sich in dieser allerersten Spielsituation bereits 127 verschiedene Interaktionsmöglichkeiten. Ihre Zahl wird mit jedem aufgesammelten Gegenstand und jeder geöffneten Tür wachsen. Im Vergleich dazu hat man bei einer Partie Skat maximal zehn Interaktionsmöglichkeiten pro Spielsituation.

Man kann also nicht sagen, dass Skat dem Spieler mehr Freiheiten bietet. Durch eine Zufallskomponente, das Mischen der Karten, wird hier jedoch ein vom Autoren nicht planbares, chaotisches Element erzeugt. Sind die Karten einmal verteilt, ergibt sich eine unübersichtlich grosse, aber endliche Anzahl von Spielverläufen, die durch die Entscheidungen des Spielers moduliert wird. Bei einem Adventure-Spiel verteilt im übertragenen Sinn der Autor die Karten. Er baut das Spielfeld so auf, dass eine möglichst Spannende Partie für den Spieler entsteht und er programmiert die Reaktionen aller Gegenspieler auf dieses eine Blatt. Würde er eine Partie Skat mit festem Blatt programmieren, müsste er in der ersten Runde zehn Fälle unterscheiden, in der zweiten Runde für jeden dieser zehn Fälle neun u.s.w.. Betrachtet man die Partie als verzweigte Narration, würden sich für die unterste Ebene des Baum-Diagramms 3628800 Fälle ergeben, und das obwohl der Spieler nur noch eine von zehn Karten auf der Hand hält, also gar keine Wahl mehr hat, welche Karte er spielt.

Bei einem Adventure-Spiel sieht der Ereignis-Baum anders aus: Nur wenige der vielen Optionen des Spielers führen tatsächlich auf eine neue Ebene, die Anderen Entscheidungen entpuppen sich als Sackgasse und bringen den Spieler zum vorherigen Knoten zurück. Die weiterführenden Knoten laufen im fortschreitenden Spiel immer wieder zusammen oder führen zumindest zu sehr ähnlichen Spielsituationen. Schliesslich muss der Spieler versuchen, auf die unterste Ebene der Baumstruktur zu gelangen. Auf dem direkten Wege hat er bei „Edna bricht aus“ dabei mehrere tausend Ebenen passiert. Alleine um das erste Stockwerk zu verlassen muss man 76 richtige Entscheidungen treffen, deren Reihenfolge nur grob vorherbestimmt ist. Entscheidet sich der Spieler an einer beliebigen Stelle falsch, so muss er das Spiel nicht von neuem beginnen, sondern kommt lediglich an einen früheren Knotenpunkt zurück.

Solche Strukturen sind in der Spielewelt nicht unbekannt. Man nennt sie Labyrinthspiele. Und auch wenn der richtige Ausgang und die möglichen Auswege bei einem Labyrinth vom Spieleentwickler vorherbestimmt sind, würde nicht einmal Espen Aarseth auf die Idee kommen, diese Kathegorie aus seiner Definition von „Spiel“ auszugrenzen. Bei jedem Action-Spiel, jedem Ego-Shooter und jedem CRPG handelt es sich zunächst einmal nämlich um ein Labyrinthspiel. Sowohl in „Tomb Raider“, „Pac-Man“ als auch in „Doom“ findet sich der Spieler in einem Irrgarten wieder. Das Labyrinth wird hier von Kreaturen bewacht. Im Antiken Vorbild war dies der Minotaurus, bei Pac-Man sind es Geister, bei „Edna bricht aus“ die Wächter der Irrenanstalt. Auch das für das Adventure so typische Hindernis der verschlossenen Tür findt sich in Tomb-Raider und Doom wieder. Die Spiele unterscheiden sich eigentlich nur darin, was für Fähigkeiten der Spieler einsetzen muss, um an den Wächtern und Hindernissen vorbeizukommen.

Würde man bei „Edna bricht aus“ den gesamten Inhalt fortlassen und alle Gegenstände durch Schlüssel oder Schlösser ersetzen, so wäre das Produkt ein reines Labyrinthspiel. Die Handlungs- und Spielelemente überlagern sich ungestört.

„Edna bricht aus“ ist somit dieser angeblich unmögliche Hybrid: Es handelt sich um ein (Labyrinth-)Spiel, das eine vorhergeplante Handlung erzählt.

