Praxis der Software-Entwicklung

# Entwicklung eines Campus-Routenplaners

SS 2016

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

## 1 Wichtige Vorbemerkung

Dies ist *Ihr* Projekt. Dieses Dokument ist kein Katalog von Aufgaben, der Punkt für Punkt abgearbeitet werden muss, um nachher den Schein zu bekommen, sondern lediglich eine Reihe von Vorschlägen bzw. Hinweisen, was Sie tun können bzw. was wir erwarten. Wie *Ihr* Programm nachher aussieht, müssen *Sie* selbst entscheiden.

## 2 Aufgabe

Ihre Aufgabe in diesem Projekt ist die Entwicklung eines Campus-Routenplaners für den KIT-Campus. Ihr System sollte die kürzeste Route zwischen zwei Orten auf dem Campus berechnen und anzeigen können. Dabei muss es auch möglich sein, innerhalb von Gebäuden zu routen, wobei dem Benutzer dann der Grundriss des entsprechenden Gebäudes angezeigt werden soll. Die Suche nach einzelnen Orten (ohne die Berechnung einer Route) ist ein weiteres wichtiges Feature.

Zusätzlich zu dieser Routingsoftware muss es ein Administrationswerkzeug geben, mit dem Kartendaten erstellt werden können. Dabei sollte es auch für Laien möglich sein, zum Beispiel neue Gebäude (mit oder ohne Grundriss), sowie neue Knoten und Kanten, hinzuzufügen.

### 2.1 Routing

Das Routing startet mit der Benutzereingabe, bei der es leicht möglich sein sollte ein Gebäude, einen Raum oder auch einen Hörsaal als Start- oder Zielpunkt anzugeben. Bei der Berechnung eines kürzesten Weges soll Dijkstras Algorithmus zur Anwendung kommen.

Welcher Pfad der kürzeste ist, hängt natürlich von der Metrik ab, die Sie verwenden. Eine sinnvolle Metrik wäre zum Beispiel die (konstante) Laufgeschwindigkeit eines Fußgängers.

### 2.2 Kartendaten – Administrationswerkzeug

Um Kartendaten für Ihre Routingsoftware zu erzeugen, sollten Sie Ihr eigenes Administrationswerkzeug verwenden. Als Hintergrundbild können Sie den KIT-Campusplan verwenden (`https://www.kit.edu/downloads/Campus-Sued.pdf`). Um das Routing zu ermöglichen muss außerdem ein Graph erstellt werden, der Wege (Kanten) sowie Kreuzungen von Wegen (Knoten) modelliert. Außerdem muss man zusätzliche Daten speichern können, wie zum Beispiel den Namen eines Gebäudes an einem Knoten, der dessen Eingang repräsentiert. Zu Demonstrationszwecken sollten Sie das Innere von zumindest drei Gebäuden, sowie das Wegenetz des Campus-Süd erstellen.

### 2.3 Darstellung der Karte

Die einfachste Art und Weise eine Karte anzuzeigen eine Bilddatei, auf die man dann ggf. einen berechneten kürzesten Wege (oder andere Informationen) zeichnet. Wenn die Route innerhalb eines Gebäudes startet oder endet, dann muss der Grundriss des entsprechenden Gebäudes (zusammen mit der Route innerhalb des Gebäudes) angezeigt werden.

Da die gesamte Karte zu groß ist um auf den Bildschirm zu passen, muss es möglich sein in der Karte zu zoomen und sie zu verschieben.

### 2.4 Vorschläge für coole Features

Die folgende Liste enthält Vorschläge für mögliche Features. Sie sollten sich aber auch selber Gedanken darüber machen, welche zusätzlichen Features Sie gerne in Ihrem System hätten.

- Es könnte interessant sein, verschiedene Metriken für das Routing zu verwenden. Mögliche Entscheidungen sind zum Beispiel: Aufzug vs. Treppe; Kreuzung des Adenauerrings vs. Nutzung der Brücke ...
- Die Möglichkeit neue Kanten-/Knotenattribute im Administrationswerkzeug anlegen zu können.
- Ein Javaapplet für das Routingsystem wäre schön.
- Es könnte hilfreich sein, wenn der Administrator einzelne Straßen temporär als gesperrt markieren könnte (zum Beispiel bei Baustellen).
- Manchmal führt der kürzeste Weg durch ein Gebäude. Wie wird das bei Ihrem System dargestellt?
- ...

