

|  |  |
| --- | --- |
| Status | Zur Verifizierung freigegeben |
| Name des Projektes | Rocket |
| Projektleitung | J. Eckerle |
| Projektauftraggeber | J. Eckerle |
| Autoren (alphabetisch) | M. Käser, F. Schwab, M. Tschanz |
| Initialen | kasem5, schwf5, tschm23 |
| Genehmigung durch | R. Gasenzer |

Versionen

| Version | Datum | Beschrieb | Autoren |
| --- | --- | --- | --- |
| 1.0 | 13.10.2014 | Initialdokument fertig gestellt | kasem5, schwf5, tschm23 |
| 1.1 | 16.10.2014 | Änderungen im Bereich der Anforderungen | kasem5, schwf5, tschm23 |
| 1.2 | 20.10.2014 | Überarbeitung gemäss Auftrag 2 | kasem5, schwf5, tschm23 |
| 1.21 | 29.10.2014 | Überarbeitung, div. Ergänzungen | kasem5, schwf5, tschm23 |
| 1.3 | 05.11.2014 | Erw. von: Haupt-/Nebenzielen, Usecases | kasem5, schwf5, tschm23 |
| 1.4 | 12.11.2014 | Überarbeitung gemäss Meeting mit R. Gasenzer | kasem5, schwf5, tschm23 |

Table of Contents

[1 Auftrag 2: 3](#_Toc403564566)

[2 Einleitung 6](#_Toc403564567)

[2.1 Ziel des Dokuments 6](#_Toc403564568)

[2.2 Leserkreis des Dokuments 6](#_Toc403564569)

[3 Projektbeschrieb 6](#_Toc403564570)

[3.1 Stakeholderliste 7](#_Toc403564571)

[3.2 Nutzer- und Zielgruppen 7](#_Toc403564572)

[3.3 Projektmethode 7](#_Toc403564573)

[3.4 Technische Ressourcen 7](#_Toc403564574)

[3.5 Dokumente 7](#_Toc403564575)

[4 Projektziele 8](#_Toc403564576)

[4.1 Name und Hauptziel HZ1 8](#_Toc403564577)

[4.2 Teilziele TZ 9](#_Toc403564578)

[5 Scoping 10](#_Toc403564579)

[5.1 Rahmenbedingungen (RB) 10](#_Toc403564580)

[5.1.1 Technisches 10](#_Toc403564581)

[5.2 Systemkontext und Systemgrenzen 11](#_Toc403564582)

[5.3 Out of scope 12](#_Toc403564583)

[6 Anforderungen 13](#_Toc403564584)

[6.1 Quellen und Herkunft 13](#_Toc403564585)

[6.2 Anforderungsliste 13](#_Toc403564586)

[6.3 Funktionale Anforderungen 14](#_Toc403564587)

[6.3.1 Detailbeschreibung der funktionalen Anforderungen 15](#_Toc403564588)

[6.4 Qualitätsanforderungen (Nichtfunktionale Anforderungen) 19](#_Toc403564589)

[6.4.1 Detailbeschreibung der nicht-funktionalen Anforderungen 20](#_Toc403564590)

[6.5 Technische Anforderungen 21](#_Toc403564591)

[6.5.1 Detailbeschreibung der technischen Anforderungen 21](#_Toc403564592)

[7 Chancen und Risiken des Projekts 21](#_Toc403564593)

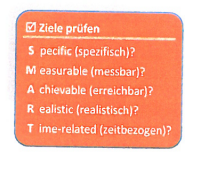
[8 Glossar 22](#_Toc403564594)

[9 Referenzen 23](#_Toc403564595)

[10 Verschiedenes 23](#_Toc403564596)

# Auftrag 1

* Projektziel erarbeiten (Hauptziel, Unterziele  SMART)
* System und Systemkontext abgrenzen / Scope definieren



# Auftrag 2:

Die folgenden Punktewerden in einer Anforderungsliste präsentiert.

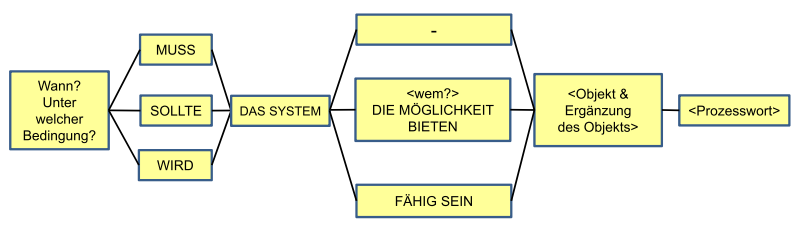
1. **Funktionalen Anforderungen 6+ - natürlichsprachig formulieren**
   1. **4 auf Detailstufe hoch, also Komplexität schon aufgebrochen und**
   2. **2 eher grob, noch mit hohem Komplexitätslevel.  
      Darstellen in einer Anforderungsübersicht und einer Detailbeschreibung**

