HTWM

Hochschule für Technik und Wirtschaft Mittweida

Technikumsplatz 17
09648 Mittweida

Vortrag Datenbanken II

Thema: Erstellung eines rundenbasierten Online-Strategiespiels

Erstellt von:
Isabel Drost (if99wP1)

SendMail@isabel-drost.de
http://www.isabel-drost.de

1 EINFÜHRUNG

1.1 Themenwahl

1.1.1 Warum ein Spiel zur ausschließlichen Benutzung im Internet entwicklen?

In der Gemeinde der Jugendlichen, die von Computerspielen begeistert sind, sind schon seit längerer Zeit Spiele beliebt, mit denen man über Netzwerk gegen Freunde und Bekannte antreten kann. In der Vergangenheit hat sich gezeigt, dass auch die Möglichkeit, über das Internet gegen vorerst unbekannte Mitspieler anzutreten, rege genutzt wird. Dies macht es einerseits einfach, Gleichgesinnte kennenzulernen, andererseits kann man so aber auch einfach Tips zur eigenen Spielweise erhalten.

Ein Spiel rein als Onlinespiel zu entwickeln, stellt, je nach Detailgrad der Story, die nachgespielt werden soll, ein Angebot dar, was dem kommerzieller Softwarefirmen durchaus nahe kommen kann. Allerdings steigt natürlich mit wachsendem Detailgrad auch die Belastung an den Server, der dann eine um so umfangreichere Datenbasis verwalten muß.

1.1.2 Besonderheiten bei der Onlinespiel - Entwicklung

1.1.2.1 Layout

Einen Nutzer für ein Onlinespiel zu interessieren, stellt den Entwickler vor Probleme, die grundsätzlich denen vergleichbar sind, vor denen auch Anbieter anderer Onlineangebote stehen:

- Derjenige, der die Seite aufruft, muß mit Mitteln des Layouts überzeugt werden, dass das Angebot für ihn oder sie interessant sein könnte. Es muß sich also um ein anspruchsvolles, themengerechtes Layout handeln.
- Die Bedienung des Spiels muß intuitiv und ohne das Lesen von Hilfetexten möglich sein. Allerdings muß sie gleichzeitig dem Thema das Spiels gerecht werden.
- Das Layout soll so gestaltet werden, dass es mit im Idealfall plattformunabhängig auf allen gängigen Browsern (Opera[1], Konqueror[2], Netscape 4.7x, IE) lauffähig ist. Alternativ dazu soll mehr als ein Layout entwickelt werden, wobei je nach Browser das passende angezeigt wird.

1.1.2.2 Technische Probleme

- Das Ziel eines solchen Angebots besteht natürlich darin, möglichst viele Nutzer zu begeistern und sie vom Mitspielen zu überzeugen. Allerdings muß auf der anderen Seite für jeden Nutzer eine möglichst hohe Geschwindigkeit bei der Bearbeitung seiner Anfragen garantiert werden können.
- Aufgrund der Tatsache, dass davon auszugehen ist, dass viele Spieler an einem solchen Projekt teilnehmen werden, die entsprechend viel Zeitaufwand in das Spiel stecken, ist eine möglichst hohe Verfügbarkeit des Servers sicher zu stellen. Weiterhin ist darauf zu achten, dass alle Daten der Spieler vertraulich behandelt und vor Angriffen von außen gesichert werden.

[1] http://www.opera.com/
[2] http://www.kde.org/konqueror/

1.1.3 Grundsätzliche Eigenschaften von Unigen[3]

1.1.3.1 Spielkatogorie

Unigen ist vornehmlich als rundenbasiertes Aufbau-Strategiespiel zu betrachten, bei dem Elemente aus Rollenspielen eingearbeitet wurden, um die Kommunikation zwischen den Mitspielern zu erhöhen. Eine Spielrunde verläuft dabei über 23 Stunden, in denen die Spieler ihre Züge einstellen können. Nachts werden diese dann ausgewertet und somit tatsächlich ausgeführt.

Das Ziel des Spiels ist es, einen Planeten mit einer bestimmten Art von Wesen (menschenähnlichen, naturgebundenen, auf die Nutzung von Energie oder auf Kriegsführung spezialisierte Kreaturen) zu besiedeln. Weiteres Ziel ist es, mit anderen Kolonien auf dem eigenen Planeten sowie mit anderen Planeten zu kommunizieren und zu handeln.

Dabei werden diverse soziale Fähigkeiten des Spielers trainiert. Er ist gezwungen, seine Umgebung zu erkunden und mit anderen Mitspielern zu kommunizieren. Weiterhin wird er dazu gezwungen werden, Konflikte zu lösen und unter Umständen Kompromisse einzugehen.

1.1.3.2 Gründe für die Positionierung in eine ferne Zukunft

- Spieler können in der Anfangsphase leicht räumlich voneinander isoliert werden. Dies verhindert einerseits, dass ein neuer Nutzer von zu vielen neuen Eindrücken (Spieler, Bündnisse und deren Relationen untereinander) überrascht wird – er kann sich einfacher und schneller zurechtfinden. Auf der anderen Seite wird es so aber möglich, dass neue Nutzer zuerst im kleinen Kreis an den im Spiel üblichen Umgangston gewöhnt werden, ehe sie mit einem Fehltritt gleich bei allen Mitspielern negativ auffallen.
- Zur Wahl stand außerdem die Positionierung in eine mittelalterliche Umgebung, die allerdings die Möglichkeiten im Spiel historisch begrenzt. Bei der Wahl eines Themas aus dem Fantasybereich sind die Grenzen ebenfalls relativ eng gesetzt (einerseits durch die Erwartungen der Spieler an ein solches Thema, andererseits durch diverse Publikationen zu diesem Thema). Im Gegensatz dazu sind die Spielräume zur Entwicklung von Umgebungen bei der Wahl eines Grundthemas aus dem Science – Fiction Bereich relativ groß.
- Das Thema Science Fiction ist neben Fantasy derzeit relativ beliebt unter Jugendlichen – egal, ob in Form von Geschichten in den Printmedien, als Kinofilm oder als Hintergrundstory zu einem Spiel. Daher sollte die Wahl eines der beiden Bereiche als Grundthema für ein Spiel genügend potentielle Spieler begeistern und anlocken.

1.1.4 Thema und grober Inhalt von Unigen

Die Geschichte, vor deren Hintergrund der Spieler mit seinen Zügen beginnt, handelt davon, dass die Erdbevölkerung in den Weltraum ausgezogen ist und versucht, andere Planeten zu besiedeln.

