

Projekt: JBomberman  
  
Projektplan

Pascal Kistler

Silvan Adrian

Fabian Binna

1. Änderungsgeschichte

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Version | Änderung | Autor |
| 23.02.15 | 1.00 | Erstellung des Dokuments | Gruppe |
| 24.02.15 | 1.01 | Neue Texte eingefügt | Fabian Binna |
| 25.02.15 | 1.02 | Logo + Neue Texte | Silvan Adrian |
| 27.02.15 | 1.03 | Arbeitspakete | Fabian Binna |
| 27.02.15 | 1.04 | Meilensteine + Phasen Iterationen | Silvan Adrian |
| 27.02.15 | 1.05 | Formatierungen + Design Anpassungen | Silvan Adrian |

1. Inhalt

1. Änderungsgeschichte 2

2. Inhalt 3

1. Einführung 4

1.1 Zweck 4

1.2 Gültigkeitsbereich 4

1.3 Referenzen 4

2. Projekt Übersicht 5

2.1 Zweck und Ziel 5

2.2 Primäre Features 5

2.3 Erweiterte Features 5

2.4 Lieferumfang 5

2.5 Annahmen und Einschränkungen 5

3. Projektorganisation 6

3.1 Organisationsstruktur 6

3.2 Externe Schnittstellen 6

4. Management Abläufe 7

4.1 Kostenvoranschlag 7

4.2 Zeitliche Planung 7

4.2.1 Phasen / Iterationen 8

4.2.2 Reviews 9

4.3 Besprechungen 9

5. Risikomanagement 10

5.1 Risiken 10

5.2 Umgang mit Risiken 10

6. Arbeitspakete 11

7. Infrastruktur 12

7.1 Entwicklungsinfrastruktur 12

7.2 Tools/Software 12

7.3 Kommunikationsmittel 12

8. Qualitätsmassnahmen 13

8.1 Dokumentation 13

8.2 Projektmanagement 13

8.3 Entwicklung 13

8.3.1 Vorgehen 13

8.3.2 Unit Testing 13

8.3.3 Code Reviews 13

8.3.4 Code Style Guidelines 13

8.4 Testen 14

8.4.1 Komponententest 14

8.4.2 Integrationstest 14

8.4.3 Systemtest 14

1. Einführung
   1. Zweck

Dieses Dokument beschreibt die Planung des SE2 Projekts.

* 1. Gültigkeitsbereich

Dieses Dokument ist während des ganzen Projekts gültig und wird laufend aktualisiert.

* 1. Referenzen

Spielprinzip/Beschreibung: <http://de.wikipedia.org/wiki/Bomberman>

Konzepte für Games: Spiele entwickeln für iPad, iPhone und iPod touch, Thomas Lucka, 2012, ISBN 978-3-446-43085-3

1. Projekt Übersicht

JBomberman ist ein Geschicklichkeitsspiel, das die gleichen Regeln wie das klassische Bomberman umfasst. Es gibt nur einen Multiplayer Modus, in dem bis zu vier Spieler gegeneinander antreten können. Der Multiplayer Modus ist über ein Netzwerk spielbar.

* 1. Zweck und Ziel

Wir wollen gelerntes aus Prog1, Prog2, Uint1 und SE1 anwenden und erweiterte Programmierkonzepte erlernen. Hinzu kommen Module wie VSS und ParProg, die wir parallel besuchen und auch für die Durchführung des Projekts notwendig sind.

Unser Ziel ist es ein Software Projekt in einem Team erfolgreich durchzuführen.

* 1. Primäre Features
* Spiellogik
* Netzwerkspiel
* Dedizierter Server
* Desktop Client
  1. Erweiterte Features
* Verschiedene Powerups
  1. Lieferumfang
* Source-Code
* Dokumentation (Projekt und Software)
* Ausführbare Applikation
  1. Annahmen und Einschränkungen

Die Applikation wird lediglich für Desktops zur Verfügung gestellt, dabei sollen folgende Betriebssysteme (Mac OSX , Linux, Windows) unterstützt werden um eine möglichst grosse Userbasis anzusprechenDas Projekt soll nach Abschluss Open Source für Interessenten zur Verfügung gestellt werden.