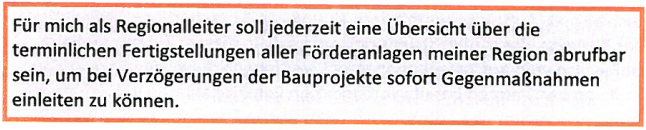
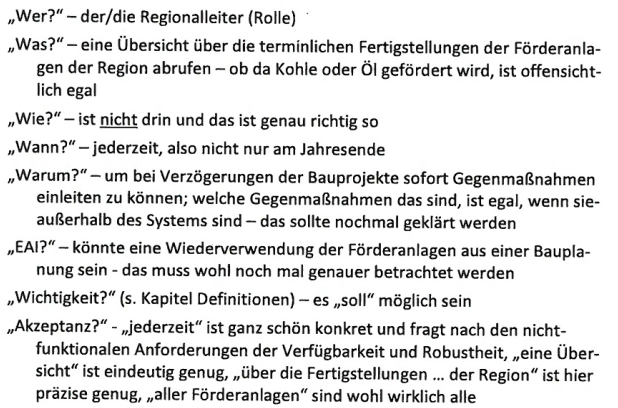
**(wie bereits angefangen) Risikoformel: Noch fakultativ**

Funktionale Anforderungen = Funktionen der Software, aber aus Kundensicht und nicht aus Software-Engineerer-Sicht (nocht nich auf technische Umsetzung eingehen):  
Was soll sie leisten, wie soll sie es leisten und wann soll sie es leisten?  
Auch das „warum“ es das leisten soll, ist hilfreich um das Requirement zu verstehen.

Funktionale Anforderungen stehen in engem Zusammenhang zu den Anwendungsfällen

Anforderung auf dieses Schema hin prüfen:



1. **Nichtfunktionale Anforderungen (Qualitätsanforderungen): Beschreiebn sie mind. 2 nicht-funktionale Anforderungen bei ihrer Themenstellung.**Hier auch Forderungen an das Vorgehensmodell aufschreiben, was gefordert wird. Auch Anforderungen bezüglich Entscheidungen und wer was einbringt und so kann man hier reinnehmen
2. **Eventuell: Technische Anforderungen**

Hardwareanforderungen, SW Anforderungen, hier auch Performance und Leistungsrequirements rein und nicht wie im SED in die NFR.

1. **Randbedingungen  
   Beschreiben sie die Randbedingungen**
   * **Charakterisieren der Aktivitäten des Systems**
   * **(analog Schablone Teil2, S.19 PDF)**

**Überarbeiten der bestehenden Inhalte:**

* Anforderungen beschreiben Funktionen mit denen die Ziele erreicht werden können.   
  WER stellt WARUM die Anforderung? Beim WER die Rolle erwähnen und nicht die Person
* Mit muss, soll, kann/darf schon die Wichtigkeit andeuten.
* Anforderungen messbar formulieren, wenn nötig schreiben wie sie das Gewünschte erreichen können.
* Ergänzen zu den FR/NFR können technische Requirements aufgenommen werden
* Traceability der Anforderungen: Von wem stammen die Anforderungen?
* Nominalisierung OK
* Substantive ohne Bezugsindex – wurden sie erklärt, genau definiert?
* Universalquantoren NOK - Mengen und Häufigkeiten quantifizieren OK
* Unvollständig spezifizierte Bedingungen (if-then nicht ausgearbeitet, nicht auf  
  die Alternativen und anderen Möglichkeiten eingegangen)
* Unvollständig spezifizierte Prozesswörter – wenn wir „übertragen“ verwenden,  
  müssen wir spezifizieren „was“, „wohin“ und „wie“ etwas übertragen wird +

Konstruktion von Anforderungen mittels Satzschablone (Requirements Template)

-> welche Punkte man bei einer Anforderung beachten muss. Syntaktisch eindeutig:

* muss, sollte, wird – Einsatz der Wörter je nachdem, wie die Anforderung ist
* kern der anforderung detailliert beschreiben: bsp. „drucken“
* Funktionale Req.: Wie wird die Aktivität durchgeführt? Durch:  
   Usersteuerung -> Das System muss die Möglichkeit bieten <wem?> ...dies und das zu machen

System(autonom) -> Das System sollte/muss/wird: ...  
Fremde Systeme -> Das System wird fähig sein / wird durch...

Anforderungen beschrieben in: [Buerger - SSWE Anf - Kap 3-4 (Scan)](../../../../Privat/OneDrive/Studium/FH/Semester%205/Projektmanagement/Thema%202/Buerger%20-%20SSWE%20Anf%20-%20Kap%203-4%20(Scan).pdf)

# Einleitung

Das zu realisierende Spiel mit dem Namen „Rocket“ und die in diesem Zusammenhang entstehenden Artefakte müssen den Anforderungen des Projekts gerecht werden.

Die Mitarbeiter am Projekt kommunizieren deshalb fortgehend mit dem Projektverantwortlichen, um die notwendigen Informationen und Anforderungen elaborieren zu können. Auf der Gegenseite sind die Mitarbeiter verpflichtet, die Vorgaben originalgetreu umzusetzen.