Mit Beginn der Suche nach anderen Planeten wurden für die auszusendenden Raumgleiter Besatzungen mit verschiedenen Eigenschaften ausgewählt. Diese Eigenschaften haben sich während der langen Suche nach einem neuen Heimatplaneten so stark ausgeprägt, dass dem Spieler vier verschiedene Rassen zur Verfügung stehen, die seine zukünftige Bevölkerung darstellen. Diesen Rassen werden im Spiel dann natürlich auch unterschiedliche Technologien zur Verfügung stehen.

Ziel des Spiels ist der Aufbau einer eigenen Kolonie und Verteidigung dieser gegen Feinde. Feinde werden nicht vom Spiel selbst implementiert. Ihre Rolle übernehmen die anderen Mitspieler. Allerdings können sich unter den Spielcharakteren natürlich auch Freundschaften und Allianzen entwickeln.

[3] http://unigen.nwnt.de/unigen/

1.2 Anforderungsdefinition für Unigen

1.2.1 Vorbemerkungen

Die Muß – Kriterien beziehen sich auf die erste Releaseversion von Unigen. Bei der zur Zeit online verfügbare Version des Spiels handelt es sich um eine Alphaversion. Das heißt, das technische Grundgerüst steht zwar schon, allerdings fehlen einerseits noch einige Funktionen andererseits kann nicht garantiert werden, dass die zugrundeliegenden Scripte sowie die Auswertung vollständig fehlerfrei sind. Die Alphaphase soll jetzt dazu dienen, evtl. Fehler zu finden, sowie das Spiel inhaltlich auszubauen.

1.2.2 Muß – Kriterien

- Unigen soll unter Nutzung von mySQL, PHP und C umgesetzt werden. MySQL soll die dem Spiel zugrunde liegenden Daten beinhalten. Das Datenbankmodell soll einen hohen Grad an Erweiterbarkeit für das Spiel garantieren.
- Bei der Entwicklung des ERM soll vor allem Augenmerk darauf gelegt werden, dass das Modell einerseits den Anforderungen der ersten 4 Normalformen entspricht, andererseits die Ausrichtung auf die Normalformen in den Fällen bewußt außer Acht gelassen wird, in denen es bezogen auf das aktuelle Projekt sinnvoll erscheint.
- Im Datenbankmodell soll sich eine möglichst saubere Trennung zwischen drei Teilgebieten abzeichnen: den Daten, die sich auf die realen Identitäten der Spieler beziehen; den Daten die spielrelevante Definitionen beinhalten und jenen Daten, die von Spielern im Laufe des Spiels erzeugt werden.
- Der Zugriff auf die Datenbank soll in der releasefähigen Version von Unigen vor böswilligen Angriffen gesichert sein – das heißt vor allem, dass die PHP-Scripte, die das Nutzerinterface generieren mit jedweder Art von Angriffen rechnen und diese abfangen müssen.
- Sowohl die Scripte als auch das die Spielzüge regelmäßig auswertende Programm sollen so portabel wie möglich programmiert werden, so dass eine Veränderung der Serveradressen für sie keine Rolle spielt.
- Auf einem Planeten soll es möglich sein, dass sich mehrere Kolonien ansiedeln und entweder friedlich miteinander leben oder kriegerische Aktionen gegeneinander vornehmen. Weiterhin soll es möglich sein, einen fremden Planeten zu bekämpfen und schließlich zu übernehmen.

1.2.3 Kann – Kriterien

- Das Spiel soll alternativ zur reinen HTML – Oberfläche eine auf Java3D basierende Spieloberfläche anbieten.
- Diverse Spielregeln waren nicht vom Beginn der Entwicklungen als zwingend umzusetzen angegeben. Während der Entwicklung war festzustellen, welche Möglichkeiten sich umsetzen lassen und welchen Ideen ein unverhältnismäßiger Aufwand gegenübersteht.

1.2.4 Abgrenzungskriterien

- Das Spiel soll nach Beendigung des Praktikums lediglich als Alphaversion zur Verfügung stehen – das heißt, dass die Game – Engine funktionsfähig ist. Der Inhalt des Spiels (z.B. baubare Gebäude, einsatzfähige Waffen und Raumgleiter) maximal für eine Rasse von Spielern zur Verfügung stehen um die Alphatestphase einzuleiten.
- Das Spiel soll auf einem Server, der PHP und MySQL zur Verfügung stellt, gespielt werden. Es soll sich nicht um eine Standalone – Anwendung handeln, die sich einfach auf dem heimischen Rechner installieren läßt.

1.2.5 Das Zielsystem

Serverseitig wurde Unigen auf einem Cobalt – Raq4 – System getestet, das im Besitz der Firma Network Nation ist. Das Datenbanksystem arbeitet meines Wissens auf einem 100MHz – Rechner. Allerdings wird sich dies unter Umständen bei entsprechend hoher Nachfrage in naher Zukunft ändern müssen. Diese Aufstellung soll verdeutlichen, dass das Spiel serverseitig eigentlich nur einen Rechner braucht, der über PHP4 und mySQL verfügt. Dies ist heute sogar bei recht kostengünstigen Webhostern der Fall.

Clientseitig sind die Anforderungen noch niedriger: Ein einfacher Arbeitsplatzrechner mir Internetzugang (bei der Verbindung kann es sich auch um eine einfach 56kbit – Modemverbindung handeln) sowie einem HTML4-fähigen Browser sollten genügen, um am Spielspaß teilzuhaben.

1.2.6 Der Anwender

Als Zielgruppe von Unigen lassen sich alle Nutzer im Alter von 10 bis 99 bezeichnen. Dabei sind die Altersgrenzen nicht als fest anzusehen. Es ist lediglich vor Beginn des Spieles sicherzustellen, dass der potenzielle Mitspieler im Stande ist, sich an bestimmte Umgangsformen zu halten und die Kommunikationskonventionen im Internet kennt. Außerdem sollte er oder sie in der Lage sein, den eigenen Browser zu konfigurieren und zumindest wissen, was nutzerseitig in etwa unter JavaScript oder einer SessionID zu verstehen ist.

Außer dem technischen Verständnis sollte der potentielle Spieler von Unigen natürlich ein Interesse an Aufbau – Strategie mitbringen.

1.2.7 Copyright

Da Unigen MySQL verwendet und da die Entwickler von Unigen von der Idee „Open Source und GPL“ überzeugt sind, liegt es nahe, das Spiel auch der GPL[4] zu unterstellen.