5.2 Das bessere Erzählmedium

Mit „Edna bricht aus“ ist es mir geglückt, in Form eines Computerspiels eine interaktive Geschichte zu erzählen. Der erzielte dramaturgische Effekt ist, dass sich der Spieler selbst für seine durchlebte Handlung verantwortlich fühlt. Anhand einer unmöglich richtig zu treffenden Entscheidung am höchsten Punkt der Spannung wird er in eine Verantwortung gezwungen, der er mittels der sich bietenden Optionen nicht gerecht werden kann. Er ist also gezwungen, über das Wesen des Mediums zu reflektieren und muss sich zum Beispiel fragen, ob er im Spielverlauf einen Fehler begangen hat. Er muss sich weiterhin fragen, warum er, obwohl ihm die Wahl überlassen wird, nicht in der Lage ist ein zufriedenstellendes Ergebnis zu erzielen. Letztlich muss er sich auch fragen, ob er trotz der vermeintlichen Handlungsfreiheit, schon vorher gezwungen wurde, diesen einen Handlungsweg einzuschlagen. Die ganze Interaktivitätsillusion fällt wie ein Kartenhaus hinter ihm zusammen. Der Spieler hat im Verlauf des Spiels jede Entscheidung bewusst getroffen. Er hat sich diebisch gefreut, dass es ihm gelungen ist, Wächter auszutricksen und Schlösser zu knacken. Je tiefer er in die Handlung einstieg, um so mehr vergaß er, dass dieser Weg von einem Autoren für ihn vorgedacht wurde. Keinen Augenblick lang zog es der Spieler in Betracht, Edna einfach in ihrer Zelle zu lassen. Am Ende der Geschichte schwebt der Mauszeiger über zwei Optionen. Jetzt darf der Spieler zum ersten Mal eine wirkliche Entscheidung treffen und muss sich fragen, ob er das überhaupt jemals wollte.

Denselben dramturgischen Effekt im Rahmen eines nicht interaktiven Mediums zu erzielen, wäre vollkommen undenkbar gewesen. Das Ziel, die Interaktivität des Mediums sinnvoll zum Erzählen einer Geschichte einzusetzen, ist somit erreicht.

Auf der anderen Seite ist der nötige Aufwand, der hierfür betrieben werden musste, ungleich höher als bei anderen Medien. Sowohl der Roman- als auch der Drehbuchautor kommen im Notfall mit Stift und Papier aus, um ihre Geschichte niederzuschreiben. Das Werk eines Computerspielautoren ist alledings nicht linear. Seine Dialoge finden sich in den Skripten wieder, die in einer vom Spieler bestimmten Reihenfolge aufgerufen und modifiziert werden. Will der Autor die Reaktionen dynamisch auf die jeweilige Spielsituation anpassen, so steigert sich der Aufwand fast exponentiell mit jedem weiteren Faktor, der Einfluss auf das entsprechende Objekt nimmt. Um bei den verschiedenen Möglichkeiten nicht die Übersicht zu verlieren, benötigt man ein hohes

Maß an Konzentration, technischen Sachverstand und Abstraktionsvermögen. Eine zweidimensionale Darstellung der möglichen Zustände eines Objekts ist mitunter gar nicht mehr möglich. Letztlich stellt sich heraus, dass die Formulierung in der Skriptsprache oft bereits die einfachste, reduzierteste und übersichtichste Darstellung ist. Eine getrennte Dokumentation der Skripte, wo sie auftreten und wodurch sie beeinflusst werden ausserhalb der Skriptsprache wäre höchst unökonomisch und von sehr zweifelhaftem Nutzen.

Dieses hohe Maß an Komplexität führt dazu, dass der Computerspielautor nicht beliebig genau arbeiten kann. Oftmals müssen die Dialogzeilen die er schreibt auf eine Vielzahl von Situationen anwendbar sein. Das selbe gilt für Animationen. Dialogzeilen, die sich auf bereits geführte Dialoge beziehen, können nur sehr eingeschränkt verwendet werden. Der Aufwand, in jedem Gespräch erst einmal abzuprüfen, was für andere Gespräche bisher stattgefunden haben oder andersherum am Ende eines Gesprächs alle kommenden Gespräche in der Spielewelt zu modifizieren, würde zu chaotischen Resultaten führen.

