Voranalysebericht

Status	In Arbeit / In Prüfung / **Abgeschlossen**
Projektname	<CGR>
Projektleiter	<Ilija Tovilo>
Auftraggeber	< Martin Frieden (Gibb)>>
Autoren	<Ilija Tovilo> <Lukas Seglias>
Verteiler	<-->

Änderungskontrolle, Prüfung, Genehmigung

Version	Datum	Beschreibung, Bemerkung	Name oder Rolle
0.1	12.02.2013	Start der Erstellung des Voranalyseberichts	Projektleiter
0.5	19.02.2013	-	Projektleiter
0.5.1	23.02.2013	UML-Entitätsdiagramm hinzufügen	Lukas Seglias
1.0	26.02.2013	Beenden des Voranalyseberichts	Projektleiter

Referenzen

Referenz	Titel, Quelle
[1]	Buch „Hermes Grundwissen“

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Zusammenfassung

Durch den Voranalysebericht haben wir die Schwachstellen und Stärken der bestehenden Spiele untersucht. Ausserdem konnten wir die Systemziele des Produkts festlegen und die fachlichen Entitäten erarbeiten. Aufgrund dessen konnten wir verschiedene Lösungsvarianten untersuchen und die Beste herauszufiltern.

2 Ist-Aufnahme und Ist-Analyse

2.1 Ist-Zustand

Es gibt kein vorhandenes Spiel, jedoch ist die Entwicklungs-Umgebung, sowie das nötige Wissen zum Aufbau des Spiels vorhanden.

Es gibt zahlreiche Foren im Internet, wo Informationen über Objective-C, Cocoa oder Cocos2d zu finden sind. Diese sind permanent verfügbar, solange keine Störungen vorliegen.

Es sind zwei Macintosh Computer vorhanden, auf welchen entwickelt werden kann. Es müssen keine weiteren Investitionen gemacht werden.

Die beiden Computer sind permanent verfügbar. Das Versionen-Verwaltungs-System (BitBucket) ist ebenfalls permanent verfügbar, solange keine Störungen vorliegen.

2.2 Schwachstellenanalyse

ID	Beschreibung
W1	Es gibt kein qualitativ-hochwertiges Super Mario Remake für das iPhone.
W2	Die Plattformer-Spiele auf den Smartphones bieten oft eine schlechte Steuerung an.
W3	Die Plattformer-Spiele bieten kein solch nostalgisches Spielerlebnis an.
W4	Es sind oft nur wenige Levels vorhanden

2.3 Stärkenanalyse

ID	Beschreibung
S1	Besitzen oft aufwendige Grafiken
S2	Simple und unterhaltende Spielmechanik

2.4 Sicherheit

Das Spiel enthält keine heiklen Daten, welche eine spezielle Schutzmassnahme erfordern.

Es wird nicht mit User-Daten gearbeitet, wodurch es keinen Anlass zum Schutz dieser gibt.

3 Systemziele

ID	Beschreibung	Priorität	Erreicht wenn …
1	Es wird ein Spiel erstellt, welches lauffähig auf einem Smartphone ist.	Muss	Die Benutzer können das Spiel starten und geniessen.
2	Das Spiel erlaubt die Benutzer-Interaktion und handelt basierend auf diesen.	Muss	Der Benutzer kann den Spieler benutzerfreundlich steuern.
3	Das Spiel enthält mehrere Levels.	Muss	Nach Abschluss eines Levels wird das nächste freigeschaltet. Der Benutzer kann nach der Freischaltung dieser frei auswählen.
4	Das Spiel enthält verschiedene, ansteigende Herausforderungen	Muss	Die Gegner sind mit der Zeit stärker, schwerer Bezwingbar.
5	Es gibt verschiedene Gegenstände, welche das Spiel erleichtern.	Muss	Der Spieler kann Gegenstände wie Pilze, Blumen usw. sammeln und benutzen.
6	Das Spiel speichert den Score jedes Levels.	Kann	Der Spieler kann im Menu den Highscore anzeigen.
7	Das Spiel vermittelt das Gefühl des alten Super Mario Bros.	Muss	Der Spieler fühlt sich in alte Zeiten versetzt.