Die Verwendung ist weiterhin vorerst sowohl für private als auch für kommerzielle Zwecke absolut kostenlos. Sie liegt neben der ausführbaren Version im Quelltext vor, welcher nach Belieben ergänzt und erweitert und zu Teilen oder komplett in anderen Projekten weiterverwendet werden kann. Jede Verwendung der Quellen von Unigen verpflichtet den Programmierer mittels der GNU GPL dazu, das Resultat ebenfalls unter diese Lizenz zu stellen.

Die Verwendung der GPL bedeutet vor allem einen großen Vorteil: Jeder Mitspieler kann für das Spiel Unigen eigene Hilfsprogramme schreiben. Es wurde schon in vielen anderen Onlinespielen gezeigt, dass einige Spieler durchaus Ehrgeiz in diese Entwicklung stecken und die Ergebnisse durchaus auch für andere Spieler interessant sein könnten. Wird für die kleinen Helferlein zum Beispiel auf die Scripte, die derzeit existieren oder auf die Datenbankinhalte zurückgegriffen, so wird der Entwickler verpflichtet, den Quellcode offen zu legen – so kann der zukünftig für die Administration von Unigen verantwortliche diese Programme auf die Legalität ihrer Zugriffe überprüfen und sie auch anderen im Spiel zur Verfügung stellen. Unigen wächst also nicht nur aufgrund des Einsatzes einiger weniger Entwickler, die sich von der Community abschotten, sondern auch durch die Spieler selbst.

[4] http://www.gnu.org/licenses/gpl.txt

2 TECHNISCHE DETAILS VON UNIGEN

2.1 Spielablauf aus Sicht des Systems

2.1.1 Vertikale Gliederung der Entwicklung in zwei Teilbereiche

Bei dem Spiel Unigen handelt es sich um ein rundenbasiertes Aufbaustrategiespiel, wie in der Einführung schon angedeutet. Das heißt, im Verlauf eines Tages stellen die Nutzer ihre Spielzüge ein. Dazu soll ein einfacher Browser, mit dem man die Webseite von Unigen anwählt und sich über einen Login als User identifiziert genügen.
Nach Ablauf eines Tages soll eine Auswertung die Spielzüge letztendlich berechnen und ausführen.
Diese Trennung sollte laut den ersten Entwürfen auch in den letztendlich arbeitenden Programmteilen sichbar werden. Eine Hälfte ist für das Interface, mit dem der Nutzer spielt, verantwortlich. Diese Seite wurde in PHP geschrieben und war möglichst portabel zu halten.
Die zweite Seite ist ein Programm, das täglich einmal die Spielerdaten und Züge auswertet – da während der Auswertung der Server für Zugriffe von außen gesperrt wird, war die Zeit der Auswertung kurz zu halten. Daher wurde hier auf die Verwendung einer Scriptsprache verzichtet und stattdessen C eingesetzt.

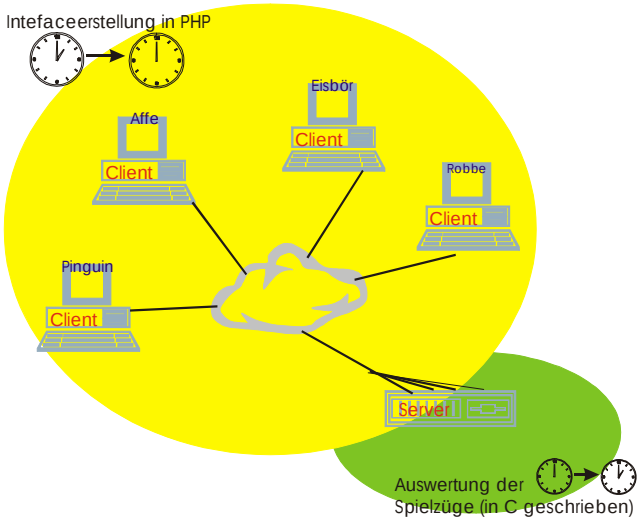

Abbildung 2-1: Vertikale Zweiteilung des Spiels

Bei beiden Teilen erfolgt der Zugriff auf die Datenbank, in der sowohl die Spielzüge gehalten werden, als auch die Userdaten und der aktuelle Spielstand. Dazu jedoch mehr im Kapitel „ERM – das Rückgrat von Unigen“.

2.1.1.1 Generieren von Nutzeroberflächen – die erste Schicht von Unigen

Im Idealfall soll Unigen 23 Stunden am Tag erreichbar sein. In dieser Zeit ist es für den Nutzer möglich, sich anzumelden und seine Spielzüge einzustellen. Dabei wird beim Login eine Session erzeugt, in der Daten über den Nutzer gespeichert werden können, um später schneller auf sie zuzugreifen. Derzeit wird lediglich eine ID in dieser Session abgelegt, über die sich die Identität des Nutzer bestimmen läßt.

Jedem Nutzer ist eine bestimmte Rasse zugeordnet – je nach Rasse wird eine andere Nutzeroberfläche geladen. Derzeit drückt sich das lediglich in verschiedenen Grafiken und StyleSheets aus. Allerdings ist es durchaus vorstellbar, die Unterschiede zwischen den Interfaces auszubauen. Außerdem stehen jeder Rasse andere Gebäude und Waffen zur Verfügung, die produziert werden können.

Die verschiedenen Layouts je Rasse werden derzeit recht einfach realisiert: Für jede Rasse gibt es eigene Scripte, die die Oberflächen generieren. Nun könnte man annehmen, indem man z.B. versucht als Mensch die Scripte der Energiewesen aufzurufen, könnte man nicht nur die Benutzeroberfläche ändern, sondern gleichzeitig Gebäude in Auftrag geben, die eigentlich nur der Energierasse zur Verfügung stehen. Allerdings unterscheiden sich die Scripte nur insofern als sie verschiedene Grafiken ausgeben. Die Selects sind von Rasse zu Rasse nicht verschieden, müssen sie auch nicht, da mit der in der Session enthaltenen Nummer der Nutzer und damit auch die Rasse seiner Kolonie bestimmt werden kann.

Die Scripte wurden in PHP4, dass das Sessionhandling unterstützt, realisiert. Dabei wurde aber darauf verzichtet, die SessionID in einem Cookie auf dem Client-Rechner abzulegen: Die meisten potenziellen Spieler schalten Cookies aus Sicherheitsgründen lieber ab, daher würde bei ihnen ein Unigen auf Cookiebasis nicht funktionieren. Stattdessen wird die SessionID per URL von Script zu Script weitergegeben.