5.3 Der Computerspielautor

Dass große Teile der Computerspiel-Handlung direkt beim Skripten entstehen, lässt sich nicht vermeiden. Vorformulierte Texte, die von einem nicht programmierenden Autoren zum Skripter gelangen, müssen zwangsläufig beim Skripten angepasst werden. Es ist unmöglich, die Verstrickungen des Spiels ohne die ständige Kontrolle durch das Einfügen in das Programm bis ins Detail durchzuplanen. Es lässt sich sogar behaupten, dass der Hergang, die metalinearen Handlungsabläufe eines Computerspiels in all ihren Zuständen und Effekten eindeutig und logisch zu formulieren, bereits einer Programmierung gleichkommt. Um ein Computerspiel zu schreiben, kommt der Autor also nicht umhin, selbst als Programmierer tätig zu werden. Der Skripter hingegen ist gezwungen, Aufgaben eines Autoren wahrzunehmen. Im Optimalfall sind Autor und Skripter ein und dieselbe Person.

5.4 Die Zukunft des Spiels

Seit der Erfindung des Computerspiels gab es keinen Augenblick des Stillstands in seiner Entwicklung. Jedes neue Produkt bemühte sich, den Vorgänger mit technischen Neuerungen zu übertrumpfen. Dadurch bekommen Computerspiele die Rolle des Zugpferdes in der Entwicklung der Heimcomputer: Allein für Textverarbeitung und E-Mail Korrespondenz wäre es nicht nötig, alle zwei Jahre auf ein neues System umzusatteln. Es sind die hohen Anforderungen von Computerspielen an die Hardware, die die technologische Evolution vorantreiben. Solange es also möglich ist, dass Computer immer leistungsstärker werden, werden Computerspiele verusuchen, die damit verbundenen neuen Möglichkeiten auszunutzen. Mit einem Abklingen dieses ständigen Fortschritts ist vorerst nicht zu rechnen.

Manche von diesen Neuerungen, wie der Sprung von reiner Textdarstellung zu 2D-Grafik oder von 2D- zu 3D-Grafik, eröffnen ein ganz neues Feld an Möglichkeiten und ziehen damit grundsätzliche Veränderungen für optische, dramaturgische und spieltechnische Mechanismen mit sich. Der Wechsel von 2D zu 3D war die letzte große Revolution dieser Größenordnung. Die aktuelle Umwälzung ist die Entwicklung zum Massively-Multiplayer-Online-Spiel.

Die Ludologisten prophezeien, dass sich das Spiel mit zunehmender Interaktivität bald vollständig vom Autoren verabschiedet haben wird. Die massive Abwendung der Spielefirmen vom Adventurespiel und die Reihenweise zu beobachtenden Kündigungen der bekannten Autoren scheinen ihnen hierbei Recht zu geben.

Möglicherweise gilt es aber auch, die nächste Revolution in der Entwicklung der Computertechnologie abzuwarten, bevor man voreilige Schlüsse zieht. Vielleicht kommt es sogar in ferner Zukunft dazu, dass das Entwicklungspotential des Mediums ganz erschöpft ist. Man wird sehen, ob das schliesslich der Untergang des Computerspiels sein wird, oder ob es den Spieleherstellern dann gelingt, eine einheitliche Theorie zu entwickeln, die es ihnen gestattet, die selben Mechanismen wiederzuverwenden, um neue Geschichten zu erzählen.

5.5 Wo ist der Spieler hin?

Es lässt sich feststellen, dass die Konzeption eines erzählenden Computerspiels den Autor mit Aufgaben konfrontiert, die so widersprüchlich sind wie die Standpunkte von Narrativisten und Ludologisten: Für die spielerischen Aspekte ist es nötig, dem Spieler den größtmöglichen Handlungsspielraum zu geben. Der Spieler möchte auf die Welt einwirken, verschiedene Handlungsweisen erproben, was nur dann möglich ist, wenn es ihm erlaubt wird, von einem vorbestimmten Pfad abzuweichen. Für die Handlung des Spiels ist es jedoch erforderlich, diese Freiheiten einzuengen. Nur wenn man als Autor in jeder Spielsituation eine Vorstellung davon hat, in was für einem Stadium des Spieles sich der Protagonist gerade befindet, ist es möglich, einen dramaturgischen Druck auf ihn auszuüben, Spannung zu generieren oder Hinweise über seine nächsten Spielziele einzustreuen.