## 3 Ablauf

### 3.1 Arbeitsweise

Ziel dieser Lehrveranstaltung ist, Verfahren des Software-Entwurfs und der Qualitätssicherung praktisch einzusetzen, Implementierungskompetenz umzusetzen, und arbeitsteilig im Team zu kooperieren. Entsprechend orientiert sich der Ablauf an den Prinzipien der Softwaretechnik. Deutlich wird dies auch am Zeitplan der Veranstaltung (siehe Abschnitt 3.4).

Es geht also nicht darum irgendein System irgendwie zu implementieren, das dann irgendwie funktioniert, sondern phasenweise vorzugehen, wie im Zeitplan angegeben. Entsprechend sind von Ihnen nicht nur das fertige System sondern nach jeder Phase auch das zugehörige Dokument abzugeben:

- Pflichtenheft (ca. 30 Seiten)
- Entwurf (ca. 80 Seiten)
- Implementierungsbericht (ca. 15 Seiten)

- Testbericht (ca. 25 Seiten)

Zum Abschluss der Veranstaltung müssen Sie Ihr System in einer Präsentation vorstellen. Am Ende jeder Phase findet zudem ein Kolloquium statt. Die jeweiligen Termine werden während des Semesters vereinbart.

Der vorgegebene Zeitplan (Abschnitt 3.4) soll nicht nur eine Hilfestellung sein, sondern wir erwarten, dass dieser von ihnen auch eingehalten wird. Dazu wird in jeder Phase ein *Phasen-Verantwortlicher* bestimmt, der für den korrekten Phasenablauf und die fristgerechte Abgabe der Dokumente verantwortlich ist.

## 3.2 Dokumentation

Die von Ihnen erstellte Dokumentation ist ein Hauptbestandteil Ihres Projekts. Ohne eine vollständige und umfassende Dokumentation Ihrer Tätigkeit ist es nicht möglich die Veranstaltung erfolgreich zu bestehen.

Es wird erwartet, dass Sie Ihre Planung und Ihr anschließendes Vorgehen präzise, vollständig und konsistent dokumentieren. Was den Inhalt und die Form der Dokumente betrifft, verweisen wir an dieser Stelle ausdrücklich auf die Inhalte der Softwaretechnikvorlesung.

Die Benutzung und der Betrieb eines Versionsmanagementtools für Quellcode **und** Dokumentation ist Pflicht. Wir empfehlen den Einsatz von GIT.

## 3.3 Empfohlene Tools

Die empfohlene Programmiersprache der Veranstaltung ist Java. Andere Sprachen sind nur nach vorheriger Absprache erlaubt.

Das JUnit Testing Framework[1] ist der defacto Standard für Unit Tests unter Java. Wir empfehlen den Einsatz dieses Tools dringend. Sowohl im WWW, als auch in der Bibliothek finden Sie die grundlegende Literatur zur Handhabung von JUnit. Sie sollten außerdem eine möglichst hohe Überdeckung des Codes durch die Tests erreichen. Die Codeüberdeckung können Sie mit einem Werkzeug wie z.B. EclEmma[2] berechnen. Beide Tools sind als Plugins in die Programmierumgebung Eclipse[3] integriert, so dass wir diese auch als IDE für Ihr Projekt empfehlen. Für den Entwurf empfiehlt sich ein UML-Tool. Lucidchart[4] ist für Studenten kostenlos.

## 3.4 Zeitplan

- **Erstes Gruppentreffen:** 29. April 2016
- **Abgabe des Pflichtenhefts:** 24. Mai 2016, kurz danach müssen Sie ihr Pflichtenheft im Rahmen eines Kolloquium erläutern (Termin nach Vereinbarung).
- **Abgabe des Entwurfs:** 21. Juni 2016, kurz danach müssen Sie ihren Entwurf im Rahmen eines Kolloquiums verteidigen (Termin nach Vereinbarung).