2.1.1.2 Die Auswertung der Züge – die zweite wichtige Schicht von Unigen

Das Programm, dass für die Auswertung verantwortlich ist, wurde in C realisiert. Aufgrunddessen, dass zum Zeitpunkt seiner Entwicklung ein Großteil der Scripte schon fertig und somit die weiter unten erklärte horizontale Gliederung von Unigen bereits ausgeprägt war, konnte dieses Programm weitaus einfacher modular gestaltet werden, als die Nutzeroberfläche.

Die Züge der Nutzer werden nicht Nutzer für Nutzer abgearbeitet, sondern nach Bereichen geordnet. Dabei dient die main-funktion wie üblich lediglich als Steuerfunktion, die nacheinander die Arbeit der einzelnen Module und Funktionen anstößt. Somit sind eventuelle Fehlern im Programm recht einfach einzelnen Funktionen zuzuordnen. Während der Entwicklung zeigte sich die granulare Untergliederung der Aufgaben in einzelne Module als besonders hilfreich, da sie ein umfangreiches Testen jedes einzelnen Abschnitts gemäß dem V-Modell ermöglichte.

2.1.2 Horizontale Gliederung des Spieles in Spielzugarten

Die Spielzüge selbst gliedern sich grob in 4 Teilbereiche, die sich in den Modulen der Auswertung, aber auch in der Aufteilung der Aufgaben auf die Scripte wiederspiegelt:

1) Bau von Gebäuden (Rohstoffproduzierer, Technologien, Kommunikationstationen)
2) Handel mit Mitspielern und Veranstaltung von programmgesteuerten Auktionen
3) Aufbau einer Flotte. Die Flotte ermöglicht den Transport von Gütern mit Raumgleitern. Somit ist auch ein Handel außerhalb von Auktionen einfach möglich. Aber auch die Kriegsführung und Eroberung von feindlichen Planeten setzt eine starke Flotte mit entsprechenden Waffen voraus.
4) Kommunikation mit Mitspielern, derenVoraussetzung der Bau von Kommunikationsstützpunkten (entweder aktiv oder passiv) ist.

Abbildung 2-2: Horizontale Gliederung

Die Aktionen eines Spielers können in diese drei grundlegenden Gruppen untergliedert werden. Entsprechend können auch die Module der Auswertung eingeteilt werden. Weiterhin erleichtert diese Unterteilung das Verständnis des ERM.

2.1.3 Trennung von Daten und Verarbeitungseinheiten von Unigen

In einigen bereits entwickelten Onlinespielen stellt sich die Frage, ob die Trennung von reinen Spieldaten und auswertenden Scripten gelungen ist.

Die Überlegung, diese Trennung bei Unigen einzuführen, ergab sich daraus, dass ein Großteil des Spielinhaltes änderbar sein sollte, ohne auf die Scripte oder die Auswertungsfunktion Zugriff zu haben. Auf der anderen Seite, sollte es aber auch möglich sein, die Scripte gegen neue auszutauschen, ohne darauf achten zu müssen, dass darin eventuelle spielrelevante Daten enthalten sind.

Durch den konsequenten Einsatz der Datenbank zur Organisation des aufkommenden Datenvolumens war es bei der Entwicklung von Unigen recht einfach, von vornherein darauf zu achten, dass sich im Quellcode möglichst nichts aus den Spielregeln wiederfindet, was nicht ebenso in der Datenbankstruktur hätte festgehalten werden können. So finden sich derzeit nicht nur alle relevanten Spielerdaten in der Datenbank wieder. Auch alle Informationen, die gebraucht werden, um die Spielumgebung zu definieren, wurde in ihr abgelegt. So wurde eine eigene Tabelle für Gebäudedefinitionen angelegt, in der je Datensatz genau ein Gebäude mit all seinen spielrelevanten Eigenschaften beschrieben wird.

Allerdings war es aufgrund der Verwendung von MySQL als DBMS zumindest derzeit noch nicht möglich, Constraints anzulegen. Das heißt letztendlich dass trotz aller getoffenen Vorkehrungen in den Selects, die die für den User darzustellenden Datensätze aus der Datenbank holen, Bedingungen enthalten sind. Diese hätten laut Datenbanktheorie eigentlich in der Datenbank selbst gespeichert werden müssen.

2.2 MySQL – Das DBMS zur Verwaltung der Daten von Unigen

2.2.1 Gründe für die Entscheidung für MySQL

Gründe für die Verwendung von MySQL zur Realisierung eines zwar bezogen auf das Datenvolumen recht umfangreichen, bezogen auf die zu erwartende Gewinnspanne im laufenden Betrieb verhältnismäßig kostenintensive Anwendung gibt es viele.
Als Entscheidungskriterien dienten mit Beginn der Entwicklung einerseits die durch die Verwendung des fraglichen Datenbankmanagement Systems entstehenden Kosten, die Geschwindigkeit und Stabilität des Systems sowie seine Verfügbarkeit auf dem Zielserver.
Außerdem wurde die allgemeine Verbreitung in Betracht gezogen. Dabei wurde davon ausgegangen, dass eine hohe Verbreitung automatisch einen großen Kreis von Entwicklern nach sich zieht, die sich über das System und seine Verwendung austauschen und bei Problemen weiterhelfen können. Weiterhin wurde der unterstützte Befehlsumfang betrachtet, der anfangs aber nur eine untergeordnete Rolle spielte, da Unigen als relativ kleines Projekt begann.
Als Alternativen standen sich vor allem MySQL und PostgreSQL gegenüber – Oracle schied aufgrund der fehlenden Finanzierungsmöglichkeiten vorerst aus.

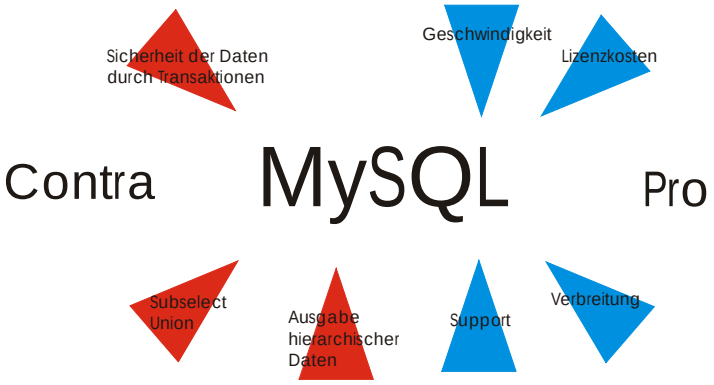

Abbildung 2-3; Vor und Nachteile von MySQL gegenüber anderen DBMS

2.2.1.1 Kosten, die bei der Verwendung des DBMS entstehen

Die Verwendung von MySQL ist grundsätzlich kostenlos. Die Firma erwirtschaftet ihre Gewinne aus dem Support von Firmen bei der Verwendung ihres Produktes sowie durch den Verkauf von MySQL unter Nicht-GPL-Lizenzen. Das bedeutet für die Entwicklung von Unigen, das keine Kosten für die zu verwendente Software entstehen.