Man sollte meinen, dass es möglich ist, einen Mittelweg zwischen den beiden sich bietenden Extremen zu wählen. Wenn man bedenkt, dass es den Spieledesignern der 80er und 90er Jahre gelungen ist, beide Aspekte – Spielfreiheit und autorengesteuerte Spielhandlung – zu sehr erfolgreichen und spannenden Produkten zusammenzufügen, so ist schwer einzusehen, warum dem Spieledesigner des 21. Jahrhunderts nur noch folgende zwei Optionen zur Auswahl stehen:

```
Höre auf Janet Murray
Höre auf Espen Aarseth
```

6 Danksagung

Ich bedanke mich bei meinem Pofessor Gunther Rehfeld für seine Motivation und die Möglichkeit, ein so umfangreiches Werk wie das vorliegende Computerspiel im Rahmen meiner Diplomarbeit zu erstellen. Ein weiterer Dank geht an meinen Zweitprüfer Prof. Dr. Norbert Witt, der ebenfalls von Beginn an ein großer Unterstützer meines Vorhabens war.

Das Spiel wäre nicht entstanden ohne die großartige Zusammenarbeit mit Olaf Casper und Felix Engel, die massgeblich an der Programmierung der Eba-Engine beteiligt waren. Ebenfalls möchte ich mich bei Andreas Endler und Stefan Hütter für ihre Beiträge bedanken.

Ein weiterer Dank geht an René Anhaus, der durch seine Mitarbeit am Skripting sicherlich mehrere Wochen Schlaf verloren hat. Er und Erkan Yilmaz standen mir ausserdem mit Rat und Tat bei der Entwicklung der Geschichte zur Seite. In diesem Zusammenhang möchte ich mich auch bei Anne Beutel und Daniela Pusch bedanken.

Ein ganz besonders Dank geht an Finn Seliger für seine genialen Musik-Layouts. Ich wünsche ihm viel Glück bei seiner Diplomarbeit. Ich hoffe sehr, dass er sich mit der Vertonung von „Edna bricht aus“ Gehör verschaffen wird.

Ich bedanke mich bei dem Illustrator Jan Bauer für seine ersten Entwürfe, auch wenn er aus zeitlichen Gründen leider nicht an unserem Projekt teilnehmen konnte, und bei Christoph Bertram für seine selbstlose Mitarbeit am Demo-Spiel.

Ein vorletzter Dank geht an meine Eltern und meine Geschwister, die mich ebenfalls großartig unterstützt haben.

Schließlich möchte ich mich bei meiner Freundin Christina Lange bedanken, die mich durch die letzten Monate gerettet hat (und ohne die ich vielleicht mittlerweile selbst ein Fall für Doktor Marcels Anstalt wäre).

7 Quellen

Aarseth, E.: "Cybertext: Perspectives on Ergodic Literature", John Hopkins University Press, 1997

Aristoteles: „*Poetik"*, Reclam, Ditzingen 2001

Campbell, J.: „Der Heros in tausend Gestalten", Insel Verlag, Frankfurt am Main, 1999.

Field, S.: „Das Handbuch zum Drehbuch", Zweitausendeins, Frankfurt am Main, 1991.

Freud, S.: „Totem und Tabu" (1912), Fischer Taschenbuch Verlag, Frankfurt am Main 1982.

Glassner, A.: „Interactive Storytelling - Techniques for 21st Century Fiction", A K Peters Ltd., Natick, Massachusetts, 2004.

Hagebölling, H.: „Interactive Dramaturgies", Springer Verlag, Heidelberg, 2004.

Ab Hugh, D.: „Doom (1) – Kneedeep in the Dead", Panini Verlags GmbH, Stuttgard 2005.

Israel, J.: „Handlung und Interaktion", Kassel University Press GmbH, Kassel, 2003, Handlungen und Ursachen 32 -39.

*Jung, C.*G.: „Der Mensch und seine Symbole" (1968), Walter Verlag, 1999

Knaack,R.: „WarCraft (1)- Day of the Dragon", Blizzard Entertainment, 2005

Mertens, W.: „Psychoanalyse. Geschichte und Methoden", C. H. Beck, München, 2004

Murray, J.: „Hamlet on the Holodeck", Free Press, New York, 1997.

