Webanwendung zur Visualisierung von rettungsrelevanten Daten zur Einsatzunterstützung

Software-Entwicklungspraktikum (SEP)
Sommersemester 2023

Angebot

Auftraggeber
Technische Universität Braunschweig
Peter L. Reichertz Institut für Medizinische Informatik
Prof. Dr. Thomas Deserno
Mühlenpfordtstraße 23
38106 Braunschweig

Betreuer: Viktor Sobotta

Auftragnehmer:

Name	E-Mail-Adresse
Mohamed Wassim Chebili	m.chebili@tu-braunschweig.de
Omar Farouk Khayat	o.khayat@tu-braunschweig.de
Jonas Stepanik	j.stepanik@tu-braunschweig.de
Azhar Rahadian	a.rahadian@tu-braunschweig.de
Kacem Abdennabih	k.abdennabih@tu-braunschweig.de
Torben Oelerking	t.oelerking@tu-braunschweig.de
Qiyue Zhang	qiyue.zhang@tu-braunschweig.de

Braunschweig, 26. April 2023

Bearbeiterübersicht

Kapitel	Autoren	Kommentare
1	Qiyue Zhang	keine
1.1	Qiyue Zhang	keine
1.2	Qiyue Zhang	keine
2	Torben Oelerking	keine
3	Kacem Abdennabih	keine
3.1	Kacem Abdennabih	keine
3.2	Kacem Abdennabih	keine
4	Mohamed Wassim Chebili	keine
4.1	Mohamed Wassim Chebili	keine
4.2	Mohamed Wassim Chebili	keine
4.3	Omar Farouk Khayat	keine
5	Jonas Stepanik	keine
5.1	Jonas Stepanik	keine
5.2	Jonas Stepanik	keine
5.3	Jonas Stepanik	keine
6	Azhar Rahadian	keine
6.1	Azhar Rahadian	keine
6.2	Azhar Rahadian	keine
6.3	Azhar Rahadian	keine
6.4	Torben Oelerking	keine
7	Omar Farouk Khayat	keine

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

In der heutigen Welt erfordern Notfall- und Rettungseinsätze schnelle und effiziente Entscheidungsprozesse, die ein besseres Verständnis des Unfallgeschehens erfordern. Genau hier setzt die zu entwickelnde Webanwendung an - sie wird entwickelt, um Unfälle zu visualisieren und zu simulieren, wobei der Schwerpunkt auf den Daten von beschädigten Fahrzeugen und Verletzten liegt, um die Reaktion und das Management von Unfällen zu unterstützen.
Das Projekt besteht hauptsächlich aus 3 Teilen: einer Fahrzeugsimulation, einer Karte zur Visualisierung des Unfalls und einer Rettungskarte für den Fahrzeugtyp. Wenn ein Autounfall passiert, empfängt die Webanwendung durch eine ISAN (International Standard Accident Number) Informationen über den Unfall, darunter solche, die für das Erzeugen von einem 3D-Modell des beschädigten Fahrzeugs benutzt werden. Das Modell wird dann auf der Karte angezeigt. Wenn der Nutzer auf das Modell klickt, werden weitere Informationen zum Unfallfahrzeug wie die Anzahl der Insassen angezeigt und eine Verknüpfung zu einer Rettungskarte bereitgestellt.

1.1 Ziel

Die zu entwickelte Webanwendung ist ein Prototyp, der speziell für die Planung von Rettungseinsätzen konzipiert wird. Ziel ist es, dass die Möglichkeiten aufgezeigt werden, dass so ein System die Koordination und Organisation der für einen erfolgreichen Rettungseinsatz erforderlichen Ressourcen erleichtern kann. Dies wird erreicht indem die Autounfälle visualisiert und die Unfalldaten aggregiert werden.

1.2 Motivation

Autounfälle können in vielerlei Hinsicht erhebliche Auswirkungen auf die Gesellschaft haben. Weltweit sterben ca. 1,3 Millionen Menschen bei Verkehrsunfällen, und die finanziellen Auswirkungen belaufen sich in den meisten Ländern auf 3 % des BIP. Daher ist es wichtig, den Rettungskräften eine bessere Unterstützung zu bieten und die Effektivität bei der Rettung zu verbessern, um letztlich mehr Leben zu retten und die Folgen von Autounfällen zu verringern.