2.2.1.2 Support und Dokumentation

Wie oben schon erwähnt, verdient die Firma MySQL AB durch die Unterstützung von Unternehmen beim Einsatz von MySQL.
Auch bei der Entwicklung von Unigen war von vornherein klar, dass stellenweise Support bei der Entwicklung nötig sein würde. Allerdings gibt es für die meisten Fragen Hilfe in der umfangreichen Onlinedokumentation von MySQL. Dies wird ergänzt um die Möglichkeit in Newsgroups wie de.comp.datenbanken.mysql auf konkrete Fragen Antworten zu bekommen.

2.2.1.3 Verfügbarkeit auf Webservern

Das für das Spiel zum Einsatz kommende DBMS sollte möglichst auf einem Webserver verfügbar sein – aufgrund der weiten Verbreitung von MySQL in diesem Bereich, fiel die Entscheidung recht schnell.

2.2.1.4 Geschwindigkeit

Ein letzter Grund für MySQL war die Geschwindigkeit mit der das DBMS arbeitet. Diese konnte bisher noch nicht ausreichend im laufenden Betrieb von Unigen getestet werden. Somit verließ man sich in diesem Punkt stark auf die Aussagen diverser renomierter Fachverlage.

2.2.2 Gründe, die gegen eine Entscheidung für MySQL sprechen

Vor dem Beginn der Entwicklung von Unigen war der Umfang des Spiels sowie der Umfang des ERMs und die daraus folgende Komplexität der zu stellenden Queries noch nicht klar. Während der Entwicklung zeigten sich dann vor allem da Schwierigkeiten, wo Funktionen zu ersetzen waren, die in MySQL noch nicht implementiert sind. Beispielsweise ließen sich aber Sub Selects aufgrund diverser Techniken, die nicht zuletzt der Onlinedokumentation von MySQL entnommen werden konnten, umgehen.

Weiterhin fehlen dem DBMS wichtige Möglichkeiten zu Sicherung der Integrität der in den Datenbanken gehaltenen Daten. Es gibt zwar den Tabellentyp „InnoDB“, der sowohl Transaktionen als auch Foreign Keys unterstützt. Dieser ist aber in der Version von MySQL, die in Form von Binaries zum Download angeboten wird, nicht enthalten. Leider kam genau diese Version auf dem Datenbankserver von Unigen zum Einsatz. Somit gilt für Unigen leider immernoch die Aussage: Es sind keine Transaktionen möglich. Foreign Keys kennt das System dort ebensowenig.

Entgegen der Auffassung diverser Datenbänker wurde ein nicht zu verachtender Teil der Absicherungen demzufolge in die Oberfläche integriert. Allerdings stellt dies aufgrund des Einsatzgebietes ein vertretbares Risiko dar: Nutzer des Spiels haben lediglich über Oberflächen Zugriff auf die Datenbanken, die von Scripten generiert werden. Somit ist es relativ einfach möglich, Angriffe auf die Datenbank schon in diesen Scripten abzufangen.
Erst wenn ein Nutzer direkten Zugriff auf die Datenbank bekommt, eröffnen sich ihm Möglichkeiten, die Daten aus Versehen oder mutwillig zu sabotieren. Dies sollte aber nicht der Regelfall sein – für jede Veränderung, die an den Daten durchgeführt werden soll, werden Scripte entwickelt, die eine Oberfläche bieten, mit deren Hilfe diese Veränderungen einfach durchgeführt werden können.
Hat jedoch ein unauthorisierter Nutzer eine Möglichkeit gefunden, direkt auf die Datenbank zugreifen zu können, nutzen auch die besten Constraintbedingungen nicht mehr – er kann sie mithilfe seines Logins außer Kraft setzen. Um in der Datenbank wirklichen Schaden anrichten zu können, müssen aber nicht nur Nutzername und Paßwort für den Zugang bekannt sein, sondern auch die URL des Datenbankservers – die nicht identisch ist mit dem Webserver, der die Oberflächen von Unigen generiert. Weiterhin wird der Zugriff auf den Datenbankserver von außen natürlich nicht zugelassen und von einer Firewall verhindert: Nur Requests von seiten des Webservers, auf dem die PHP-Scripte laufen werden vom Datenbankserver akzeptiert.

Eine weitere im Befehlsumfang von MySQL fehlende Funktion ist das aus Oracle bekannte connect by prior. Laut FAQ von de.comp.datenbanken.mysql läßt sich diese mithilfe von nestet sets[5] nachbilden. Allerdings wäre diese Funktion lediglich für eine kleine Verbesserung der Bedienoberfläche von Unigen notwendig gewesen. Die Verwendung von nested Sets hätte jedoch einen dafür nicht zu rechfertigenden Umbau der Datenbankstruktur bedeutet.

2.2.3 Zusammenfassung

Das Spiel Unigen ist vom finanziellen Standpunkt her ein relativ großes Risiko: Sein Erfolg hängt nicht nur von seiner korrekten Entwicklung ab. Bevor der Entwurf und die Umsetzung von Unigen begonnen wurden, fanden keinerlei Studien darüber statt, ob und in welcher Form es sich am derzeitigen Markt durchsetzen kann. Es begann als Entwurf einer Spielidee durch einige interessierte Informatiker und wurde schließlich im Rahmen eines Praktikums umgesetzt.

Sein Erfolg hängt allerdings nicht ausschließlich von der Spielidee und ihrer korrekten Umsetzung ab: Derzeit entstehen unzählige Spiele wie Unigen im Web. Der Erfolg hängt dabei davon ab, wie schnell das Spiel an Bekanntheit gewinnt. Weiterhin spielt der Umgang der Nutzer unterenander eine Rolle.
Derzeit läuft das Spiel noch in seiner ersten Testphase – das heißt, um den Ruf eines „instabilen und unreifen“ Spiels von vornherein zu vermeiden, wurde vorerst nur ein kleiner Kreis von Personen zum Testen angesprochen. Außerdem wurden feste Regeln für den Test eingeführt, auf deren Einhaltung möglichst genau geachtet wird, um so zu verhindern, dass sich unter einigen Spielern das Gefühl verbreitet, man habe einen Sonderstatus, in dem man sich mehr erlauben kann, als andere.