4 Fachlicher Soll-Zustand

4.1 Fachliche Entitätstypen

- Menu
  - Auswahl eines Levels
  - Einstellungen vornehmen
    - Ton ein/ausschalten
    - Musik ein/ausschalten
- Level
  - Start-Punkt
  - End-Punkt
  - Spiel-Elemente
  - Spieler
  - Gegner
  - Zeitlimit
    - Das Spiel ist nach Ablauf der Zeit zu ende
- Mario
  - Anzahl Leben
  - Lebens-Status
  - Aktive Items
    - Dies sind Items, welche Mario eingenommen hat und momentan noch aktiv sind.
- Item
  - Stern
    - Wirkungsdauer
    - Unverwundbarkeit für kurze Zeit
  - Pilz
    - Verwandelt den kleinen Mario zum grossen Mario
  - Feuer-Blume
    - Mario wird gross und kann Feuerbälle spucken
  - Münzen
    - Mario erhält mehr Punkte durch das Sammeln von Münzen
    - Pro 100 Münzen erhält Mario ein weiteres Leben
- Röhre
  - Eingangspunkt
  - Ausgangspunkt
  - Transportiert Mario zu einem anderen Ort
- Block
  - Beinhaltet 0, 1 oder mehrere Münzen

- Gegner
  - Koppa
    - Läuft vom einem Abgrund/Hindernis zum nächsten, und zurück
    - Beim Sprung auf den Kopf ziehen sich die Koppas in den Panzer zurück
      - Der Spieler kann den Panzer anstossen, worauf dieser in eine Richtung geschleudert wird. Dieser ist dann schädlich für alle Spielelemente.
  - Gumba
    - Läuft vom einem Abgrund/Hindernis zum nächsten, und zurück
    - Stirbt beim Sprung auf den Kopf
  - Lakitu
    - Wirft Spiny Gegner herunter
    - Fliegt auf einer Wolke
    - Kann durch einen Sprung auf den Kopf getötet werden.
  - Spiny
    - Besitzt Stacheln, welche schädlich für Mario sind
    - Können nur durch einen Koppa-Panzer getötet werden