## Ziel des Dokuments

Die Ziele und Anforderungen an das Projekt, an die einzusetzenden Technologie und die Hilfsmittel müssen vorgängig dokumentiert werden. Im Verlaufe des Projektes wird man so erkennen können, ob man auf dem richtigen Weg ist und kann abschliessend auch eine sinnvolle Evaluierung durchführen.

## Leserkreis des Dokuments

Das vorliegende Dokument steht der Öffentlichkeit zur freien Einsicht zur Verfügung. Primär richtet es sich an die Dozierenden der Module BTI7082q und BTI7301p, im Rahmen des fünften Semesters der q-Klasse an der BFH in Bern.

# Projektbeschrieb

**Zielsetzung** Ein 3D-Adventure-Spiel mit der Unity-Game-Engine entwickeln

**Technologien**  Unity Game Engine, Blender, Java-Script / UnityScript, Automaten

**Zeitressourcen** 24 Stunden geführter Unterricht, 216 Stunden Selbststudium

Entwicklung einer 3D-Adventure-Game Basis (Minimum an Levels & Funktionen) mit Hilfe der Unity3D Engine. Der Spielzustand und der Zustand der Agenten soll durch Automaten modelliert werden. Das Verhalten der Agenten resp. deren Intelligenz wird über die JavaScript nahe Sprache UnityScript realisiert. Die Agenten passen sich der jeweiligen Spielsituation an. Ein Aufbau von Wissen im Bereich der Automatentheorie ist dafür Voraussetzung.

Für die Spielidee und das Konzept wird dem Team freie Hand gelassen. Rücksprache mit dem Projektleitung bzw. mit dem Projektauftraggeber garantiert, dass die Spielidee und das Konzept den Erwartungen und Vorstellung gerecht werden.

Im Moment (Stand KW: 45) sind noch einige Zielvorstellungen und Anforderungen in Abklärung und müssen noch genauer mit dem Stakeholder geklärt werden.

## Stakeholderliste

* J. Eckerle als leitender Dozent und somit Projektauftraggeber
* Mitarbeitende am Projekt Rocket, namentlich:   
  Martin Käser  
  Fabian Schwab  
  Marcel Tschanz

## Nutzer- und Zielgruppen

* Testpersonen
* Spieler von Computerspielen und Interessierte an unserem Projekt als operative Anwender

## Projektmethode

* Das Projekt wird anlehnend an SCRUM agil entwickelt.

## Technische Ressourcen

* Unity Editor (Entwicklungsumgebung)
* UnityScript, Monodevelop
* Automatentheorie („Artificial Intelligence for Games“)
* Blender (Gestaltung 3D Modelle)
* GitHub (FileShare und Sicherheitskonzept für Projekt)
* Laptops der Projektmitglieder (Testen und Anwendung des Adventures)

## Dokumente

* Projektdokumentation
* Anforderungsdokument
* Projektzeitplan
* Back- / Sprintlog mit den untereinander abhängigen Tasks
* Diary (Kurzbeschrieb der Arbeitsaufwände)
* Arbeitsjournal (Ausführlichere Informationen zu den Tasks)
* Code Repository

# Projektziele

## Name und Hauptziel HZ1

**Projekt ROCKET:** .

Eine spielbare Alphaversion eines First-Person-Adventuregames, dessen Basislevel und die implementierte, künstliche Intelligenz durch selbstgeschriebene Spielklassen offen für Erweiterungen in Komplexität und Umfang bleiben. Eine Kurzgeschichte begleitet den Spieler bei seinen Handlungen und sorgt für zusätzliche Unterhaltung.

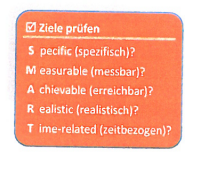
**OLD: Dieses Projekt hat zum Ziel, das Team anhand eines konkreten Auftrags mit der Praxis der Projektführung vertraut zu machen. Zu diesem Zweck entwickelt die Gruppe weitgehend selbständig ein spielbares 3-D-Spiel.**

**Projekt Rocket: Eine an der Berner Fachhochschule entwickelte Basis eines spielbaren First-Person-Adventures, in welchem die Spielintelligenz der Akteure mit hierarchischen Automatenmodellen modelliert wird und dem Entwicklungsteam durch das Erlernen der Werkzeuge Unity3D und C# einen Einblick in die Thematik „Virtual Reality“ ermöglicht. Das Team wird mit passenden Pattern aus der Domäne Spieleentwicklung** **arbeiten, um die spielinterne Kommunikation und die Erweiterbarkeit des Spiels an und für sich sicherzustellen.**