Shakespear, W.: „Macbeth", Reclam, Stuttgart, 1997.

Sheldon, Lee: „Character Development and Storytelling for Games", Thomson Course Technology PTR, Boston 2004.

Tolkien, J.R.R.: "Der Herr der Ringe", George Allen and Unwin, London, 1954

Whitton, S.: „Sacred–Die Chronik von Ancaria", Panini Verlags GmbH, Stuttgard 2005.

Internetquellen

0001: Wikipedia "Erzählung", 06.06.2006, http://de.wikipedia.org/wiki/Erz%C3%A4hlung

0002: Ursprung des Rollenspiels, 07.06.06 http://www.die3sphaere.de/hort-des-wissens/geschichte-d-r.htm

0003: 8-Bit-Museum, 09.06.06
http://www.8bit-museum.de/

0004: History of Computergames, 09.06.06
http://www.jesperjuul.net/thesis/2-historyofthecomputergame.html

0005: Wikipedia “CRPG”, 10.06.06
http://en.wikipedia.org/wiki/Computer_role-playing_game

0006: Wikipedia “Don Woods” & “Adventure”, 10.06.06
http://en.wikipedia.org/wiki/Don_Woods
http://de.wikipedia.org/wiki/Adventure_%28Spiel%29

0007: Wikipedia “MMORPG” 10.06.06
http://de.wikipedia.org/wiki/MMORPG

0008: Wikipedia “Text-Adventure/Interactive Fiction” 10.06.06
http://en.wikipedia.org/wiki/Interactive_fiction

0009: Wikipedia “Adventure” 10.06.06
http://de.wikipedia.org/wiki/Adventure

0010: Ron Gilbert Interview 92 mit Mix’n’Mojo, 11.06.06
http://www.tentakelvilla.de/interview/rongilbert.html

0011: Ron Gilbert Interview 2004, 11.06.06
http://fm4.orf.at/knoke/170025/main

0012: Old Man Murray: Adventures death, 11.06.06
http://www.oldmanmurray.com/features/77.html

0013: Robert Ripley, 11.06.06
http://www.ripleysf.com/ripley/about/about.html

0014: Tim Shafer: Tod des Adventures, 11.06.06
http://www.gamespot.com/features/question/112999/page2.html

0015: Death of Adventure @ Ludology.org, 11.06.06
http://www.ludology.org/article.php?story=20010630210300000

0016: Death of Adventure from Al Lowe, 11.06.06
http://www.allowe.com/AL/adventuredead.htm

0017: Claude Lévi-Strrauss, 18.06.06
http://www.mnsu.edu/emuseum/information/biography/klmno/levi-strauss_claude.html

0018: Janet Murray: From Game story to Cyberdrama (05.01.2004), 12.06.06
http://www.electronicbookreview.com/thread/firstperson/murray

0019: Espen Aarseth: Espen Aarseth responds (05.01.2004) , 12.06.06
http://www.electronicbookreview.com/thread/firstperson/murray
0020: Die deutsche MUD-Liste, 10.06.06
www.mud.de/dml
0021: Aarseth vs. Murray, 16.06.06
http://grandtextauto.gatech.edu/2004/05/06/cyberdrama-iebri/