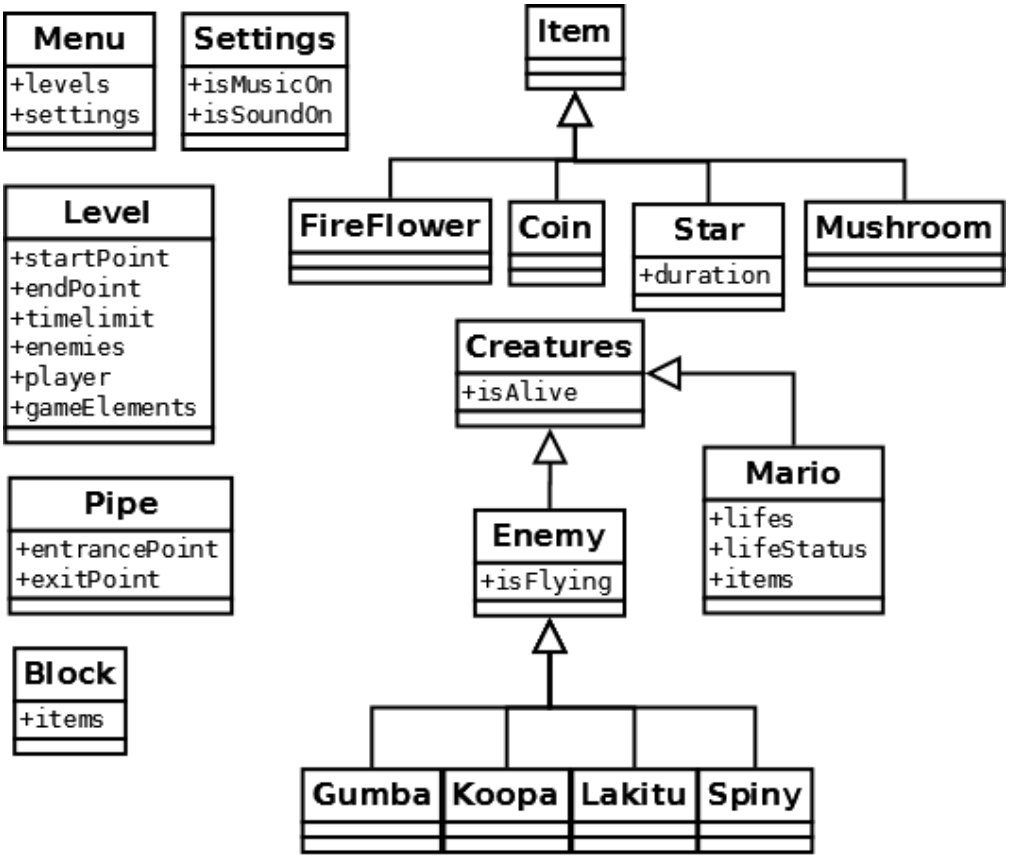

Abbildung 1: UML-Diagramm der Enittätstypen

4.2 Hauptaufgaben der neuen Lösung

- OpenGL als Schnittstelle zum Grafik-Chip, da die grafische Darstellung von iOS auf OpenGL basiert

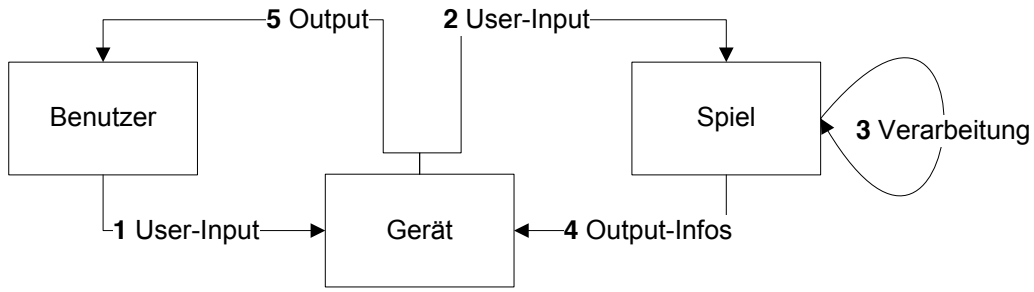

Abbildung 2: Ablauf der Interaktion des Benutzers und des Spiels

4.2.1 Ablauf eines Spiel-Durchgangs

1. Benutzer startet das Spiel.
2. Der Benutzer wählt im Menu einen freigeschalteten Level seiner Wahl.
3. Im Level erscheinen die Spielelemente wie Gegner, Blöcke, Items auf dem Bildschirm.
4. Der Benutzer interagiert mit den Spielelementen.
5. Der Benutzer absolviert den Level.
6. Ein neuer Level wird freigeschaltet.
7. Der Benutzer kann den Vorgang wiederholen, oder das Spiel beenden.

4.3 Anforderungen bzgl. Informationssicherheit und Datenschutz

Es werden keine weiteren Massnahmen für das sicherstellen der Informationen benötigt, da nicht mit den persönlichen Informationen des Benutzers gearbeitet wird.

5 Lösungsvarianten

5.1 Lösungsvariante Cocos2d

5.1.1 Beschreibung der Lösungsvariante

Cocos2d ist eine Game-Engine für Objective-C. Diese vereinfacht die Entwicklung eines Spieles enorm.
Cocos2d basiert auf Objective-C, und nimmt die direkte Arbeit mit OpenGL ab.
Die Engine hat eine riesige aktive Community und ist Open Source.