[5] http://develnet.org/tech/tutorials/3.1.html

Da das Spiel aber auch unter günstigsten Startbedingungen noch längst nicht zu der vielgepriesenen Goldgrube werden muß, galt es bei der Wahl der Entwicklungswerkzeuge auf einen möglichst geringen Kostenaufwand zu achten. Somit kamen als DBMS sowohl MySQL als auch PostGreSQL in Frage. Bei einer Betrachtung im Nachhinein wäre PostGreSQL wahrscheinlich die einfachere Wahl gewesen – da hier z.B. Transaktionen und Subselects unterstützt werden. Allerdings wog zu Entwicklungsbeginn der Geschwindigkeitsvorteil von MySQL deutlich schwerer.

2.3 Aussagen zum ERM, das Unigen zugrundeliegt

2.3.1 Das „Universe of Discource“ von Unigen

In Unigen müssen einerseits die relevanten realen Daten der Spieler erfaßt werden, um so jeden Spieler eindeutig identifizieren zu können. Dies hat zwei Gründe: Einerseits ist es sowohl für die Betreiber von Unigen als auch für teilnehmende andere Spieler recht ärgerlich, wenn sich einige wenige mehrere Accounts im Spiel anlegen und sich damit evtl. Vorteile im Spiel schaffen. Andererseits soll, sofern Unigen eine ausreichend hohe Beliebtheit erreicht, kostenpflichtig werden – um dies durchzusetzen ist es erforderlich, jede reale Person eindeutig identifizieren zu können.

Weiterhin muß ein virtuelles Universum modelliert werden, in dem jeder Spieler auf einem oder mehreren Planeten residiert und seine Kolonie dort ausbaut. Andererseits muß es möglich sein, auf einem Planeten mehr als nur eine Kolonie anzusiedeln. Das heißt, in der Datenbanken müssen Informationen über die einzelnen Planeten der Spieler sowie ihre Einwohner gehalten werden. Hinzu kommt, dass ein Spieler zum Aufbau seiner Kolonie diverse Gebäude bauen und verschiedene Waffen sowie Raumgleiter produzieren kann. Auch diese Zusammenhänge müssen sich im ERM widerspiegeln.

Den dritten und letzten Teil der Discourcewelt machen die Handlungen der Spieler aus. Sie können ihre produzierten Rohstoffe in einer Auktion zum Verkauf anbieten, können mit anderen Spielern kommunizieren. Weiterhin ist es möglich, Rohstoffe von einem Planeten zu einem anderen zu transportieren oder aber feindliche Planeten anzugreifen.

Schon bei diesem kurzen Abriß über die Diskurswelt von Unigen sollte deutlich werden, dass das logische Schema des Spiels sehr umfangreich werden wird und eine hohe Anzahl von Entity-Sets beinhalten wird. In dieser Dokumentation soll deshalb nur ein kleiner Einblick in die Struktur der Datenbank von Unigen gegeben werden.

2.3.2 Das konzeptuelle Schema von Unigen

Um das Verständnis des ERM zu erleichtern, soll hier eine Einordnung der einzelnen Entitysets in das Gesamtprojekt gegeben werden. Dabei wird deutlich werden, dass sich in dieser Ordnung vor allem die horizontale Gliederung von Unigen widerspiegelt. Das konzeptuelle Schema von Unigen wurde so wie es hier dargestellt wird, soweit möglich in das logische Schema übernommen. Dabei mußte natürlich zum Beispiel auf die Definition von Foreign Keys verzichtet werden, da die vom verwendeten DBMS nicht unterstützt werden.

2.3.2.1 Aufstellung der Entitysets

Die verschiedenen Entiytypes wurden in der untenstehenden Tabelle den in der Beschreibung des Universe of Discourse erwähnten Teilgebiete in Nutzerdaten, Spieldaten und Spielaktionen eingeordnet.

Dabei wird deutlich, dass sich die horizontale Untergliederung von Unigen auch in der Datenbankstruktur widerspiegelt.
Die vertikale Gliederung hat sich nur insofern niedergeschlagen, als es einige wenige Tabellen gibt, die zwar vom User mit Daten gefüllt werden, die bei der Auswertung aber keinerlei Rolle spielen. Dazu gehört z.B. die Kommunikation der Nutzer untereinander, die sozusagen in Echtzeit stattfindet und nicht über ein System eingerichtet wurde, das jedem Nutzer alle Nachrichten erst nach 24 Stunden zuschickt.

Abbildung 2-4; Entitytypes

2.3.2.2 Kurze Erläuterung der Entitysets

Name des Entity-type	Inhalt
	Nutzerdaten
users	Daten über den realen Spieler (z.B. Adresse, Mailadresse, Anmeldedatum)
security_checker	Hier werden die Kennungen der einzelnen SessionIDs abgelegt und dem sie nutzenden Spieler zugeordnet. Dies war nötig, um zu verhindern, dass ein Fremder noch offene Sessions ausnutzen kann, um Zugriff zum Spiel zu erlangen. Der Zeitpunkt, zu dem die Session gestartet wurde wird abgelegt, um abgelaufene Sessions zu erkennen. Weiterhin wird die ID dessen abgelegt, der die Session eröffnet hat sowie die sessionID, die ihm zugewiesen wurde. Somit kann in den PHP-Scripten kontrolliert werden, ob eine ehemals gültige sessionID genutzt wird, um sich nach Ablauf des eigentlich wirksamen Timeouts einzuloggen.

	Spieldaten – Definitionen
build_type	Enhält Daten über die verfügbaren/ baubaren Gebäude sowie Informationen darüber, wer sie bauen kann.
build_require	Enthält Daten über die für den Bau eines Gebäudes nötigen Rohstoffe und Voraussetzungen.
build_influence, build_influence_res, build_cat	Die ersten beiden Entitytypes enthalten Daten über den Einfluß eines Gebäudes auf das eigene Rohstofflager (z.B. erhöht eine Mine verständlicherweise täglich die verfügbare Menge Metall um einen gewissen Betrag). Außerdem werden Informationen darüber gehalten, wie sich der Bau eines Gebäudes auf den Planeten auswirkt. Im letzten Entitytype werden schließlich Informaionen zum Gebäudetyp gehalten (Rohstoffproduzierer, Kommunikationseinrichtung ...).

troop_type	Enthält Daten über die verschiedenen Waffen und Raumgleiter, die der Spieler in seiner Kolonie erstellen lassen kann. Dabei werden auch Werte gespeichert, die die Verteidigungsfähigkeit oder die Kampfkraft einer Waffe bestimmen.
troop_require	Enthält wieder die Voraussetzungsgebäude und für den Bau nötigen Rohstoffe.
troop_class, troop_attack_class	Diese beiden Entitätstypen geben Auskunft über die Art der Waffe (Verteidigungsschild, Laserwerfer) und die Angriffsarten, die der Spieler befehlen kann.
troop_attach_points	Hier werden Informationen gehalten, die Auskunft darüber geben, welche Waffen an einem Raumgleiter angebracht werden können und wie schwer diese sein dürfen.
resources, races, planet_influtype, %_interface	Hier werden Informationen über die im Universum verfügbaren Rohstoffe und Rassen gehalten. Außerdem werden die Arten von Planeteneigenschaften gespeichert, die sich auf die Rohstoffproduktion auswirken. Die letzten beiden Entitytypes enthalten schließlich Daten, die für die Interfaceerstellung interessant sind.

