****

**FH - Studiengang für**

**Informationstechnik und System-Management**

**Salzburg**

**ITS**

**Software Architektur Dokumentation einer Verkehrssimulation**

**Version: 1**

**Datum: 25.05.2017**

**Autoren: Stephanie Kaschnitz , Christopher Wieland, Martin Wieser, Andreas Lippmann, Hannes Kleiner**

**Unterschrift des Autors / der Autorin:**

**Tabellenverzeichnis**

[Tabelle 1.1: Qualitätsmerkmale 5](#_Toc483947124)

[Tabelle 1.2:Stakeholder 6](#_Toc483947125)

[Tabelle 2.1: Technische Rahmenbedingungen 6](#_Toc483947126)

[Tabelle 2.2: Organisatorische Rahmenbedingungen 7](#_Toc483947127)

[Tabelle 4.1: Lösungsstrategie 9](#_Toc483947128)

**Abbildungsverzeichnis**

[Abbildung 3.1: Kontext 8](#_Toc483947106)

**Inhaltsverzeichnis**

[1 Einführung 5](#_Toc483947152)

[1.1 Aufgabenstellung 5](#_Toc483947153)

[1.2 Qualitätsziele 5](#_Toc483947154)

[1.3 Stakeholder 6](#_Toc483947155)

[2 Randbedingungen 6](#_Toc483947156)

[2.1 Technische Randbedingungen 6](#_Toc483947157)

[2.2 Organisatorische Rahmenbedingungen 7](#_Toc483947158)

[3 Kontext 7](#_Toc483947159)

[3.1 Fachlicher Kontext 8](#_Toc483947160)

[4 Lösungsstrategie 9](#_Toc483947161)

[4.1 Aufbau der Verkehrssimulation 9](#_Toc483947162)

[5 Bausteinschicht 10](#_Toc483947163)

[6 Laufzeitschicht 10](#_Toc483947164)

[7 Verteilungsschicht 10](#_Toc483947165)

[8 Konzepte 10](#_Toc483947166)

[9 Entscheidungen 10](#_Toc483947167)

[9.1 Graphische Darstellung 10](#_Toc483947168)

[9.2 Kommunikation zwischen den Gruppen 10](#_Toc483947169)

[10 Risiken 11](#_Toc483947170)

# **Einführung**

Dieser Abschnitt führt die Aufgabenstellung ein und skizziert die Ziele, welche die Verkehrssimulation verfolgt.

## **Aufgabenstellung**

Die Aufgabenstellung beinhaltet eine mikroskopische Simulation von Fahrzeuge, sowohl PKW als auch LKW, und Ampelanlagen. Ein weiterer wichtiger Punkt stellt das zusammenhängende Verkehrs-/Straßennetz dar. Weiters ist es möglich ungeregelte und geregelte Kreuzungen darzustellen. Ein wichtiger Faktor ist die Parametrisierbarkeit der Verkehrsteilnehmer und der Lichtsteueranlage.

Die Lichtsteueranlage wird über einen eigenen Prozess geregelt. Die verfügt die Applikation über eine grafische Darstellung sowie über eine einfache Benutzerschnittstelle.

In weiterer Folge sollen nun die Simulationen der verschiedenen Gruppen untereinander kommunizieren können. Das heißt, dass Fahrzeuge bei einer Straße einer Simulation ausfahren und dann in der anderen Simulation über eine Straße wieder einfahren können. Dem Benutzer wird ermöglicht beliebige Hindernisse auf der Straße per Mausklick ein-/auszuschalten.

