

Bearbeitungsbeginn: 01.09.2014

Vorgelegt am: TBA

# Thesis

zur Erlangung des Grades

**Master of Science**

im Studiengang Medieninformatik

an der Fakultät Digitale Medien

***Dominik Steffen***

***Matrikelnummer: 245857***

**TODO: Code Content Verschränkung im Bezug auf 3D Game Engines.**

*Erstbetreuer:* Prof. Christoph Müller

*Zweitbetreuer:* Prof. Dr. Wolfgang Taube



# Abstract

---

Arbeitsprozesse in heutigen Game Engines verlangen von Entwicklern meist das Erlernen neuer Toolsets und das während eines meist zeitlich sehr beschränkten Projektes. Es wäre für Entwickler einfacher sich mit den bereits bekannten Tools zu beschäftigen und mit diesen großartige Ergebnisse zu erreichen. Designer müssen sich oft in unbekannte Editoren und SDKs einarbeiten während Entwickler sich in Grafische Editoren einarbeiten um ihren Code an der richtigen Stelle des Projekts zu platzieren. Diese Arbeit baut eine Brücke zwischen beiden Welten. Durch die Konzeption und Umsetzung eines Software Tools wird eine Trennung der Abhängigkeiten in einem Projekt erreicht. Mit Hilfe eines Plugins ist es möglich, dass Designer oder Entwickler jederzeit mit ihren eigenen Tools in die Entwicklung eines Projektes einsteigen. Es wird ermöglicht mit Cinema 4D und einer IDE wie VS2013 an einem Projekt mit der FUSEE Engine zu arbeiten ohne die bereits bekannte Welt zu verlassen. Ein Engine Projektstruktur managed sich durch die Nutzung des entstandenen Plugins selbst. Das zuerst Konzeptionell entworfene Tool wurde während dieser Arbeit umgesetzt und bietet ausreichende Basisfunktionalität um ein Projekt zu erstellen. Hierzu wurden verschiedene Konzepte betrachtet und andere Game Engines auf Workflow und Anwendbarkeit untersucht. Es wurden einige Kernkonzepte erkannt und für eine Implementierung in FUSEE analysiert und weiter entwickelt.



# Eidesstattliche Erklärung

---

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Masterthesis selbstständig und ohne unzulässige fremde Hilfe angefertigt habe. Alle verwendeten Quellen und Hilfsmittel die sowohl zum schreiben dieser Arbeit als auch zum Entwickeln des dazugehörigen Sourcecodes benutzt wurden, habe ich angegeben.

---

Dominik Steffen, Küssaberg den 26. Januar 2015



"Hier steht ein wichtiges Zitat zur Entstehung dieser Arbeit."

- TBD.

Kontakt:

Dominik Steffen (Matr.-Nr.: 245857)

Hochschule Furtwangen

E-Mail:

dominik.steffen@hs-furtwangen.de, dominik.steffen@gmail.com





# Inhaltsverzeichnis

---

<b>1</b>	<b>Anforderungen, Ziele und eine Fragestellung</b>	<b>1</b>
1.1	Entwicklungsprozesse in Interaktiver 3D Software und Games . . . . .	1
1.1.1	Projektmanagement Modelle . . . . .	1
1.1.2	Internes Tool Developing anstatt Tool licencing . . . . .	1
1.2	Mitglieder eines Entwicklerteams . . . . .	1
1.2.1	Artists . . . . .	1
1.2.2	Designer . . . . .	1
1.2.3	Engineer . . . . .	1
1.2.4	Weitere für diese Arbeit nicht relevante . . . . .	1
1.3	Stakeholderanalyse intern . . . . .	1
1.4	Ein Arbeitsprozess wird entwickelt . . . . .	1
1.4.1	Game Authoring / Game Development . . . . .	1
1.4.2	Tool Development . . . . .	2
1.4.3	Planung . . . . .	2
1.4.4	Analyse . . . . .	2
1.4.5	Design . . . . .	2
1.4.6	Implementierung . . . . .	3
1.4.7	Warum eine Trennung von Code und Content? . . . . .	3
<b>2</b>	<b>Entwicklung eines Konzeptes</b>	<b>4</b>
2.1	Use Cases der verschiedenen Entwickler . . . . .	5
2.1.1	Was möchten Artists? . . . . .	5
2.1.2	Was möchten Designer? . . . . .	5
2.1.3	Was möchten Entwickler? . . . . .	5
2.1.4	Projekt bezogen . . . . .	5
2.1.5	Prozess bezogen . . . . .	5
2.2	Aktuelle Engines und deren Arbeitsprozesse . . . . .	5
2.2.1	Prozesse in Game Engines und einem Framework . . . . .	5
2.2.2	Unreal Engine 4 . . . . .	5
2.2.3	Unity 3D . . . . .	5