2 Formale Grundlagen

In diesem Kapitel werden die Formalen Grundlagen für das Projekt und die Entwicklung festgelegt.

Die Verwaltung des Softwareprojektes findet in dem vom Institut zur Verfügung gestellten GitLab Repository statt.
Die Dokumentation erfolgt mit den zur Verfügung gestellten Vorlagen, den entsprechenden Reglementierungen und dem Latex-Editor Overleaf und Draw.io für die Diagrammerstellung.
Die Reglementierungen, Richtlinien und aktuellen Terminen für die Entwickler im Rahmen des SEP finden sich auf der Webseite des ISF.

Die zu entwickelnde Softwarekomponente nimmt Informationen über einen Verkersunfall über die übertragende ISAN entgegen. Das System wertet die ISAN aus.
Anschließend wird auf einer Website mit eingebetteter Karte ein simuliertes 3D-Modell des Unfallfahrzeuges mit Unfallschaden auf der Karte angezeigt. Sowie bei Auswahl des Modells weitere Informationen.

Die Website zur Visualisierung des Unfalls mittels Karte und 3D-Modell ist mit dem Typescript-Framework Angular zu realisieren, um die Kompatbilität mit bereits vorhandenen Softwarekomponenten zu gewährleisten und zu vereinfachen.

Die ISAN enthält die unfallrelevanten Daten (z.B Koordinaten, aufgetretene Kräfte, IMU-Daten, Fahrzeugtyp, Anzahl der Fahrzeuginsassen, Fahrtrichtung usw.).

Der Kartenanbieter ist abhängig von der zur Verfügung stehenden API sowie weiterer Beschränkungen wie begrenzte Anzahl der Anfragen oder etwaiger Kosten. Mögliche Kartendienste sind z.B. Google Maps oder OpenStreetMap. Es wird sich im Laufe des Projektes herausstellen, welche sich besser für die Anzeige des 3D-Modells eignet.

Das Einbetten des 3D-Modells in die Website bzw. das Rendern im Browser soll mittels WebGL und Three.js bewerkstelligt werden.

Das Erstellen des 3D-Modells (Mesh) erfolgt mittels der Simulationssoftware BeamNG.tech (Forschungslizenz) und den Daten aus der ISAN. Die Simulation des Unfallschadens am Fahrzeug

wird durch das Fahren gegen ein starres Hindernis simuliert. Es findet keine Simulation der Unfallschäden mehrerer Unfallbeteiligter, ineinander gefahrener Fahrzeuge statt. Jedes gemeldete Fahrzeug wird für sich allein simuliert und angezeigt.
Mögliche Verletzungen der Insassen sollen ebenfalls aus der Simulationdes Autounfalls mit Hilfe eines Dummys ermittelt werden und als 3D-Ansicht mit Hilfe der übermittelten Kräfte als Texture/Vertexcolor Heatmap dargestellt werden. Die Simulation wird über die Python API BeamNGpy angesprochen. Nach erfolgreicher Simulation wird das Fahrzeugmodell exportiert. Der Ablauf und die Funktionsweise, sowie die verwendete Sprache/Protokolle des Exports müssen noch geklärt werden.

Zu Testzwecken ist zudem eine ISAN zu erstellen und zu hinterlegen, sodass die Simulation stattfinden kann.

Die Rettungskarte liegt bereits als fertige Softwarekomponente (geschrieben in Angular) vor.

3 Projektablauf

Die Abgabetermine und zugehörigen Meilensteinbezeichnungen werden in der folgenden Tabelle sowie in Abbildung 3.1 präsentiert. Das Projekt startet mit dem Kick-off-Meeting am 13.04.2023 und endet am Tag der jungen Software EntwicklerInnen am 20.07.2023.