## **Qualitätsziele**

Die folgende Tabelle beschreibt die zentralen Qualitätsziele.

|  |  |
| --- | --- |
| **Qualitätsmerkmal** | **Erläuterung** |
| Parametrisierbarkeit | Sowohl die Lichtsteueranlage als auch das Verhalten der Verkehrsteilnehmer sollen veränderbar sein. |
| Leicht bedienbar | Die Simulation und die dazugehörigen Einstellungen sind leicht einzustellen. |
| Effizienz | Die Berechnungen, welche im Hintergrund stattfinden, sollten rasch geschehen, ohne eine flüssige Simulation zu verhindern. |

Tabelle .: Qualitätsmerkmale

## **Stakeholder**

Die folgende Tabelle stellt die Stakeholder dar.

|  |  |
| --- | --- |
| **Wer?** | **Interesse und Bezug** |
| Einfache Benutzer | Möchten die Auswirkungen bezüglich der Ampelanlagen, Hindernissen und des Verhaltens der einzelnen Verkehrsteilnehmer beobachten. |
| Stadt/Land | Möchten die Auswirkungen bezüglich der Ampelanlagen, Hindernissen und des Verhaltens der einzelnen Verkehrsteilnehmer beobachten, um bei einem geplanten Bau/Umbau Staurisiken zu minimieren. |

Tabelle .:Stakeholder

# **Randbedingungen**

Bei einem Lösungsentwurf waren zu Beginn verschiedene Randbedingungen zu beachten. Dieser Abschnitt behandelt die Technischen und Organisatorischen Randbedingungen.

## **Technische Randbedingungen**

|  |  |
| --- | --- |
| **Randbedingung** | **Erklärung** |
| Implementierung in C# | Die gesamte Simulation basiert auf der Programmiersprache C#. Mit dieser ist es möglich eine einfache grafische Oberfläche zu gestalten und den dazugehörigen Code zu implementieren. |
| Moderate Hardwareausstattung | Der Betrieb der Simulation sollte auf einem marktüblichen Standard-Notebook ausführbar sein. |

Tabelle .: Technische Rahmenbedingungen

## **Organisatorische Rahmenbedingungen**

|  |  |
| --- | --- |
| **Rahmenbedingung** | **Erklärung** |
| Team | Hannes Kleiner, Andreas Lippmann, Stephanie Kaschnitz, Christopher Wieland, Martin Wieser |
| Zeitplan | Beginn der Entwicklung: 23.03.2017  Erster lauffähiger Prototyp: 11.05.2017  Fertigstellung bis 22.06.2017 |
| Vorgehensmodell | 1. Entwicklung einer Softwarearchitektur 2. Entwicklung des Softwaredesigns 3. Implementierung der Software   Zur Dokumentation der Architektur kommt arc42 zum Einsatz. |
| Entwicklungswerkzeuge | Erstentwurf mit Stift und Papier.  Architektur in Enterprise Architect.  Erstellung der C# Quelltexte in Visual Studio 2016. |
| Versionsverwaltung | Git |

Tabelle .: Organisatorische Rahmenbedingungen

# **Kontext**

Dieser Abschnitt beschreibt das Umfeld. Wie interagieren die Benutzer und mit welchen Fremdsystemen interagiert es?

## **Fachlicher Kontext**

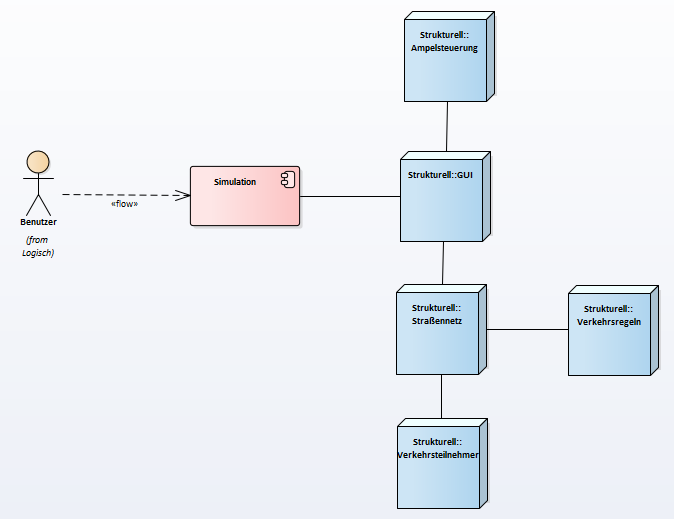


Abbildung .: Kontext

Benutzer

Ein Benutzer kann die Simulation starten und verschiedene Werte einstellen, wie zum Beispiel das Verhalten der Verkehrsteilnehmer oder die Ampelsteuerung.