## Teilziele TZ

SMART formulieren – Punkte abdecken, WAS das System leisten soll, aber nicht wie:

|  |  |
| --- | --- |
|  | Teilziele erledigt bis: Release Alpha-Version |
| **TZ** | Die Klassenhierarchie ermöglicht den einfachen Ausbau des Basislevels durch zusätzliche Spielkomponenten wie Räume und Gegenstände und kann durch einen Aussenstehenden mit Erfahrung im Umgang mit Unity ohne zusätzliche Einführung implementiert werden. |
|  |  |
| **TZ** | Das Spiel enthält einen nicht spielbaren Gegner, der auf den Ausgang des Spiels Einfluss nimmt, indem er den Spieler aufsucht und am Verlassen des Levels hindert. |
|  |  |
| **TZ** | Zu jedem Zeitpunkt kann man durch Drücken einer bestimmten Taste das Spiel unterbrechen. Auch das Verlassen und Speichern des Spiels ist zu jedem Zeitpunkt möglich. |
|  |  |
| **TZ** | Durch vom Spieler ausgeführte Aktionen in einzelnen Räumen des Levels werden andere Bereiche freigeschaltet, so dass eine Abhängigkeit zwischen den Einrichtungen besteht. Um den Spielerfolg zu garantieren ist der Besuch jeder dieser Räume notwendig. |
|  |  |
| **TZ** | Die nicht sichtbare, aber menschliche Spielfigur bewegt sich aufrecht gehend durch das Level und der Spieler besitzt die Möglichkeit, seine Figur in zwei verschiedenen Geschwindigkeiten fortzubewegen. |
|  |  |
| **TZ** | Der Spieler hat die Möglichkeit ausgezeichnete Gegenstände, die in seiner Reichweite liegen an sich zu nehmen und in seinem Inventar abzulegen. Das Spielinventar kann jederzeit durchsucht werden. Ein bestimmter Gegenstand ist von zentraler Bedeutung und wird für das Verlassen des Levels benötigt. |
|  | |
|  | Teilziele erledigt bis: Release Betaversion (Ausblick): |
| **TZ** | Die Integrierung von Waffen für Spieler und Gegner ist möglich, ohne an der Klassenhierarchie Änderungen vornehmen zu müssen. |
|  |  |
| **TZ** | Die Aufenthaltsorte der Artefakte im Spiel werden bei jedem Start des Spiels ändern. |



Unterziele definieren unter den SMART Gesichtspunkten:

S Spezifisch Ziele müssen eindeutig definiert sein (nicht vage, sondern so präzise wie möglich).

M Messbar Ziele müssen messbar sein (Messbarkeitskriterien).

A Akzeptiert Ziele müssen von den Empfängern akzeptiert werden/sein (auch: angemessen, attraktiv, abgestimmt ausführbar oder anspruchsvoll[2]).

R Realistisch Ziele müssen möglich sein.

T Terminiert zu jedem Ziel gehört eine klare Terminvorgabe, bis wann das Ziel erreicht sein muss.

# Scoping

## Rahmenbedingungen (RB)

### Technisches

|  |  |
| --- | --- |
| **RB** | Als Programmiersprache ist die objektorientierte Sprache C# zu verwenden |
| **RB** | Zielplattformen sind Windows (ab Windows 7) und MacOS (ab OSX) |
| **RB** | Als Laufzeit-und Entwicklungsumgebung ist das Produkt „Unity 3D“ zu verwenden. |
| **RB** | Als Entwicklungsumgebung ist der Editor MonoDevelop zu verwenden. |
| **RB** | Zustände der Agenten müssen mit Zustandsmaschinen (hierarchisch oder einfach endlich) modelliert werden und müssen sich gegenseitig beeinflussen. |

## Systemkontext und Systemgrenzen

Grobe Architektur des 3-D-Adventures Rocket



## Out of scope

Der Fokus liegt nicht auf der Entwicklung eigener Grafiken, 3D-Modellen oder Audiodateien. Das Spiel soll ausserdem keine Rennsimulation werden. Auch ein Shooterspiel ist nicht Ziel der Entwicklung. Das Spiel enthält kein Mehrspielermodus und kann nicht über das Netzwerk oder über das Internet gespielt werden.

Die Skripte zu den Grundfunktionalitäten was Bewegungen der Spielfigur betreffen werden nicht direkt verändert. An der Konfiguration des Windows-Clients, auf dem das Spiel gespielt wird werden keine Änderungen vorgenommen.

# Anforderungen

## Quellen und Herkunft

Quelle(n) woher die Anforderung stammt:  
JE: Projektauftraggeber Jürgen Eckerle

Team: Projektmitarbeiter im Plenum

Erarbeitet wurden die Anforderungen an Treffen mit J. Eckerle.  
Diese fanden an folgenden Daten statt:

17.09.2014

03.10.2014  
10.10.2014

17.10.2014

24.10.2014

31.10.2014

Wo das Team als Quelle angegeben wird, ist dies als Artefakt der Gruppenarbeit zu interpretieren und wurde jeweils von allen Mitgliedern und dem Auftraggeber validiert.

## Anforderungsliste

Die Anforderungsliste ist ständigen Mutationen unterworfen. Dies aus dem Grund, da noch immer verschiedenste Ideen für die Umsetzung des Games plausibel sind und man sich auch noch nicht über alle zu verwendende Pattern im Klaren ist.