3.1 Meilensteine

Nummer	Meilenstein	Dokumente	Abgabetermin
1	Projektstart	-	13.04.2023
2	Angebot	Angebot	26.04.2023
3	Pflichtenheft	Pflichtenheft	17.05.2023
4	Abnahmetestspezifikation	Abnahmetestspezifikation	17.05.2023
5	Prototyp Website & Simulation	-	23.05.2023
6	Zwischenpräsentation	-	26.05.2023
7	Fachentwurf	Fachentwurf	07.06.2023
8	Technischer Entwurf	Technischer Entwurf	28.06.2023
9	Vorletzte Version des Produkts	-	30.06.2023
10	Testdokumentation	Testdokumentation	12.07.2023
11	Funktionierendes Produkt	-	15.07.2023
12	TDSE*	-	20.07.2023

* Tag der jungen Software EntwicklerInnen

3.2 Geplanter Ablauf

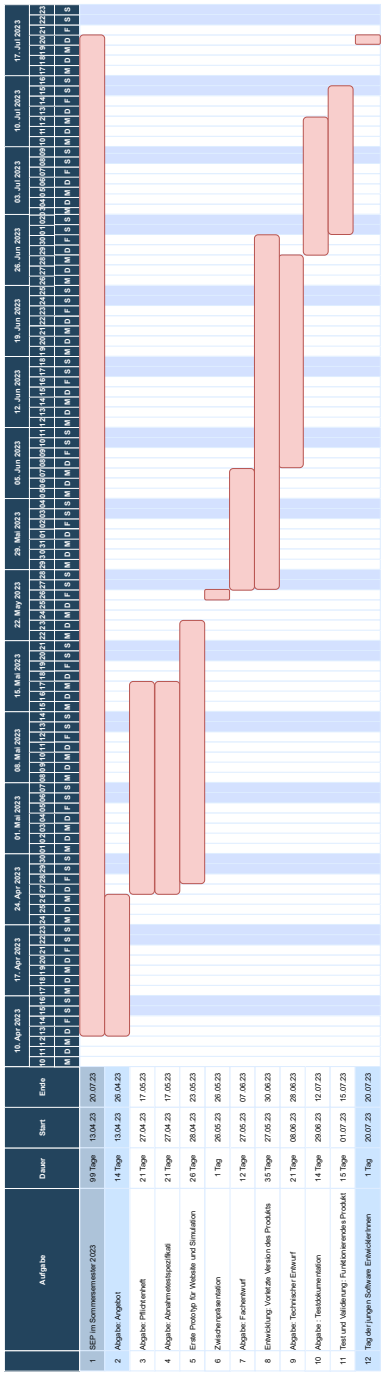

Abbildung 3.1: Gantt-Diagramm

4 Projektumfang

Dieses Kapitel dient dazu, den Projektumfang zu definieren. Dies umfasst den Lieferumfang, den Kostenplan und den funktionalen Umfang.

4.1 Lieferumfang

Am Ende des Projekts wird der Quellcode der Webanwendung geliefert. Dies beinhaltet sowohl die Backend- als auch die Frontend-Anwendungen. Ausreichende Dokumentation jeder Entwicklungsstufe wird auch vorgelegt.
Diese Umfassen: Angebot, Pflichtenheft, Abnahmetestspezifikation, Fachentwurf, Technischer Entwurf und Testdokumenation.

4.2 Kostenplan

Das Team besteht aus 7 Mitgliedern. Für die Planung und die Dokumentation werden 100 Arbeitsstunden pro Person veranschlagt, während für die Programmierung und das Testen der Software 60 Arbeitsstunden pro Person anfallen. Bei einem Stundensatz von 100€ ergeben sich Gesamtkosten von 112.000€.

	Stunden pro Entwickler	**Gesamtstunden**	**Kosten**
Planung	30	210	21.000€
Dokumentation	70	490	49.000€
Entwicklung	40	280	28.000€
Testing	20	140	14.000€
Gesamtkosten	-	-	**112.000€**

4.3 Funktionaler Umfang

Unser Projekt besteht aus einem Frontend-Teil und einem Backend-Teil. Hier wird beschrieben, welche Funktionalitäten der Server der Client-Anwendung und die Client-Anwendung dem Nutzer zur Verfügung stellen.