1. Projektorganisation

Jedes Teammitglied konzentriert sich auf seine eigenen Themengebiete, jedoch werden die Entscheidungen im Team besprochen und das Knowhow auf alle verteilt. Entscheidungen die das gesamte Projekt beeinflussen werden im Team besprochen und müssen von jedem akzeptiert werden.

* 1. Organisationsstruktur

**Daniel Keller**

* Projektbetreuer

**Silvan Adrian**

* Serveradministrator
* Build-Management
* Entwicklung

**Pascal Kistler**

* Qualitätsmanagement
* Entwicklung

**Fabian Binna**

* Projektleiter
* Entwicklung
  1. Externe Schnittstellen

Projektbetreuer: Daniel Keller ([d1keller@hsr.ch](mailto:d1keller@hsr.ch))

1. Management Abläufe
   1. Kostenvoranschlag

Wir rechnen mit 420 Stunden Gesamtaufwand. 60 Stunden davon sind Reserve aufgrund von potenziellen Risiken.

Gesamtaufwand: 420h

Davon Risikoaufwand: 60h

Aufwand pro Woche: 28h

Aufwand pro Woche pro Teammitglied: ~10h

* + 1. Phasen / Iterationen

|  |  |  |
| --- | --- | --- |
| Iteration | Beschreibung | Datum |
| Inception 1 | Projektantrag einreichen | 20.02.15 |
| Elaboration 1 | Anforderungen definiert, Technologien ausgewählt und Projektplan fertig gestellt.  Erster Prototyp zum Testen. | 05.03.15 |
| Elaboration 2 | Use Cases, Funktionale und nicht funktionale Anforderungen, Diagramme, Architekturentscheidungen | SW05 |
| Elaboration 3 | Architektur Prototyp  Fully dressed Use Cases (80%) | SW06 |
| Construction 1 | Client und Server Kommunikation umgesetzt.  Basic Character movements. | SW09 |
| Construction 2 | Hauptfeatures implementieren (Kollisionsdetektion, Bomben funktionieren, Powerups funktionieren) | SW11 |
| Construction 3 | Spielablauf und Matchmaking fertig.  Falls genügend Zeit werden noch erweiterte Features eingebaut. | SW13 |
| Transition | Vorbereitung zur Schlusspräsentation und Abgabe. | SW15 |

* + 1. Reviews

|  |  |  |
| --- | --- | --- |
| Datum | Reviews | Beschreibung |
| 05.03.15 | MS1-RV Projektplan | Review Projektplan mit Zeitplan und aktuellen Iterationsplänen |
| SW05 | MS2-RV Anforderungen und Analyse | Review der Anforderungsspezifikation und der Domainanalyse |
| SW07 | MS3-RV Ende Elaboration | Zwischenpräsentation mit Demo eines Architekturprototypen, Review |
| SW09 | MS4-RV Architektur/Design | Review von Architektur/Design und Architekturdokumenten |
| 28.05.15 | MS5-RV Schlusspräsentation/- abgabe | Präsentation und Demo der Software |

* 1. Besprechungen

Es werden regelmässige Besprechungen am Donnerstag morgen eingeplant um den Stand der jeweiligen arbeiten zu erfahren und gegebenenfalls Hilfestellung anzubieten ,falls etwas nicht funktionieren sollte.

1. Risikomanagement
   1. Risiken

Risiken werden im Dokument TechnischeRisiken-JBomberman.xslx beschrieben

* 1. Umgang mit Risiken

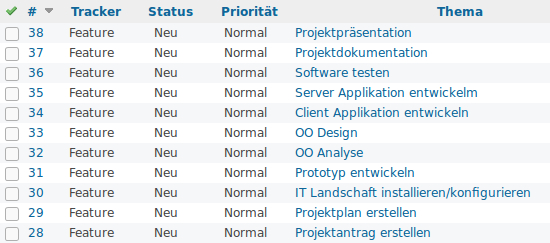
Für die Risiken werden Reserven eingeplant. Die Reserven werden direkt in die einzelnen Tickets eingerechnet. Falls Risiken eintreffen werden diese sofort kommuniziert. Jede Woche werden eingetroffene Risiken und potenzielle Risikogefahren diskutiert. Eingetroffene Risiken werden im Team besprochen und mögliche Lösungen evaluiert.