Computer- und Videospiele

Discworld, Psygnosis, PC, 1995.
Discworld 2, Psygnosis, PC, 1997.
Discworld Noir, Perfect Entertainment, PC, 1999.
Doom, id Software, PC, 1993.
Full Throttle, LucasArts, PC, 1995.
In 80 Tagen um die Welt, Frogwares, PC, 2005.
Indiana Jones and the Fate of Atlantis, LucasArts (Hal Barewood), PC, 1992.
Indiana Jones and the Infernal Machine, LucasArts, PC, 1999.
Indiana Jones and the Last Crusade, LucasFilm Games (Ron Gilbert), Amiga, 1989.
King's Quest, Sierra On-Line, C64, 1984.
Leisure Suit Larry, Sierra On-Line (Al Lowe), C64, 1987.
The Longest Journey, Funcom / Egmont Interactive, PC, 2000.
The Lost Files of Sherlock Holmes, Electronic Arts, PC (DOS), 1992.
Runaway, Pendulo Studios, PC, 2004.
Sherlock Holmes: Das Geheimnis des silbernen Ohrrings, Frogwares, PC, 2004.
Sherlock Holmes: Mystery of the Mummy, Wanadoo, PC, 2003.
Simon the Sorcerer, Adventure Soft, Amiga, 1993.
The Sims, Maxis / Electronic Arts, PC, 2000.
Space Quest, Sierra On-Line (Scott Murphy Mark Crowe), C64, 1986.
Star Wars: Jedi Knight II, LucasArts, PC, 2002.
Super Mario Bros., Nintendo (Shigeru Miyamoto), NES, 1985.
Syberia, Mindscape, PC, 2003.
The Three Musketeers, Legendo Entertainment, PC, 2005.
Tomb Raider, Eidos Interactive (Toby Gard), Sega Saturn / Play Station, 1996.
Tom Clancy's: Rainbow Six, Red Storm Entertainment (Tom Clancy), PC, 1998.
ZakMacKracken, LucasFilmGames (David Fox, Ron Gilbert), Amiga, 1988.