4.3.1 Server

Die zu entwickelnde serverseitige Softwarekomponente nimmt zur Simulation des Unfalls Informationen über einen Verkehrsunfall über die übertragende ISAN entgegen. Das System wertet die ISAN aus und extrahiert die benötigten Daten (z.B Koordinaten, aufgetretene Kräfte, IMU-Daten, Fahrzeugtyp, Anzahl der Fahrzeuginsassen, Fahrtrichtung usw.).
Mit Hilfe dieser Daten wird eine Unfallsimulation gestartet. Die Simulation des Unfallschadens am Fahrzeug wird durch das Fahren gegen ein starres Hinderniss simuliert. Es findet keine Simulation der Unfallschäden mehrerer Unfallbeteiligter, ineinander gefahrener Fahrzeuge statt. Jedes gemeldete Fahrzeug wird für sich allein simuliert und angezeigt. Es wird eine neue Simulation/Szenario eines Unfalls mit den benötigten Parametern aus der ISAN erstellt. Nach erfolgreicher Simulation wird das Fahrzeugmodell an die Webanwendung exportiert.
Der Server stellt zusätzlich relevante Daten (z.B. Anzahl Passagiere, Standort, usw.) des Unfalls der Webanwendung zur Verfügung stellen.

4.3.2 Webanwendung

Auf einer Website mit eingebetteter Karte wird ein 3D-Modell des Unfallfahrzeuges mit Unfallschaden auf der Karte angezeigt. Das 3D-Modell befindet sich auf der Karte an den Koordinaten des Unfallorts und ist in die richtige Fahrtrichtung ausgerichtet.
Das 3D-Modell und die rettungsrelevanten Daten werden vom Server zur Verfügung gestellt. Die Karte zoomt auf den Unfallort. Die Karte kann vom Nutzer bewegt und vergrößert und verkleinert werden. Durch ein Klicken auf das 3D-Modell mittels Mauszeiger erhält der Nutzer weitere Informationen zur Anzahl der Insassen des Fahrzeuges, sowie eine Verknüpfung zu einer externen Rettungskarte, die Informationen über das Fahrzeug beinhaltet. Zudem wird eine Texture/Vertexcolor-Heatmap angezeigt, die eine Visualisierung der Verletzungen ermöglicht.

5 Entwicklungsrichtlinien

In diesem Kapitel geht es um die Entwicklungsrichtlinien, die bei der Programmierung der Software eingehalten werden sollen, um eine effektive und standardisierte Entwicklung zu gewährleisten.

5.1 Konfigurationsmanagement

Es gibt ein zentrales Repository, welches in zwei Unterordner getrennt ist: Visualisierung (Website, HTML/CSS/Typescript), Simulation (Python).

Es soll einen Haupt-Branch geben und mehrere Entwicklungs-Branches.

Jeder Commit auf dem Haupt-Branch sollte lauffähigen Code pushen und ein Kommentar sollte genau angeben, was geändert wurde. Probleme im Code werden bei GitLab als Issues vermerkt, sodass ständig sichtbar ist, welche Probleme derzeit bestehen, welche Gedanken sich dazu gemacht wurden und welche Probleme behoben sind. Die Entwicklungs-Branches werden während der Entwicklung einzelner Features benutzt und zuletzt in den Haupt-Branch gemerged.

5.2 Design- und Programmierrichtlinien

Im Code sollte für jede Klasse ein Kommentar existieren, der erklärt, was die Klasse repräsentiert, welche Attribute sie hat und welche Methoden. Für jede Methode sollte außerdem ein Kommentar angeben, was die Methode macht, welche Argumente sie annimmt und was sie ausgibt.

Im Code wird für jegliche Kommentare und Bezeichner die englische Sprache genutzt.

In Typescript wird der Angular coding style guide eingehalten. In Python wird der PEP8 style eingehalten.

5.3 Verwendete Software

Overleaf	LaTeX
Angular	Web
Draw.io	Diagramme
Gitlab, git	Softwareversionierung
VSCode, Vim, PyCharm, Jupyter Notebook	IDEs/Texteditors

6 Projektorganisation

In diesem Kapitel geht es um die interne und externe Kommunikation des Teams sowie die Teamstruktur und die Aufteilung des Projektes auf die Teammitglieder.