Spieleraktionen	
characters	Informationen über die angemeldeten Spielcharaktere, wie zum Beispiel die Rasse des Spielers oder Informationen über den letzten Loginzeitpunkt.
ownership, planets, planet_influences	In ownership werden die Verbindungen zwischen Charakteren und Planeten hergestellt. Außerdem werden pro Entity Informationen darüber gehalten, wie weit eine Kolonie ihr Gebiet erforscht hat oder wie stark ihre Sende und Empfangsanlagen sind. planets enthält wie der Name schon sagt, einige Informationen über die besiedelten und damit bekannten Planeten, planet_influences Informationen darüber, wie hoch z.B. die Wahrscheinlichkeit für Regen auf dem jeweiligen Planeten ist.
chars_stock	Hier werden die Verbindungen zwischen den Kolonien und den für sie verfügbaren Rohstoffen hergestellt.
chars_build_inProcess, chars_build_ready	Diese beiden Entitytypen enthalten Informationen über die im Bau befindlichen bzw. fertigen Gebäude einer Kolonie.
chars_troops	Wie der Name schon sagt, werden hier die Waffen gehalten, die in Produktion bzw. fertiggestellt sind.
chars_troops_bundles chars_troops_ bundlenames	Der Spieler kann anordnen, dass ein Schiff mit Rohstoffen und Waffen beladen wird oder aber zum Kampf mit Waffen ausgerüstet wird – die fertig bestückten Raumgleiter werden dann mit ihrer Ladung in diesen Tabellen vermerkt.
chars_troops_attacks	Ein so beladenes Schiff kann zum Angriff oder einfach zum Flug zu einem fremden Planeten beordert werden – die entstehenden Aufträge werden hier abgelegt.
chars_repair, chars_demolition	Will ein Spieler beschädigte Gebäude/ Waffen reparieren oder abreißen, so werden diese Aufträger hier abgelegt.
chars_offers, chars_trades	Diese beiden Entitätstypen sollen Handelsangebote aufnehmen, die eine Kolonie in einer Auktion macht sowie die Gebote der Handelspartner. Wird der Handel regulär über diese Auktion abgewickelt, erfolgt der Transport nicht mithilfe eines Raumschiffes sondern unter Einsatz einer von der Entfernung des Handelspartners abhängigen Energiemenge.
commu_threads, commu_postings	Hier wird die gesamte Unigeninterne Kommunikation gehalten. Es gibt eine strikte Trennung zwischen interner und externer Kommunikation um es den Spielern zu erleichtern, Realleben und Spiel nicht miteinander zu verwechseln. Somit sollen intern auch andere Umgangsformen und –regeln gelten als in den externen Foren.

Die Darstellung des ERM von Unigen erfolgt im Anhang. Das ERM kann genutzt werden, um die Beziehungen der Entitätstypen untereinander zu erkennen. Diese Beziehungen konnten letztendlich im logischen Schema nicht mehr explizit über Foreign Keys erhalten werden, da diese in der auf dem zur Verfügung gestellten Webserver MySQL-System noch nicht unterstützt wurden. Dies bedeutet unter anderem, dass die Programmierung von insert und delete-queries stärker abgesichert werden müssen als in einem System, das diese Keys unterstützt.

2.3.2.3 Verstöße gegen die Normalformtheorie

Bei der Entwicklung des ERM von Unigen wurde darauf geachtet, die Normalformen eins bis vier genau einzuhalten um so die Datenredundanz so gering und die Integrität der Daten so hoch wie möglich zu halten. Aufgrund dessen, dass das ERM aber während der Entwicklung des Gesamtprojektes immer neuen Anforderungen angepaßt werden mußte, kann nicht garantiert werden, dass alle Relationen und Entitätstypen noch immer den Normalformen entsprechend angelegt wurden.

Es ist inzwischen zumindest eine Stelle bekannt, die gegen die erste Normalform verstößt. Dabei galt es aber zum Zeitpunkt der Änderung der Datenbankstruktur zu entscheiden, wie schwerwiegend dieser Verstoß gegenüber der Tatsache ist, dass eine Beseitigung gleichzeitig eine Änderung unzähliger Scripte und Auswertungsfunktionen nach sich gezogen hätte. Die Entscheidung fiel zuungunsten der absoluten Sauberkeit des Datenbankmodells aus aufgrund dessen, dass der Verstoß im Sinne des Projektes geringfügig ist und keine Datenanomalien hervorufen kann.

2.3.2.4 Verstöße gegen die Datenbanktheorie

Das Datenbankmodell enthält wenigsten zwei Tabellen, die eigentlich in einer 1:1-Beziehung zueinanderstehen: build_type und build_type_interface. Dabei enthält build_type wirklich spielrelevante Daten, die auch während der Auswertung eine Rolle spielen können. Build_type_interface hingegen enthält Daten, die für die Erstellung der Nutzerinterfaces von Interesse sind: Also Beschreibungen oder Pfade zu diversen Grafiken. Der Grund hierfür ist relativ einfach: Die Beschreibungen können teilweise recht lang werden sind aber meines Erachtens nach für die meisten Abfragen nicht von Interesse. Somit wurden sie aus Performancegründen ausgelagert.