Anhang 1: Raumpläne

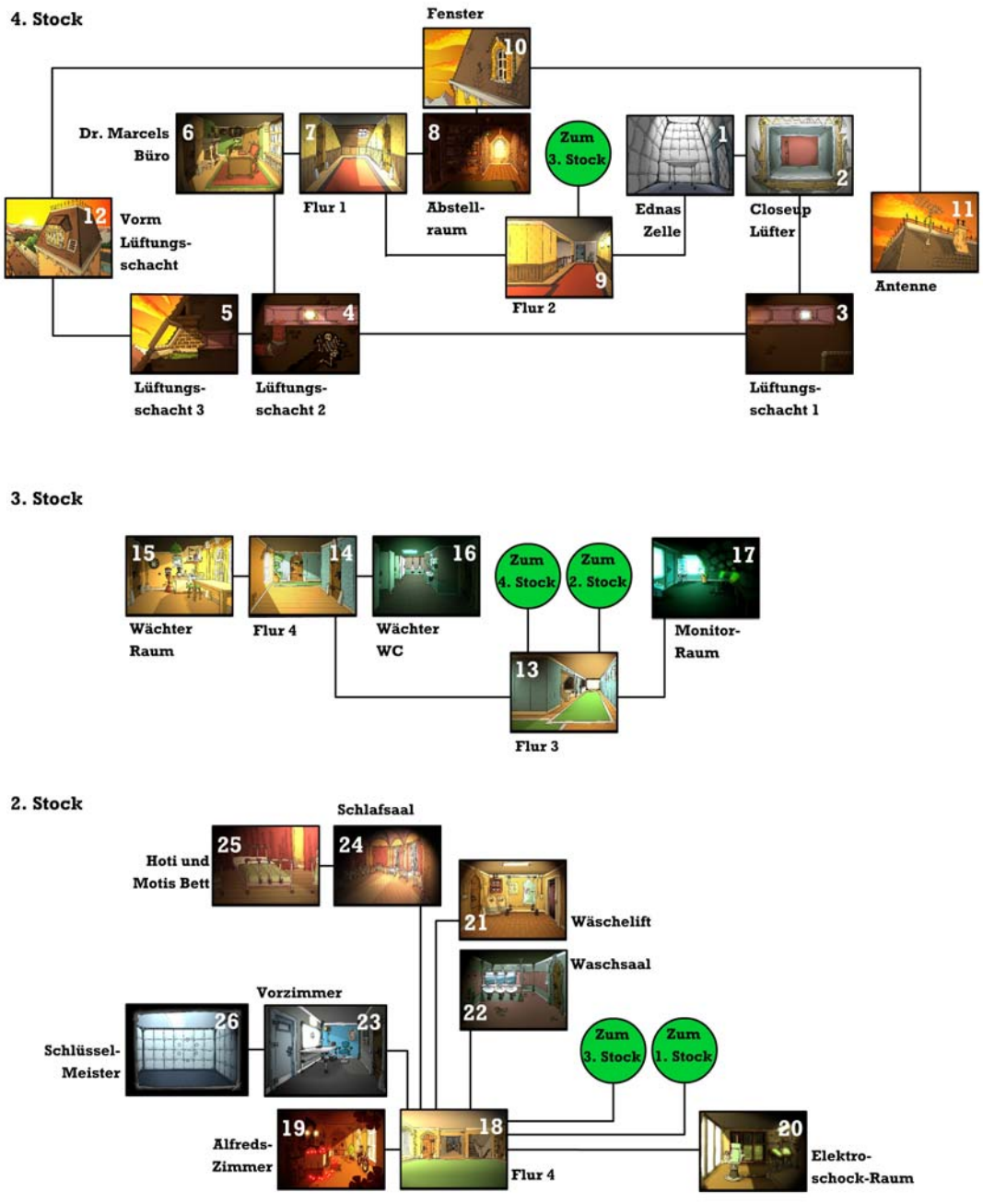

Abb. I.1: Raumpläne Akt 1 – 1

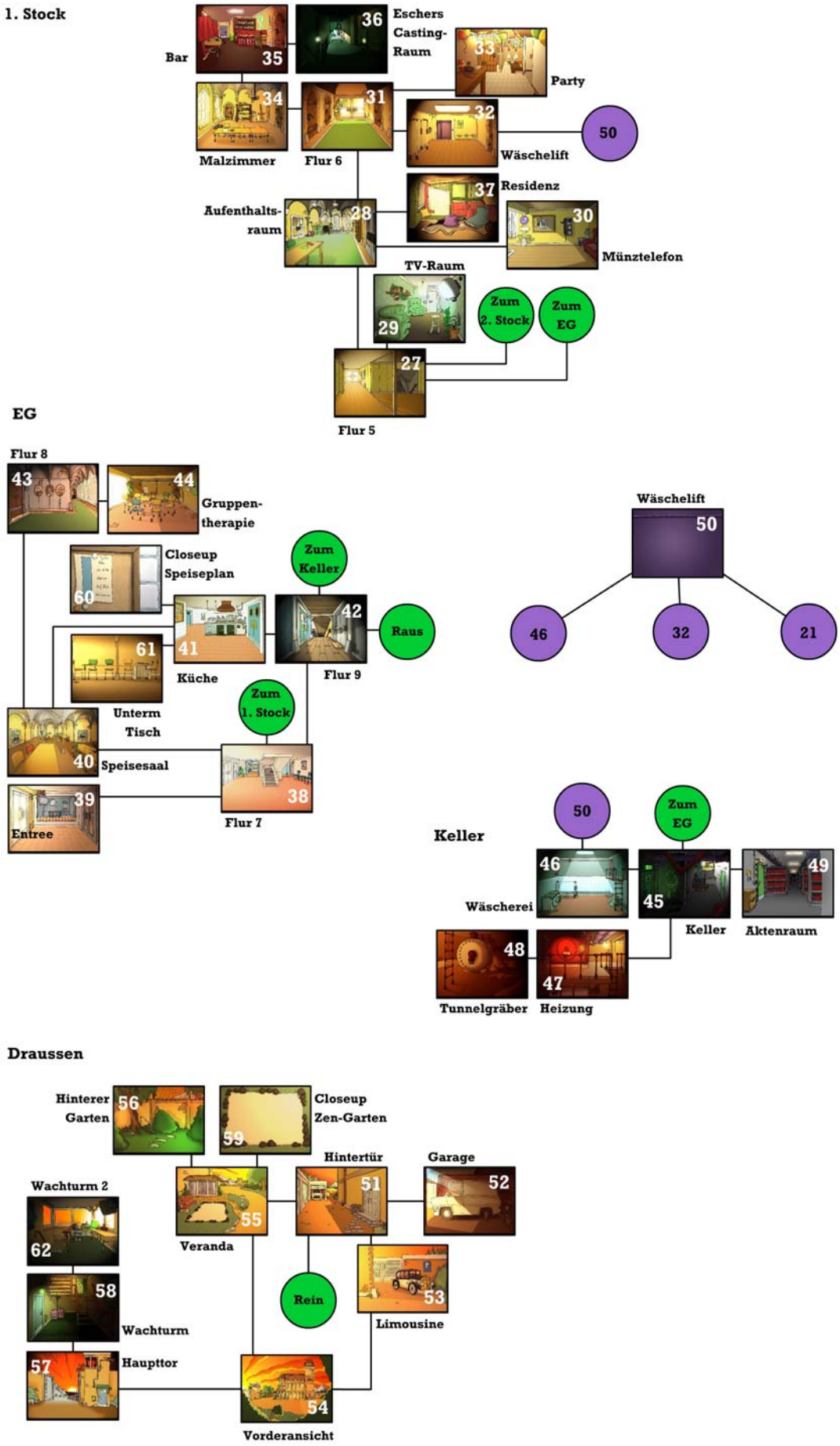