6.1 Schnittstelle zum Auftraggeber

Das Entwicklungsteam kommuniziert regelmäßig per Mail mit dem Auftraggeber und jede Woche wird ein Teamtreffen abgehalten, bei dem alle Teammitglieder und Auftraggeber entweder persönlich oder über ein Medium wie Discord teilnehmen werden. Der Ansprechpartner seitens des Auftragnehmers ist Torben Oelerking.

Kontakt Ansprechpartner Auftraggeber	
Name	Viktor Sobotta
Institut	PLRI Braunschweig
Adresse	Mühlenpfordstraße 23
Raum	IZ 439
E-Mail	v.sobotta@tu-braunschweig.de
Tel.	+49 531 391-9518
Kontakt Ansprechpartner Auftragnehmer	
Name	Torben Oelerking
E-Mail	t.oelerking@tu-braunschweig.de

6.2 Schnittstelle zu anderen Projekten

Es müssen folgende Schnittstellen zu anderen Softwarekomponenten existieren:

- Damit die Rettungskräfte auf die Daten eines „richtigen" Unfall zugreifen können, muss eine Schnittstelle existieren, welche via ID auf den „richtigen" Unfall zoomt. „Richtiger" Unfall im Zusammenhang mit der neu entwickelten Softwarekomponente und anderen bestehenden Komponenten im Zusammenspiel im Praxiseinsatz.
- Die Realisierung des Zugriffs auf die entsprechende 3D Rettungskarte ist ebenfalls mittels Schnittstelle darzustellen.

6.3 Interne Kommunikation

Das Entwicklungsteam wird über Telefon, WhatsApp und Discord kommunizieren. Einmal wöchentlich wird sich das Team sich zu einem ausführlichen Meeting treffen (online oder präsenz), um den aktuellen Stand des Projektes zu besprechen. Allerdings ist die Kommunikation nicht nur auf diese wöchentlichen Treffen beschränkt. Probleme werden über verschiedene Kommunikationskanäle wie z.B. Discord diskutiert und gemeinsam bei den wöchentlichen Meetings gelöst. Dringende Probleme werden jedoch umgehend gelöst und durch spontane Meetings behoben.

6.4 Teamstruktur

In diesem Projekt gibt es insgesamt 7 Mitglieder. Das Projekt wird in zwei Teilprojekte, die Simulation und die Webanwendung aufgeteilt. Die Gruppenmitglieder werden wie folgt verteilt:

BeamNG Simulation	**Webanwendung**
Azhar Rahadian	Torben Oelerking
Jonas Stepanik	Qiyue Zhang
-	Omar Farouk Khayat
-	Mohamed Wassim Chebili
-	Kacem Abdennabih

Innnerhalb der Entwicklerteams werden die anfallenden Aufgaben jede Woche auf ihren Fortschritt überprüft und gegebenenfalls neu vergeben. Hierbei werden die Kompetenzen und Interessen der einzelnen Teammitglieder berücksichtigt, um einen flüssigen Entwicklungsprozess zu gewährleisten.

7 Glossar

Angular ist ein TypeScript-basiertes Front-End-Webapplikationsframework. Es wird von einer Community aus Einzelpersonen und Unternehmen, angeführt durch Google, entwickelt und als Open-Source-Software publiziert.

BeamNGpy ist eine offizielle Bibliothek, die eine Python-API für BeamNG.tech bereitstellt, den akademie- und industrieorientierten Fork des Videospiels BeamNG.drive.

Branch in Git repräsentiert eine unabhängige Entwicklungslinie.

CSS ist eine Stylesheet-Sprache für elektronische Dokumente und zusammen mit HTML und JavaScript eine der Kernsprachen des World Wide Webs.

Discord ist eine Online-Plattform, die hauptsächlich für die Kommunikation in Gruppen und Communitys verwendet wird. Benutzer können sich auf verschiedenen Servern anmelden und in Text- oder Sprachkanälen miteinander kommunizieren.