2.2.4	idTech X . . . . .	5
2.2.5	Weitere . . . . .	5
2.3	Konzeptentwurf . . . . .	5
2.3.1	Systemdesign für ein Plugin . . . . .	5
2.3.2	Systemdesign für einen Project-Handler . . . . .	5
2.3.3	Entfernen von Abhängigkeiten . . . . .	5
2.3.4	Zeitersparnis durch bekannte Tools . . . . .	5
2.3.5	Warum Fusee und Cinema 4D? . . . . .	5
2.4	Die Implementierung . . . . .	5
2.4.1	Cinema 4D Plugin API und SDK . . . . .	5
2.4.2	Fusee . . . . .	5
2.5	Das eigentliche Plugin . . . . .	6
2.5.1	Visualisierung der Systemarchitektur . . . . .	6
2.5.2	Generieren eines Fusee Projektes . . . . .	6
2.5.3	Code Generation und die Vermeidung von Roundtrips (nicht so ganz roundtrips, generierung um generierung etc.) . . . . .	6
2.5.4	XPresso Schaltungen - Programmieren ohne Program- mieren . . . . .	6
2.5.5	Partial Classes in .NET . . . . .	6
<b>3</b>	<b>Ergebnisse und Erkenntnisse</b>	<b>7</b>
3.1	Game Authoring Entwicklungsprozesse jetzt und in Zukunft . .	7
3.2	Wie weit ist die Implementierung fortgeschritten? . . . . .	7
3.3	Welcher Mehrwert wurde erreicht? . . . . .	7
3.4	Integration des Systems in den weiteren Projektverlauf von FU- SEE . . . . .	7
	<b>Verzeichnis der Sourcecode Beispiele</b>	<b>9</b>
	<b>Tabellenverzeichnis</b>	<b>10</b>
	<b>Abbildungsverzeichnis</b>	<b>11</b>
	<b>Literaturverzeichnis</b>	<b>12</b>
	<b>UML Diagramme</b>	<b>13</b>



# 1 Anforderungen, Ziele und eine Fragestellung

---

## 1.1 Entwicklungsprozesse in Interaktiver 3D Software und Games

### 1.1.1 Projektmanagement Modelle

Agile Modelle vs. klassische Modelle

Scrum

Wasserfall

### 1.1.2 Internes Tool Developing anstatt Tool licencing

## 1.2 Mitglieder eines Entwicklerteams

### 1.2.1 Artists

### 1.2.2 Designer

### 1.2.3 Engineer

### 1.2.4 Weitere für diese Arbeit nicht relevante

## 1.3 Stakeholderanalyse intern

Das bedeutet, diese Gruppen müssen bei der entwicklung eines Tools beachtet werden. Am besten wäre eine Rücksprache mit diesen Teammitgliedern und anschliessend eine analyse der Bedürfnisse für die jeweiligen Aufgabengebiete. Hierzu würde es sich empfehlen ein Gespräch zu führen und weiterhin einen praktischen Besuch am Arbeitsplatz zu vereinbaren.

## 1.4 Ein Arbeitsprozess wird entwickelt

### 1.4.1 Game Authoring / Game Development

Was ist das, was beschreibt es, wieso ist es hier relevant?

### 1.4.2 Tool Development

### 1.4.3 Planung

- Eine Software soll es ermöglichen, dass Artists, Designer und Developer an ein und dem selben Projekt arbeiten können ohne die gewohnte Arbeitsumgebung (3D-Modellierungssoftware, IDE) zu verlassen und etwas komplett neues (Level-Editor) zu erlernen.
- Die komplette Entwicklung und der Entwurf bedienen sich der FUSEE Engine.
- Ziel ist es in Cinema 4D ein FUSEE Projekt anzulegen, zu speichern und es zu öffnen (In Form einer Szene, Level, Welt.). Weiterhin müssen Assets ins Spiel integriert werden können die von Artists, Designern und Entwicklern bearbeitet werden können.
- Das Projekt sollte aus C4D heraus gebaut werden können.