Abb. I.2: Raumpläne Akt 1 – 2

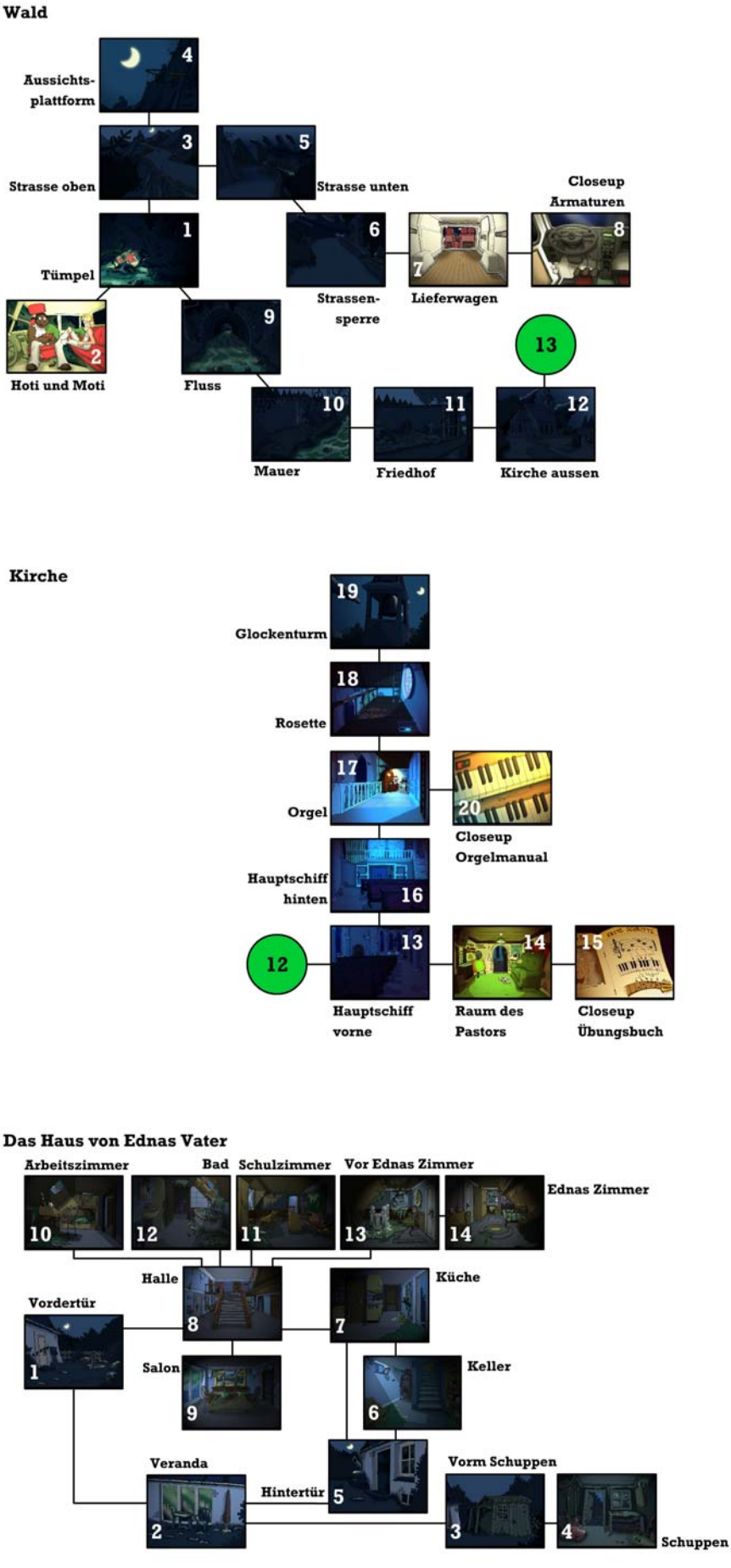

Abb. I.2: Raumpläne Akt 2 und3