**Legende und ergänzende Hinweise:**

P= Priorität (Skala: 1(low), 2(medium), 3(high))

V= Variabilität

K= Komplexität

R= Aus {P;V;K} "berechnetes" Risiko (Zahlenwert)

Zur Gewichtung der Attribute hinsichtlich der Risikoabschätzung:  
K am "höchsten", V am "zweithöchsten", P am "niedrigsten"

Datum:

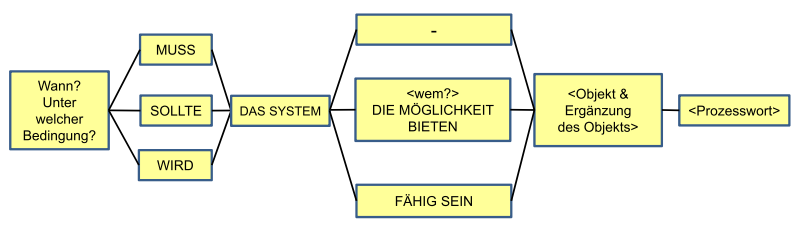
Wann dieses Requirement aufgenommen wurde. Pro Gruppe häufig das gleiche Datum drin, da wir Anforderungen zum gleichen Thema oft am gleichen Tag elaboriert / in Erfahrung gebracht haben.

## Funktionale Anforderungen (Nummerierungen von Hand machen)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nr.** | **Kurzbezeichnung** | **Status** | **P** | **V** | **K** | **R** | **Quelle** | **Datum** |
| 1 | Allgemeines |  | | | | | | |
| 1.1 | Startsequenz bei Spielstart | geplant | 3 | 1 | 1 | low | Team | 01.10.2014 |
| 1.2 | Briefing des Spielers | geplant | 3 | 1 | 1 | low | Team | 01.10.2014 |
| 2 | Leveldesign |  | | | | | | |
| 2.1 | Ein spielbares, durch Wände abgegrenztes Level mit fixem Start und fixem End-Punkt | geplant | 3 | 3 | 2 |  | Team | 24.09.2014 |
| 3 | GUI / Menus |  | | | | | | |
| 3.1 | Spielsteuerung (Tastenbelegung) an eigene Bedürfnisse anpassen können | geplant | 2 | 2 | 1 |  | Team | 01.10.2014 |
| 3.2 | Speichern der Spielstände | erledigt | 1 | 2 | 2 |  | Team | 24.10.2014 |
| 3.3 | Laden der Spielstände | erledigt | 1 | 2 | 2 |  | Team | 24.10.2014 |
| 3.4 | Endsequenz mit Statistik | geplant | 2 | 3 | 2 |  | Team | 08.10.2014 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4 | Spielfigur |  | | | | | | |
| 4.1 | Erweiterter Sichtradius durch Spezialkamera:  „Um-die-Ecke-Sicht“. | geplant | 3 | 2 | 2 |  | Team | 10.10.2014 |
| 4.2 | Gegenstände in Inventar aufnehmen | geplant | 3 | 1 | 2 |  | Team | 10.10.2014 |
| 5 | Gegner (Roboter) |  | | | | | | |
| 5.1 | Robotergegner patrouilliert durch das Level | erledigt | 3 | 2 | 3 |  | Team | 03.10.2014 |
| 6 | Interaktionen Gegner <-> Spieler |  | | | | | | |
| 6.1 | Robotergegner macht Jagd auf den Spieler | geplant | 1 | 3 | 3 |  | Team | 03.10.2014 |