5.1.2 Struktur (grobe Architektur)

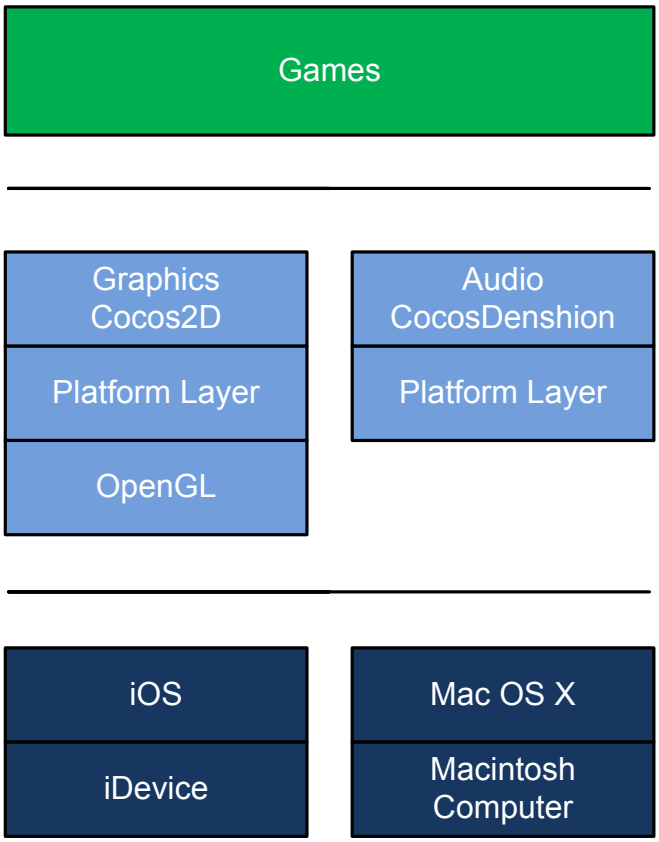

Abbildung 3: Struktur von Cocos2d

5.1.2.1 Schnittstelle

Cocos2d ist eine Schnittstelle von Objective-C zu OpenGL.
OpenGL ist die Grafik-Schnittstelle zum Grafik-Chip des iPhones.

5.1.3 Abdeckung der Anforderungen

Mit dieser Spiel-Engine ist das Modellieren einer solchen Umgebung möglich, um die erforderte Spielmechanik zu erstellen.

5.1.4 Realisierbarkeitsbetrachtung

Cocos2d ist sehr gut realisierbar, da es eine sehr mächtige, und einfach zu benutzende Game-Engine ist.
Ausserdem hat diese eine aktive Community.

5.2 Lösungsvariante Cocos2d-5

5.2.1 Beschreibung der Lösungsvariante

Cocos2d-5 ist eine auf HTML5 und Javascript basierte Game-Engine.
Die Engine ist relativ neu, da HTML5 noch im Entwicklung-Stadium ist.