GUI

Das Grafical User Interface (= GUI) behandelt die graphische Darstellung aller Elemente der Verkehrssimulation. Dabei werden zum Beispiel Ampeln, sich bewegende Autos und Straßen dem Benutzer angezeigt.

Ampelsteuerung (Eigener Prozess)

Bei der Ampelsteuerung handelt es sich um einen eigenen Prozess, welcher die verschiedenen Phasen (grün, gelb, rot) einer Ampel regelt.

Verkehrsregeln

Die Verkehrsregeln beinhalten allgemeine Regeln, welche bei ungeregelten und geregelten Kreuzungen eingehalten werden müssen.

Straßennetz

Das Straßennetz behandelt die Anzahl der Kreuzungen und der Ampeln, sowie kümmert es sich um die Verbindung der Kreuzungen mit Straßen. Auf dem Straßennetz werden die fahrenden Autos abgebildet.

Verkehrsteilnehmer

Unter Verkehrsteilnehmer versteht man LKWs und PKWs. Darunter fällt auch deren Verhalten, welches vom Benutzer veränderbar ist.

# **Lösungsstrategie**

Die folgende Tabelle stellt die Qualitätsziele von der Verkehrssimulation passenden Architekturansätzen gegenüber.

|  |  |
| --- | --- |
| **Qualitätsziel** | **Ansatz in der Architektur** |
| Parametrisierbarkeit | Schnittstellen für die Veränderungen des Verhaltens der Verkehrsteilnehmer und der Ampeln.  Geschieht über Json-Files. |
| Leicht bedienbar | Beim Start der Applikation öffnet sich zu der GUI eine Konsole, in der das momentane Verhalten der Verkehrsteilnehmer per Mausklick angezeigt und geändert werden kann. |
| Effizienz | Lesen/Schreiben von Json-Files benötigt wenig Zeit. |

Tabelle .: Lösungsstrategie

## **Aufbau der Verkehrssimulation**

Bei der Verkehrssimulation handelt es sich um ein C#-WPF-Programm, wobei der grafische Teil in xaml geschrieben und der benötigte Code für die Simulation in der dazugehörigen Main-Routine realisiert ist.

Der Code zerfällt in folgende Teile:

* Straßennetz
* Verkehrsregeln
* Ampelsteuerung
* GUI
* Verkehrsteilnehmer

Die Zerlegung ermöglicht es, einzelne Sachen einfach auszutauschen und in das Programm einzugliedern. Alle Teile sind durch Schnittstellen abstrahiert.

# **Bausteinschicht**

Dieser Abschnitt beschreibt die Zerlegung von der Simulation in Module, wie sie sich auch in der Struktur widerspiegelt.

# **Laufzeitschicht**

# **Verteilungsschicht**

# **Konzepte**

# **Entscheidungen**

Dieser Abschnitt befasst sich mit den Entscheidungen, welche im Team getroffen wurden.

## **Graphische Darstellung**

Die Fragestellung lautete, wie die graphische Darstellung funktionieren sollte. Ob man auf ein Fremdsystem wie zum Beispiel Unity zurückgreifen sollte.

Nachdem die Aufgabenstellung beinhalt, dass die gesamte Simulation in C# und Visual Studio zu implementieren ist, fiel die Entscheidung auf eine WPF-Applikation. Die WPF-Applikation bietet zu der Main eine grafische Oberfläche, welche einfach mit XAML zu programmieren ist. Geeinigt haben wir uns auf ein Koordinatensystem und Canvas.

## **Kommunikation zwischen den Gruppen**

Hierbei lautete die Fragestellung wie eine Kommunikation zwischen den einzelnen Simulationen stattfinden kann. Nachdem sich die Team-Mitglieder geeinigt haben, fiel das Ergebnis auf die Benutzung von RabbitMQ. Hierbei sendet jede Gruppe ihre Autos über ein zuvor allgemein definiertes Protokoll an eine andere Gruppe, diese bearbetet die Requests und fügt die Autos an die dementsprechenden Stellen ein.

# **Risiken**