### Detailbeschreibung der funktionalen Anforderungen



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Detaillierte Beschreibung der funktionalen Anforderungen** | **P** | **V** | **K** | **R** |
| 1 | Allgemeines |
| 1.1 | **Startsequenz & Briefing**  Wird das Spiel *Rocket* über den Aufruf der game.exe-Datei gestartet (eine andere Möglichkeit ist nicht vorgesehen), erscheint ein Willkommensbildschirm und eine anschliessende Sequenz zum Briefing des Spielers. Darin wird dem Spieler auf einer einzelnen Bildschirmseite in textueller Form erklärt, was das zu erreichende Hauptziel des Basislevels ist. Dem Spieler wird zudem die Situation erklärt, in welcher sich die Figur beim Start des Spiels befindet. Diese Einführung ist wichtig um die Handlung innerhalb des Spiels nachvollziehen zu können.  Der Willkommensbildschirm mit der Briefingsequenz wird durch den Spieler über die Option „OK“, die als Button am unteren Rand verfügbar ist verlassen, worauf er ins Hauptmenu des Spiels gelangt.  Auf eine ansprechende Spielstory wurde vom Stakeholder JE Wert gelegt, da in der heutigen Zeit die Beliebtheit von Spielen alleine auf das Spielkonzept und dessen Story zurückgeführt werden kann.  NEU:  Sobald das Spiel über den Aufruf der game.exe gestartet wurde, wird die Startsequenz geladen. Diese Startsequenz blendet während drei (3) Sekunden nebst dem Spielelogo auch den Titel des Spiels auf dem Bildschirm ein. Diese Sequenz kann durch den Spieler nicht beeinflusst werden. Nach Ablauf der drei (3) Sekunden wechselt das System ins Hauptmenu des Spiels. | 3 | 1 | 1 | low |
| 1.2 |  |  |  |  |  |
| 2 | Leveldesign |
| 2.1 | Die spielbare Basisversion des Spiels umfasst ein einziges Level.  Dieses durch die Räumlichkeit begrenzte Level erstreckt sich über eine einzige Ebene.  Innerhalb dieses Levels müssen mindestens 3 verschiedene Räume besucht werden können.  Der Spieler beginnt das Level bei jedem Neustart vom selben Startpunkt aus. Auch der Ort, an dem die Spielfigur das Level verlässt wird einzigartig sein.  Bezüglich der Gestaltung der Texturen und der einzelnen Räume wird dem Team vom Stakeholder freie Hand gelassen. Der Einsatz der Software Blender wird empfohlen. |  |  |  |  |
|  |  |  |  |  |  |
| 3 | GUI / Menus |
| 3.1 | Einstellungsmöglichkeiten (1):  Im Hauptmenu hat der Spieler die Möglichkeit seiner Spielfigur einen Namen zu geben und Soundeffekte ein- und auszuschalten. |  |  |  |  |
| 3.2 | Einstellungsmöglichkeiten (2)  Einstellungen zur grafischen Auflösung im Hauptmenu, so wie die Möglichkeit Speicher- und Ladevorgänge über ein Pausenmenü zu starten. |  |  |  |  |
| 3.3 | Erreicht der Spieler mit seiner Figur das Ende des Spiels unterbricht der Spielfluss und es wird nicht weiter auf Eingaben reagiert. Eine Nachricht informiert den Spieler darüber, dass das Level erfolgreich abgeschlossen wurde. In der Endsequenz wird dem Spieler eine Statistik zur erreichten Punktzahl präsentiert. Am Ende der Sequenz d.h. nach dem Verlassen des Statistikbildschirms hat der Spieler die Möglichkeit haben, das Level erneut zu meistern. |  |  |  |  |
| 4 | Spielfigur |
| 4.1 | Grundlegende Steuermöglichkeiten für die Spielfigur müssen implementieren werden (WASD). Dies bedeutet, dass der Spieler fähig sein muss, nach vorne, nach hinten und auf beide Seiten zu navigieren. Mindestens zwei verschiedene Gangarten sollen realisiert werden. Eine langsamere für das Erkunden des Levels und eine schnelle, falls er vor einem Gegner flüchten muss. Optional ist dabei die Auswirkung auf die Akkustik (leise Schritte bei schleichender Fortbewegung und laute Schritte beim gehen/laufen). Ebenfalls Optional ist die Reaktion der Gegner auf wahrgenommene Geräusche, die von einem sich fortbewegenden Spieler stammen.  Der Spieler muss die Möglichkeit haben via Kamerasteuerung einen Blick „um-die-Ecke“ zu werfen, um Gegner auszumachen, und/oder sich einen sicheren Überblick über den weiteren Verlauf der Spielewelt zu verschaffen. |  |  |  |  |
| 4.2 | Die Interaktion durch die Spielfigur mit vordefinierten Gegenständen ist eine zentrale Funktion des Spiels und für den Spielerfolg unabdingbar. Aktionslabels signalisieren im Spiel die Möglichkeit zur Interaktion mit Gegenständen.  Erreicht der Spieler den Aktionsradius eines interaktiven Elements, soll er nach einem Tastendruck über ein erscheinendes Aktionsmenu in der Lage sein, zwischen den möglichen Aktionen auswählen können. Einige Gegenstände setzen für ihre Benutzung Errungenschaften voraus. Auf diese Prämissen wird der Spieler beim Briefing hingewiesen. |  |  |  |  |
| 5 | Gegner |
| 5.1 | Die Gegnerische Figur dient der komplexität des Spielgeschehens. Das lösen der Spielaufgabe wird durch das patrouillieren des gegnerischen Objekts erschwert. Der Gegner bewegt sich auf einem vordefinierten Pfad durch das Level.  Das Ziel des Gegners ist die Sicherung der Räume rsp. das Aufspüren der Spielfigur.  *Wird der Spieler aufgespürt, wird eine noch nicht definierte Aktion ausgeführt.*  *Um das Gameplay interessanter zu gestalten und taktische oder story-bedingtes Vorgehen abbilden zu können, könnte eine Projektilwaffe für den Gegner realisiert werden.*  Die Gegnerische Figur kann vom Spieler nicht kontrolliert oder zerstört werden.  Die Bewegunen des Gegners resp. dessen Aktionsradius soll so modelliert sein, dass dem Spieler eine faire Chance bleibt, dem Gegnern auszuweichen oder sich vor Ihnen verstecken zu können. Ein Teil der Gegnerintelligenz macht dessen Patrouillenverhalten aus. |  |  |  |  |

|  |  |
| --- | --- |
| 6 | Interaktionen Gegner <-> Spieler |
| 6.1 | **Jagd auf den Spieler**  Der Gegner reagiert reagiert ausschliesslich auf Reize, verursacht durch die Spielfigur. Nur so kann er von seinem vorbestimmten Patrouillienpfad abweichen.  Der Gegner reagiert auf visuelle und akkustische Signale in einem vordefinierten Radius. Sein einziges Zielobjekt ist die Spielfigur.  Ist der Gegner durch ein Signal alarmiert worden, soll er die Signalquelle ausmachen und die Umgebung in einem zu definierenden Radius genauer untersuchen.  Solange Sichtkontakt zwischen dem Gegner und der Spielerfigur besteht, wird der Gegner die Spielfigur verfolgen. Die Verfolgung der Spielfigur durch den Gegner ist so zu gestalten, dass dem Spieler eine faire Chance bleibt, dem Gegner zu entkommen. |  |  |  |  |