In den Datenbankvorlesungen wurde uns erklärt, man solle im Beleg möglichst auch Fälle dabeihaben, in denen Bilder in einer Datenbank gespeichert werden. Nun, dies ist unter MySQL mithilfe des Datentypes Blob durchaus mehr oder minder problemlos möglich. Allerdings erschien es aus zwei Gründen für dieses Projekt als nicht sehr angebracht. Einerseits läuft das DBMS, das Unigen nutzt, auf einem Rechner, der lediglich über 10 GB verfügt – werden dann viele hochauflösende Bilder in der Datenbank gespeichert, so wären diese recht schnell aufgebraucht. Im Gegensatz dazu sollte der verfügbare Platz auf dem Webserver deutlich größer sein.
Eine Reihe anderer Gründe wurden unter http://develnet.org/tech/dclp/databases.html#databases-3 diskutiert. Dazu gehört zum Beispiel, das MySQL recht ineffizient arbeitet, sobald mehr als ein großes Blob pro Tabellenzeile enthalten ist. Außerdem müßte für jedes einzelne Bildchen nicht nur ein kurzer Pfadname aus der Datenbank geholt werden, sondern ein relativ großes Datenpaket, das die Bilddaten enthält vom Datenbankserver zum Webserver und von da schließlich zum Enduser geschickt werden. Es wird also eigentlich nicht nur die Datenbank größer – auch der Traffic, den das Spielen von Unigen erzeugt, wird erhöht.

2.3.3 Das logische Schema von Unigen

Es wurde versucht, im logischen Schema möglichst genau die Strukturen des konzeptuellen Schemas zu übernehmen. Dabei war die Realisierung durch die Verwendung von MySQL natürlich eingeschränkt: Es konnten keine Foreign Keys definiert werden. Außerdem wurden keine Constraints unterstützt. Diese Features sollen zwar in zukünftigen Versionen implementiert werden, allerdings sind sie aktuell noch nicht verfügbar. Wie oben schon beschrieben werden diese Funktionen zwar durchaus von MySQL in der aktuell stabilen Version unterstützt – allerdings nur, sofern der Nutzer bereit ist, den Quellcode downzuloaden und neu zu kompilieren. Der Betreiber des Datenbankservers von Unigen war dazu leider nicht bereit ☺

Eine Übersicht der Tabellen stellt somit das im Anhang befindliche ERM dar.

3 ENTWICKLUNG VON UNIGEN

3.1 Für die Entwicklung verwendete Materialen

3.1.1 Dokumentationen

- Onlinedokumentation[6] von MySQL
- Newsgroups zum Thema MySQL (de.comp.datenbanken.mysql)
- PHP 4 für Einsteiger und Fortgeschrittene aus dem Hanserverlag
- man – pages zum Thema C und MySQL

3.1.2 Entwicklungswerkzeuge

- ErWIN zur Erstellung des ERM im Vorfeld
- Case/4.0 zur Erstellung der Dokumentation der Scripte
- Editor zum Schreiben der PHP-Scripte und des C-Quellcodes
- gcc zum Kompilieren des Auswertungsprogrammes

3.2 Entwicklerteam

Isabel Drost
(Erstellung des ERM, der PHP-Scripte und der Auswertung in C)

Thilo Fromm
(Arbeit an diversen JavaScript – Funktionen, wesentlicher Anteil am spielerischen Inhalt von Unigen)

An der Idee zu Unigen waren maßgeblich Jan Balzereit, Thilo Fromm, Kjell ..., ... , ... beteiligt.

3.3 Auswirkungen der GPL – Lizensierung auf Unigen

Da Unigen unter der GPL lizensiert werden kann, wird deutlich, dass das die Entwicklung des Produktes nicht von einem Entwicklerteam oder einer Firma allein abhängen soll. Soweit dies möglich ist, sollen Tester, Spieler und Interessierte in den Entwicklungsprozess eingebunden werden und die Möglichkeit erhalten, eigene Ideen einzubringen – und sei es nur bei der Einführung neuer Gebäude und Waffen.

Die Lizensierung eines Produktes unter der GPL wird von vielen mit dem Begriff „Freeware“ verwechselt: GPL ist nicht gleichzusetzen mit kostenlos. Vielmehr bedeutet es lediglich eine Offenlegung des Quellcodes. Derzeit ist jedoch der Trend zu erkennen, das bei vielen GPL-Produkten das Programm selbst zwar kostenfrei ist, der dazugehörige Support aber letztenlich gegen Honorar angeboten wird.

Bei Unigen kommt eine weitere Einahmemöglichkeit hinzu: Das Spiel wird erst durch einen genügend großen Nutzerkreis spannend und interessant. Außerdem ist einiges Wissen nötig, um einen Web- und Datenbankserver zur Verfügung zu stellen, auf dem das Spiel installiert werden kann. Somit ergibt sich für die Betreiber des Onlinespiels durchaus die Möglichkeit, Gebühren zu erheben und bei entsprechendem Erfolg des Spiels di Teilnehmerzahl dadurch kaum zu verringern.

3.4 Zukunft von Unigen

Das Spiel befindet sich derzeit in der Alphatestphase und sucht noch immer nach engagierten Testern. Bis Anfang 2003 sollen für alle Rassen des Spiels Gebäude und Waffen definiert sein. Weiterhin sollen dann auch die noch fehlenden Funktionen (zum Beispiel die Invasion eines fremden Planeten) implementiert sein.
In der Folgezeit wird die Grafik und das Nutzerinterface zu verbessern sowie letzte Programmfehler zu beseitigen sein. Außerdem soll in der Betaphase die Datenbankperformance unter hoher Last getestet und optimiert werden.
Für den Anfang 2004 bis 2005 soll die Betatestphase auslaufen und das Spiel der Community übergeben werden.

[6] http://www.mysql.com/doc/

4 ANHANG

4.1 Verwendete Hilfsmittel

4.1.1 Dokumentation:

- Word 97
- Adobe Acrobat 4.0 zum Drucken des ERM
- ERwin (Erstellung des ERM)
- Corel Draw 8.0 (Erstellung der Grafiken in der Dokumentation)
- Adobe Photoshop zur Erstellung einzelner Logos/ Spielgrafiken

4.1.2 Entwicklung – Designphase:

- Case 4/0 zum Entwurf des Nutzerinterfaces, erster Entwurf eines ERM-Prototypen
- ERwin zum Entwurf des ERM

4.1.3 Entwicklung – Implementierung:

- Midnight Commander Editor (Linux) und edit+ (Windows) als Editor
- gcc zur Erstellung des Auswertungsprogrammes
- MySQL als DBMS (Implementierung und Test der Datenbank großteils über Kommandozeile)
- PHP4 für die serverseitigen Scripte
- JavaScript - und HTML - Technologien für die clientseitigen Scripte
- Apache als Webserver

Zur Recherche wurden zum Beispiel folgende Internetseiten und Newsgruppen genutzt:

http://www.mysql.com
http://www.php.net
news://de.comp.datenbanken.mysql

4.2 Das ERM von Unigen