5.2.2 Struktur (grobe Architektur)

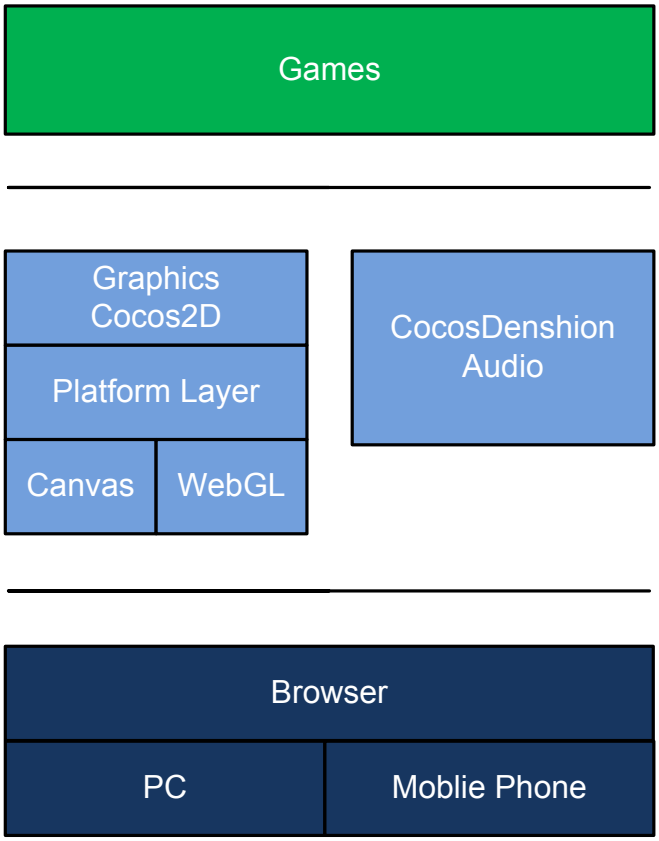

Abbildung 4: Struktur von Cocos2d-5

5.2.2.1 Schnittstelle

Cocos2d-5 ist ein Web-Framework, welches die Arbeit mit HTML5 und JavaScript vereinfacht.
Mithilfe von Canvas und Javascript werden Grafiken und Formen gezeichnet.

5.2.3 Abdeckung der Anforderungen

Mit dieser Spiel-Engine ist das Modellieren einer solchen Umgebung möglich, um die erforderte Spielmechanik zu erstellen.

5.2.4 Realisierbarkeitsbetrachtung

Cocos2d-5 hat eine weniger gute Realisierbarkeit, da diese noch in Entwicklung ist.
Auch die Community ist deshalb relativ klein.

5.3 Lösungsvariante OpenGL

5.3.1 Beschreibung der Lösungsvariante

OpenGL ist die Graphics-Library, welche die Basis der grafischen Darstellung von iOS ist.
Mit dieser Lösungsvariante ist die Entwicklung relativ komplex, da nur sehr abstrakte Funktionen zur Darstellung von Objekten verfügbar sind. Dies ermöglicht zwar eine sehr hohe Erweiterbarkeit, erschwert die Entwicklung jedoch enorm.

5.3.2 Struktur (grobe Architektur)

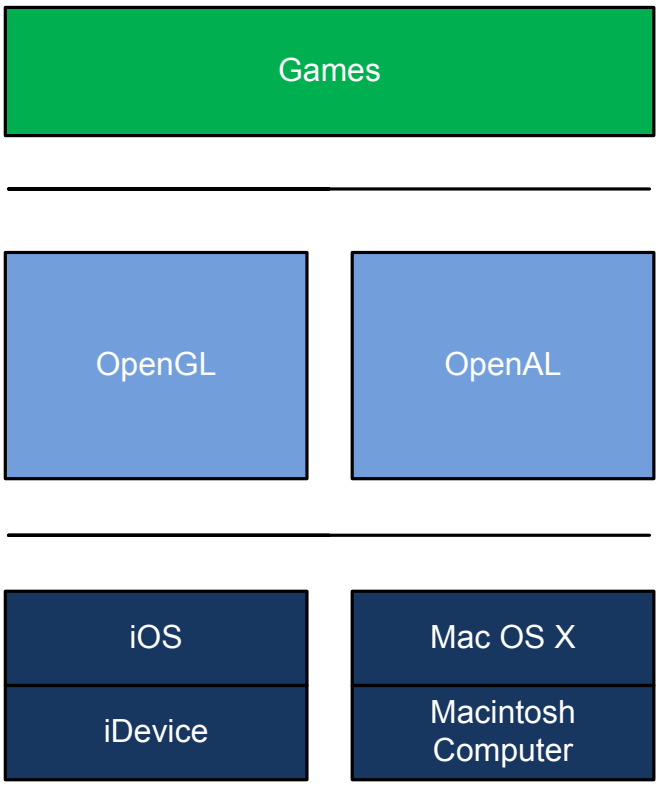

Abbildung 5: Sehr vereinfachte Struktur von OpenGL

5.3.2.1 Schnittstelle

OpenGL ist die Grafik-Schnittstelle zum Grafik-Chip des iPhones.

5.3.3 Abdeckung der Anforderungen

Mit dieser Lösungsvariante ist die Implementation der Anforderungen zwar möglich, jedoch deutlich anspruchsvoller als mit einer anderen Lösungsvariante.

5.3.4 Realisierbarkeitsbetrachtung

OpenGL ist relativ komplex, weshalb die Realisierung sich als schwierig herausstellen könnte.