## Qualitätsanforderungen (Nichtfunktionale Anforderungen)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nr.** | **Kurzbezeichnung** | **Status** | **P** | **V** | **K** | **R** | **Quelle** | **Datum** |
| 1 | Selbststudium |  | | | | | | |
| 1.1 | Unity 3D | erledigt | 3 | 1 | 2 |  | JE | 17.09.2014 |
| 1.2 | Einsatz von Automatentheorie im Game Design | in Arbeit |  |  |  |  | JE |  |
| 1.3 | JavaScript und C# | offen |  |  |  |  | JE |  |
| 2 | Konzeptionelles |  | | | | | | |
| 2.1 | Entwurf einer Spielidee mit Hintergrundinformationen zur Spielsituation | erledigt | 3 |  |  |  | JE | 17.09.2014 |
| 2.2 | Basis für Weiterentwicklungen | in Arbeit |  |  |  |  | JE | 24.09.2014 |
|  |  |  |  |  |  |  |  |  |
| 3 | Inhaltliches |  | | | | | | |
| 3.1 | Das Spiel wird in englischer Sprache entwickelt | offen |  |  |  |  | Team |  |
| 3.2 | Spielkonzept & eine Spielstory werden beim Start des Spiels vermittelt | erledigt |  |  |  |  | Team |  |
|  |  |  |  |  |  |  |  |  |
| 4 | Rechtliches |  | | | | | | |
| 4.1 | Urheberrechte (Logos, Texturen etc.) sind zu berücksichtigen |  |  |  |  |  |  |  |
| 4.2 | Protokollierung der Quellen der verwendeten Objekte |  | 2 |  |  |  | Team | 01.10.2014 |
| 5 | Dokumentarisches |  | | | | | | |
| 5.1 | Updatefunktione: Export und Import der erstellten Levels in Unity realisieren und dokumentieren. |  | 2 |  |  |  | Team | 08.10.2014 |

### Detailbeschreibung der nicht-funktionalen Anforderungen

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Kurzbezeichnung** | **P** | **V** | **K** | **R** |
| 1 | Selbststudium |
|  |  | 3 | 1 | 2 |  |
| 1.1 | Um abschätzen zu können, wie viel Aufwand für die Implementation der einzelnen funktionalen Anforderungen an das Spiel entsteht, muss im Projekt genügend Zeit eingerechnet werden, um die Entwicklungsumgebung Unity3D kennen zu lernen. Dies beinhaltet auch das Bearbeiten von Tutorials. Die Thematik ist für die Gruppe vollkommenes Neuland und der Projektauftraggeber stellt dem Team deshalb genügend zeitliche Ressourcen bereit, sich mit den Werkzeugen vertraut zu machen. | 3 | 1 | 2 |  |
| 1.2 | Eine weitere Grundlage für den Erfolg des Projektes bildet das Verständnis der Automatentheorie. Speziell der Einsatz von Automaten innerhalb von Computerspielen. Die Bearbeitung dieser Unterlagen wurde vom Auftraggeber als Aufgabe gegeben. Das Einlesen in die Fachliteratur und das Umsetzung der Theorie stellt ebenfalls eines der Hauptziele des Projektes dar und ist deshalb als eine Anfoderung an das Projekt zu verstehen. |  |  |  |  |
|  |  |  |  |  |  |
| 2 | Konzeptionelles |
| 2.1 | In Form einer User Story ist die Spielidee und das Konzept des Spiels zu dokumentieren. Die Startsequenz, der Spielablauf und das Spielende, d.h. wie der Spieler das Basislevel abschliessen kann wird darin erläutert. Eine lineare Beschreibung zum Ablauf des Spiels als Veranschaulichung wird hilfreich sein.  Das Konzept soll so aufgebaut sein, dass der Spieler die an Ihn gestellte Anforderung bereits beim ersten Spielstart versteht. |  |  |  |  |
|  |  |  |  |  |  |
| 3 | Inhaltliches |
| 3.1 | Das Spielkonzept wird dem Spieler zu Beginn anhand eines kurzen Briefings erläutert, so dass der Spieler nach dem Durchlesen dieser Anleitung grundsätzlich befähigt ist, das erste Level erfolgreich zu absolvieren. |  |  |  |  |
|  |  |  |  |  |  |