1. Arbeitspakete

Die Arbeitspakete werden in Redmine erstellt und gepflegt (siehe Screenshot).

Lesender Zugriff ist anonym möglich, schreibender nur eingeloggt (Projekt ist öffentlich).

Link zum Redmine: <http://se2p.zonk.io/redmine>



1. Infrastruktur
   1. Entwicklungsinfrastruktur

Aus persönlicher Präferenz ist die Infrastruktur breit gesät. Beim IDE konnten wir uns auf einen gemeinsamen Nenner bringen.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Hardware | Betriebssystem | IDE |
| Pascal Kistler | ASUS | Windows 8.1 | Eclipse Luna |
| Silvan Adrian | Mac Book Pro | OSX 10.9.5 | Eclipse Luna |
| Fabian Binna | Lenovo T530s | Ubuntu 14.04.1 | Eclipse Luna |

Neben den persönlichen Notebooks steht uns der beantragte SE2BuildServer als Entwicklungsinfrastruktur zur Verfügung

* 1. Tools/Software
* BuildServer: Jenkins
* CodeReview: Gerrit
* Konfigurtaionsmanagment: Git
* Projektmanagment: Redmine
  1. Kommunikationsmittel

Ausserhalb des Reviewtermins wird mit folgenden Mitteln kommuniziert:

* E-Mail
* Skype
* Redmine (Wiki Kommentare)
* Gerrit (Code Kommentare)
* Whatsapp

1. Qualitätsmassnahmen
   1. Dokumentation

Sämtliche Dokumente werden in Git gespeichert. So sind diese immer für alle verfügbar und können einfach versioniert und Änderungen nachvollzogen werden.

* 1. Projektmanagement

Als Projektmanagementtool wird Redmine verwendet. Hier werden alle ausstehenden, laufenden und erledigen Tickets verwaltet.

* 1. Entwicklung

Der Source Code befindet sich ebenfalls in einem Git Repository und wird durch regelmässige Code Reviews überprüft.

* + 1. Vorgehen

Jedem Commit wird eine Beschreibung hinzugefügt, danach können die Commits im Redmine dem passenden Ticket zugewiesen werden.

* + 1. Unit Testing

Es sollten für jede Klasse / jedes Codemodul Junit Tests geschrieben und durchgeführt werden. Vor jedem push müssen alle Test erfolgreich durchlaufen werden.

* + 1. Code Reviews

Durch Code Reviews wird sichergestellt, dass der Code den Style Guidelines entspricht.

Durch die Kontrolle durch ein oder zwei weitere Teammitglieder wird sichergestellt, dass die Codequalität stets auf hohem Niveau ist.

* + 1. Code Style Guidelines

Es werden die Code Style Guidelines aus den Modulen Prog1 und SE1 verwendet. Namen im Code (wie Variablen, Methoden, etc.) werden in Englisch benannt.

* 1. Testen
     1. Komponententest

Zu jeder Komponente werden mehrere Test geschrieben. Diese stellen sicher, dass alle Komponenten ordnungsgemäss funktionieren und kein nicht funktionierender Code gepusht wird.

* + 1. Integrationstest

Für jede implementierte Funktion werden wiederum Tests geschrieben, um die korrekte Funktionsweise und die Zusammenarbeit der einzelnen Module sicherzustellen.

* + 1. Systemtest

Systemtest werden gegen Ende des Projekts erstellt. Sie garantieren, dass das gesamte System ordnungsgemäss funktioniert und alle Komponenten richtig zusammen arbeiten.