[1] `http://junit.org/`
[2] `http://marketplace.eclipse.org/content/eclemma-java-code-coverage`
[3] `http://www.eclipse.org/`
[4] `https://www.lucidchart.com/pages/education/students`

- **Abgabe Implementierungsbericht:** 19. Juli 2016, kurz danach findet das Implementierungskoloquium statt (Termin nach Vereinbarung).
- **Klausurpause:** Individuelle 2 Wochen nach eigenem Ermessen während Validierung
- **Abgabe des Testberichtes und Systemabnahme:** 23. August 2016.
- **Abschlusspräsentation:** 31. August 2016.

Bitte beachten Sie, dass dieser Ablaufplan vorläufig ist und sich die endgültigen Daten noch leicht verschieben können. Ist dies der Fall, wird dies auch rechtzeitig bekannt gegeben.

### 3.5 Ansprechpartner


- **Valentin Buchhold**, Institut für Theoretische Informatik, Prof. Dr. Dorothea Wagner. E-Mail: `buchhold@kit.edu`, Tel.: 0721 608-46605, Büro: Raum 308, Gebäude 50.34.
- **Michael Hamann**, Institut für Theoretische Informatik, Prof. Dr. Dorothea Wagner. E-Mail: `michael.hamann@kit.edu`, Tel.: 0721 608-47331, Büro: Raum 322, Gebäude 50.34.


## 4 Bewertung

### 4.1 Minimale Leistungsmerkmale

Ihr System besteht aus zwei Hauptkomponenten, dem Administrationswerkzeug zur Erstellung der Karten und der Routingsoftware für die Benutzer. Ihr System soll folgende minimale Leistungsmerkmale aufweisen:

**Administrationswerkzeug** Folgende Operationen müssen unterstützt werden:

- Änderung des Hintergrundbildes der Karte.
- Hinzufügen, Löschen und Bewegen von Knoten und Kanten.
- Modifikation von Kantenattributen, wie zum Beispiel der Reisezeit (auch wenn es dafür einen Standardwert gibt, der von ihrer Länge abhängt).
- Modifikation von Knotenattributen, wie zum Beispiel Raumnummer oder Name des Hörsaals.
- Hinzufügen von Gebäudegrundrissen.

**Routingsoftware** Die Routingsoftware muss folgende Features haben:

- Benutzerfreundliche und selbsterklärende Benutzerschnittstelle (engl. graphical user interface (GUI)).
- Schnelle Routenberechnung ($< 1$ Sekunde).
- Ansprechende Darstellung der berechneten Route (sowohl innerhalb als auch außerhalb von Gebäuden).

### 4.2 Referenzsystem

Das System soll auf einem Standard-PC unter Linux lauffähig sein. Zum Testen stehen im Raum 305 Poolrechner bereit. Es handelt sich dabei um PCs mit einem Intel E7200@2.66 Ghz Prozessor mit 4GB RAM und SuSe 12.2.

### 4.3 Benotung

Die Benotung Ihres Systems richtet sich nach folgenden Kriterien:

- Qualität aller abgegebenen Dokumente
- Qualität der Kolloquien
- Erfüllung der minimalen Leistungsmerkmale (siehe Abschnitt 4.1)
- Sinnvolle Erweiterungen über diese Merkmale hinaus
- Konsistenz zwischen den beiden Teilen ihres Systems
- Robustheit des erstellten Systems

Diese Liste hat keine Reihenfolge, die einer Gewichtung entspricht.

Die Note für die Veranstaltung Teamarbeit in der Software-Entwicklung bewertet die Softskill-Aspekte des Projekts. Dazu gehören insbesondere die Präsentationen, das Arbeiten in der Gruppe und Ihre Tätigkeit als Phasenverantwortliche.

Zum Bestehen müssen des Weiteren alle in Abschnitt 3.1 genannten Dokumente *rechtzeitig* abgegeben werden (siehe Zeitplan in Abschnitt 3.4).

Jede Phase Ihres Projektes wird getrennt bewertet. Die Endnote ergibt sich dann aus den Noten der einzelnen Phasen, gewichtet nach Tabelle 1.

| **Phase** | **Gewicht** |
|---|---|
| Pflichtenheft | 10% |
| Entwurf | 30% |
| Implementierung | 30% |
| Testphase | 20% |
| Abschlusspräsentation | 10% |

Table 1: Gewichtung der einzelnen Phasen.