## Technische Anforderungen

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Umsetzung der hierarchischen Automation. Das Verhalten der Gegner soll trotz dem hierarchischen Aufbau der Automaten vom Spieler nicht voraussehbar sein. Realisiert werden die Automaten gemäss bekannten Algorithmen. Eine eigene Entwicklung „from scratch“ ist nicht das Ziel. Für einzelne Zustände kommen State Patterns zum Einsatz. | geplant | 3 |  |  |  | JE | 10.10.2014 |
| Das Verhalten der Gegner soll mit einem Hierarchical State Maschine mit verschiedenen State Patterns umgesetzt werden. Das Gelernte wird später wiederverwendet und bildet eine solide Grundlage um ein bestehendes Verhaltenskonzept der Gegner zu erweitern. |  |  |  |  |  |  |  |
| Die Nutzung des CharacterMotor Scripts resp. die Erweiterung des FPSInputController Scripts ist für diese Zwecke sinnvoll. |  |  |  |  |  |  |  |

### Detailbeschreibung der technischen Anforderungen

# Chancen und Risiken des Projekts

Die Entwicklungsumgebung Unity3D, die an JavaScript angelehnte Unityscript Sprache, so wie die Entwicklung geeigneter Automaten sind für die Gruppe bisher unbekannte Werkzeuge der Spieleentwicklung. Anfänglich unbekannte Themengebiete stellen von Natur aus eine grosse Herausforderung an ein realistisches Zeitmanagement.

Nachdem der Projekt Scope geklärt wurde und in der Design-Thinking-Phase erste Protoypen entwickelt wurden, konnte die Einschätzung der zeitlichen Ressourcen neu bewertet werden. Eine erste Spieleidee, verschiedene Use Cases und die Modellierung des Levels konnten das Risiko einer Fehlkalkulation minimieren. Trotzdem sind noch immer Unsicherheiten bezüglich der Machbarkeit - insbesondere was das Umsetzen der Automatentheorie betrifft - vorhanden.

Ein regelmässiger Austausch mit den Stakeholder garantiert ein Verständnis für die Prioritäten im Projekt und mögliche Risiken können mit diesen besprochen und neu bewertet werden. Da in diesem Projekt Projektleitung und Auftraggeber durch den Dozenten verkörpert werden, wirkt sich ein Ausfall von Herr Eckerle wegen Krankheit oder andere Ereignissen stark auf unser Projekt aus.

Im Requirement Engineering besteht zudem das Risiko, nicht alle Wünsche und Anforderungen des Stakeholders erfüllen zu können. Deshalb wird diesem Teil in der ersten Phase des Projekts genug Ressourcen zugeteilt. Dennoch bietet sich auf Grund unserer gewählten agilen Entwicklungsmethode die Möglichkeit dynamisch auf Änderungswünsche und Problem-stellungen einzugehen resp. diese zu lösen.

# Glossar

Verantwortlichkeit: F. Schwab

| Wort / Abkürzung | Bedeutung |
| --- | --- |
| Spieler | Die Person, welche die Spielfigur im Spiel steuert |
| Spielfigur | Die virtuelle Figur, die vom Spieler innerhalb des Systems gesteuert wird. |

## Synonyme

| Wort / Abkürzung | Bedeutung |
| --- | --- |
| Rocket, Game, Spiel, Adventure | Als Synonyme verwendet |
| Gruppe, Team, | Als Synonyme verwendet |

# Referenzen

| Bezeichnung | Titel, Quelle |
| --- | --- |
| 1 | Artificial Intelligence for Games,  Second Edition by Ian Millington&John Funge, |

# Verschiedenes

Inhalt von Pohl, Kapitel 6, Seiten 69 – 100

Anforderungen modellbasiert dokumentieren. Das Modell ist eine Abstraktion der Realität.  
So bsp. werden dann die UML Diagramme verwendet (UML=Modellierungssprache).

Aus diesen Kapiteln nur das rausnehmen, was wichtig für die Aufgabenstellung ist.

* Ziele beschreiben – die Ziele des Stakeholders:  
  Dies sind vom Stakeholder charakteristische Merkmale des zu entwickelnden Systems.  
  *Wenn Zeit, noch ein Zielmodell machen.*
* S. 75 Use Cases und Szenarien dokumentieren, wie das System genutzt wird  
  (Bei uns eher die Bedienung des Games)
  + Ein Diagramm machen
  + Use Cases in Tabelle
  + Einzelne Use Cases ausführlich mit Alternativen etc.  
    (beide Punkte wie schon aus SED bekannt)
* S. 76 Systemanforderungen = Functional Requirements
* S. 81 Perspektiven auf die Anforderungen:  
  1. Struktur 2. Funktion 3. Verhalten (ebenfalls bekannt aus SED)
  + Strukturperspektive:   
    System im Systemkontext, Struktur von Input und Output
  + Funktionsperspektive:  
    Welche Informationen aus dem Systemkontext durch das System manipuliert werden und welche Systemteile den Kontext wie beeinflussen.
  + Verhaltensperspektive:  
    Zustandsorientierte Dokumentation, wie sich das System in den Kontext zu einem gegebenen Zeitpunkt einbettet. Reaktion auf Ereignisse, Effekte etc.
* Ab S. 82 die Modellierung dieser Perspektiven erklärt