### 1.4.4 Analyse

- Eine Stakeholderanalyse schafft Klarheit, welche Parteien mit dem zu erstellenden Tool arbeiten müssen.
- Die Game Engines sollten kategorisiert werden. Hierzu ist ein "Gitternetz" aufzustellen.
- Verschiedene Game Engines müssen darauf untersucht werden, wie die Arbeitsschritte in diesen zu erledigen sind. Jede Untersuchung sollte dokumentiert werden.
- Es ist zuerst zu analysieren, welche Schritte für welche Art der Arbeit notwendig sind. Hierzu werden Use Cases der verschiedenen Rollen und Aufgaben erstellt.

### 1.4.5 Design

- Aufgrund der erstellten Use Cases wird ein System Design erstellt.
- Das Systemdesign sollte Roundtrips vermeiden
- Hierzu können Partial Classes untersucht werden.
- Es soll möglichst wenig "geparsed" oder "konvertiert" werden

- Dateien sollen Version Control kompatibel bleiben (wenig bis keine Binaries)
- Das Projekt muss Strukturiert sein
- Eine Fusee Solution soll gehandelt werden können

#### 1.4.6 Implementierung

- Die Software wird auf basis der FUSEE Engine und Cinema4D R16 erstellt.

#### Asset Pipeline und Feedback

SEHR KRITISCHE MARKE! Das Asset aus dem Editor in die Engine bekommen, die Darstellung etc kontrollieren. Zeitkritisch beim export etc. Carter Seite 6 und folgende.

#### Die Struktur von Game Assets

Assets und das aufbrechen in Bestandteile eines Projektes. Level etc. bestehen aus Assets und mehr.

#### 1.4.7 Warum eine Trennung von Code und Content?



## 2 Entwicklung eines Konzeptes

---

### 2.1 Use Cases der verschiedenen Entwickler

#### 2.1.1 Was möchten Artists?

#### 2.1.2 Was möchten Designer?

#### 2.1.3 Was möchten Entwickler?

#### 2.1.4 Projekt bezogen

Projekt anlegen

Projekt öffnen

Projekt speichern

Projekt bauen

In Projekt einsteigen

Projekt clonen etc.

#### 2.1.5 Prozess bezogen

Gleichzeitig an Projekt arbeiten

Model Datei importieren

Gleichzeitig an einem Objekt arbeiten

### 2.2 Aktuelle Engines und deren Arbeitsprozesse

#### 2.2.1 Prozesse in Game Engines und einem Framework

#### 2.2.2 Unreal Engine 4

#### 2.2.3 Unity 3D

#### 2.2.4 idTech X

#### 2.2.5 Weitere

### 2.3 Konzeptentwurf

#### 2.3.1 Systemdesign für ein Plugin

#### 2.3.2 Systemdesign für einen Project-Handler



Irgend etwas dass im Zentrum steht. Alles andere muss auf Level etc aufgeteilt werden und bis zum einzelnen Asset heruntergebrochen werden.

## 2.5 Das eigentliche Plugin

### 2.5.1 Visualisierung der Systemarchitektur

Welche Programme und Systeme sind beteiligt?

### 2.5.2 Generieren eines Fusee Projektes

### 2.5.3 Code Generation und die Vermeidung von Roundtrips (nicht so ganz roundtrips, generierung um generierung etc.)

### 2.5.4 XPresso Schaltungen - Programmieren ohne Programmieren

### 2.5.5 Partial Classes in .NET

## 3 Ergebnisse und Erkenntnisse

---

3.1 Game Authoring Entwicklungsprozesse jetzt und in Zukunft

3.2 Wie weit ist die Implementierung fortgeschritten?

3.3 Welcher Mehrwert wurde erreicht?

3.4 Integration des Systems in den weiteren Projektverlauf von FUSEE

# Anhang

## Verzeichnis der Sourcecode Beispiele

---

## Tabellenverzeichnis

---

## Abbildungsverzeichnis

---

# Literaturverzeichnis

---

- [1] Michael Abrash. *Graphics Programming Black Book*. TBD, 1997.
- [2] Ben Carter. *The Game Asset Pipeline*. Delmar Cengage Learning, 2004.
- [3] Heather Chandler. *The Game Production Handbook*. Thomson Delmar Learning, 2006.
- [4] E Gamma, R Johnson, R Helm, and J Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Pearson Deutschland, 2014.
- [5] Jason Gergory. *Game Engine Architecture*. Taylor & Francis Ltd., 2009.
- [6] Eric Lengyel. *Mathematics for 3D Game Programming and Computer Graphics*. Course Technology, Cengage Learning, 2012.
- [7] Graham Wihlidal. *Game Engine Toolset Development*. Thomson Course Technologies, 2006.

## UML Diagramme

---