Frontend und **Backend** werden in der Informationstechnik an verschiedenen Stellen in Verbindung mit einer Schichteneinteilung verwendet. Dabei ist typischerweise das Frontend näher am Benutzer, das Backend näher am System.

GitLab ist ein Version Control System (VCS). Es basiert vollständig auf Git, einem verteilten Versionierungssystem, das als Open-Source-Software zur Verfügung gestellt wird.

Google Maps Online Kartendienst

HTML ist eine textbasierte Auszeichnungssprache zur Strukturierung elektronischer Dokumente wie Texte mit Hyperlinks, Bildern und anderen Inhalten.

IMU (inertial measurement unit) ist ein elektronisches Gerät, welches Daten über die Beschleunigung, die Winkelgeschwindigkeit und Orientierung eines Körpers ausgibt.

ISAN (International Standard Accident Number) Das ISAN-Projekt zielt darauf ab große Da-

tenmengen aus diversen Quellen, wie der Notfallmedizin (EMS), der elektronischen Gesundheitsakte (EHR) und Ereignisdatenschreibern (EDR) zu sammeln und durch die Schaffung einer technischen Grundlage zu verbinden und Rettungseinsätze zu unterstützen. Die ISAN selbst enthält Unfalldaten.

Jupyter Notebook Jupyter Notebook ist eine interaktive Programmierumgebung, die es Benutzern ermöglicht, Code in verschiedenen Programmiersprachen (z.B. Python, R, Julia) auszuführen und dabei gleichzeitig Text, Bilder und andere Medien in einem Dokument zu integrieren.

Klasse in der objektorientierten Programmierung ist ein abstraktes Modell bzw. einen Bauplan für eine Reihe von ähnlichen Objekten.

Meilensteine sind erreicht Zwischenziele/ Etappen im Entwicklungsprozess.

Merging ist in Git der Weg, eine getrennte Entwicklungshistorie wieder zusammenzuführen.

Methoden sind in der objektorientierten Programmierung Unterprogramme in der Form von Funktionen oder Prozeduren, die das Verhalten von Objekten beschreiben und implementieren.

OpenStreetMap Open-Source online Kartendienst

PEP8 ist der „Style Guide for Python Code", also die Richtlinie zum Formatieren von Python-Code, an die sich jeder Python-Programmierer hält.

Python ist eine einfach zu erlernende, interpretierte, objektorientierte Programmiersprache mit dynamischer Typisierung.

Repository ist eine Arbeitskopie der Entwickler das den vollständigen Verlauf aller Änderungen enthält. Zusätzlich gibt es noch ein remote Repository in dem alle lokalen Änderungen zusammenkommen.

Rettungskarte ist eine bereits vorhandene Softwarekomponente, die Informationen eines spezifischen Fahrzeugs enthält, welche den Rettungskräften helfen, das Fahrzeug gefahrlos zu öffnen.

Texture/Vertex Heatmap ist ein Diagrammtyp zur farblichen Visualisierung von Daten im zweidimensionalen Raum/ einer Ebene.

Three.js ist eine browserübergreifende JavaScript-Bibliothek und Anwendungsprogrammierschnittstelle (API) zur Erstellung und Anzeige von animierten 3D-Computergrafiken in einem

Webbrowser mit WebGL.

TypeScript ist eine kostenlose und Open-Source-Programmiersprache, die auf JavaScript aufbaut und statische Typisierung unterstützt.

WebGL (Web Graphics Library) ist eine JavaScript-Programmierschnittstelle, mit deren Hilfe 3D-Grafiken hardwarebeschleunigt im Webbrowser ohne zusätzliche Erweiterungen dargestellt werden können.

Whatsapp ist eine Messaging-Anwendung, die es Benutzern ermöglicht, Text-, Sprachnachrichten, Bilder, Videos und andere Dateien an andere WhatsApp-Benutzer zu senden. Es wird häufig verwendet um persönliche und geschäftliche Nachrichten auszutauschen und sich zu vernetzen.

3D-Mesh ist eine dreidimensionale Struktur, die eine 3D-Objekt darstellt.