5.4 Lösungsvariante Sparrow

5.4.1 Beschreibung der Lösungsvariante

Sparrow ist eine Game-Engine, welche die Arbeit mit OpenGL abnimmt.
Sie ist Cocos2d sehr ähnlich und hat die gleichen Ziel-Plattformen.

5.4.2 Struktur (grobe Architektur)

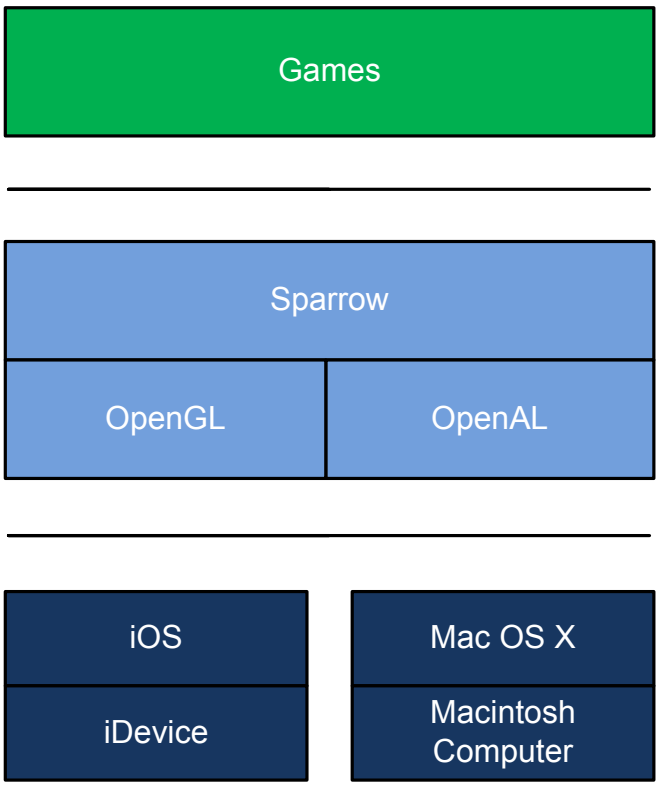

Abbildung 6: Struktur von Sparrow

5.4.2.1 Schnittstelle

Sparrow ist eine Schnittstelle zwischen OpenGL und Objective-C.
OpenGL ist die Grafik-Schnittstelle zum Grafik-Chip des iPhones.

5.4.3 Abdeckung der Anforderungen

Mit dieser Spiel-Engine ist das Modellieren einer solchen Umgebung möglich, um die erforderte Spielmechanik zu erstellen.

5.4.4 Realisierbarkeitsbetrachtung

Sparrow ist sehr performant, und hat eine gute Dokumentation.
Jedoch kennen wir diese Plattform kaum.

6 Entscheid über die Lösungsvarianten

Bewertungskriterien	Cocos2d	Cocos2d-5	OpenGL	Sparrow
Wirtschaftlichkeit	100%	100%	100%	100%
Realisierbarkeit	100%	60%	20%	100%
Bekanntheit	80%	20%	10%	20%
Performance	100%	80%	100%	100%
Resultat	95%	65%	57.5%	80%

Der eindeutige Gewinner ist Cocos2d.

7 Mittelbedarf

Es hat sich nichts geändert.

8 Wirtschaftlichkeit

Das Projekt erhält keinen sofortigen Ertrag. Dieser wird erst mit der Zeit erwirtschaftet.
Deshalb ist es sehr schwer, diese Frage zu beantworten.
Es gibt keine Kosten in dem Projekt, weshalb hierbei keine Risiken entstehen.
Jedoch kann sich das Projekt definitiv auszahlen.

9 Konsequenzen

- Ausfall des Versionen-Systems **BitBucket**
- Ausfall der benötigten Computer
- Ausfall des benötigten Test-Gerätes

10 Antrag

Wir beantragen die Genehmigung des vorliegenden Voranalyseberichts und die Freigabe für die Konzeptphase durch den Auftraggeber.

Ort: _____ Datum: _____ Unterschrift: _____