|  |
| --- |
|  |
|  |
|  |

# Projekt **1:** Anforderungsdokumentation

GraphTool

Projekt Management

Tina Gerber, Daria Schumacher

# Inhalt

[1 Projektziele 3](#_Toc420436711)

[2 Teilziele 3](#_Toc420436712)

[3 System und Systemkontext 4](#_Toc420436713)

[4 Funktionale Anforderungen 5](#_Toc420436714)

[5 Qualitätsanforderungen 8](#_Toc420436715)

[6 Randbedingungen 8](#_Toc420436716)

[7 Glossar 9](#_Toc420436717)

# Projektziele

Das Ziel des Projektes GraphTool ist folgendes:

|  |  |
| --- | --- |
| Nr. | Projektziele |
| P1 | Die Software soll das Erarbeiten und das Erlernen von Graphen-Algorithmen durch deren Visualisierung erleichtern. |

# Teilziele

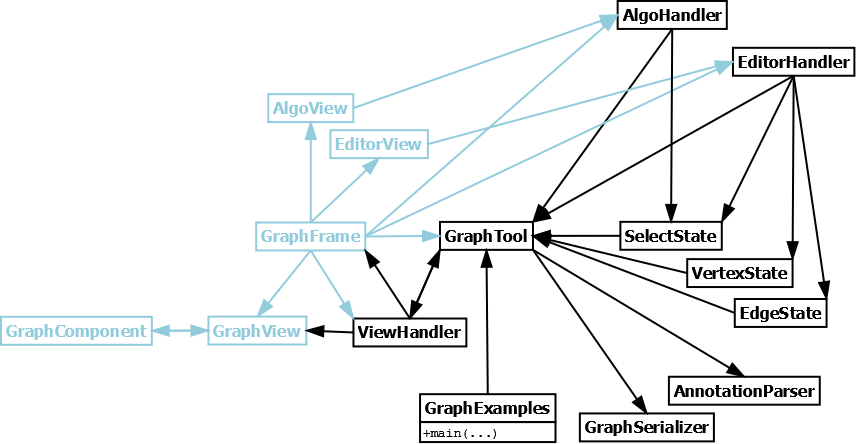
Die Teilziele des Projektes GraphTool sind folgende:

|  |  |
| --- | --- |
| Nr. | Teilziele |
| T1 | Das Tool ermöglicht Zeiteinsparungen beim Erstellen eines Graphen. |
| T2 | Das Tool ermöglicht Zeiteinsparungen bei der Suche nach Fehlern in programmierten Algorithmus. |
| T3 | Das Tool kann gleichzeitig nur einen Graphen bearbeiten. |
| T4 | Das Tool kann gleichzeitig nur einen Algorithmus darstellen. |
| T5 | Das Tool führt Algorithmen in angemessener Zeit durch. |
| T6 | Das Tool ist für Studenten und Dozenten im Informatikbereich leicht bedienbar. |
| T7 | Das Projekt muss am 12. Juni 2015 fertig sein. |

# System und Systemkontext

System

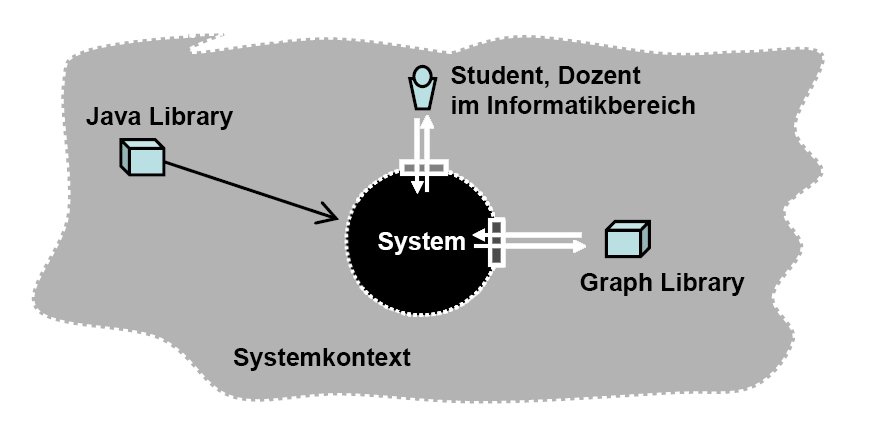
Das System des Tools soll möglichst dem Model View Controller Prinzip folgen und die graphische Ebene von der logischen Ebene lösen. Die graphische Ebene soll alle Aufgaben an die logische Ebene weiterleiten. Die graphische Ebene ist in der folgenden Graphik blau dargestellt und die logische Ebene schwarz:



Systemkontext

Das Tool soll Studenten und Dozenten im Informatikbereich als Lehrmittel sowie Lernmittel dienen. Somit definieren wir künftig als User Studenten und Dozenten im Informatikbereich.

Es soll als Java-Applikation umgesetzt und als Java-Source-Code an Studenten und Dozenten weitergegeben werden. Die Applikation soll als Basis eine bereits existierende Library nutzen.



# Funktionale Anforderungen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
| P | Priorität |  | 🡻 | niedrig |
| V | Variabilität |  | 🡾 | niedrig-mittel |
| K | Komplexität |  |  | mittel |
| R | Risiko |  | 🡽 | mittel-hoch |
|  |  |  |  | hoch |
|  | | | | |
| Quelle | | | | |
| RE | Requirements Engineering | | | |
| TS | Teamsitzung | | | |
| CS | Coachsitzung mit Herr Schwab | | | |

|  |  |
| --- | --- |
| Ziele | |
| P | Projektziele |
| T | Teilziele |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nr. | | Kurzbezeichnung | Status | P | V | K | R | Quelle | Datum | Ziele |
| 1 |  | Graphen Editor |  | | | | | | | |
|  | 1.1 | Graphen zeichnen und editieren | geplant |  |  |  |  | CS |  | P1  T1 |
| 1.2 | Graphen speichern,  laden und löschen | geplant | 🡻 | 🡻 |  | 🡾 | CS |  | T3 |
|  | 1.3 | Attribute für Vertex und Edge erstellen | geplant |  |  |  |  | CS |  | P1 |
|  | 1.4 | Attribute des Graphen darstellen | geplant |  |  |  |  | CS |  | P1 |
|  | 1.5 | Aktionen rückgängig machen und wiederherstellen | geplant |  |  |  |  | CS |  | T6 |
| 2 |  | Algorithmen Visualisierung |  | | | | | | | |
|  | 2.1 | Gekennzeichnete Algorithmen finden | geplant |  |  |  |  | CS |  | T5 |
| 2.2 | Algorithmen abspielen und stoppen | geplant |  |  |  |  | CS |  | P1 |
| 2.3 | Algorithmen schrittweise abspielen | geplant |  |  |  |  | CS |  | P1 |
| 2.4 | Abspieltempo einstellen | geplant | 🡻 | 🡻 |  | 🡾 | CS |  | T5 |
|  | 2.5 | Ergebnisse der Algorithmen an Graphen anwenden | geplant | 🡻 | 🡻 |  | 🡾 | CS |  | T1 |
|  | 2.6 | Graphen speichern und  laden | geplant | 🡻 | 🡻 |  | 🡾 | CS |  | T4 |

1 Graphen Editor

1.1 Graphen zeichnen und editieren

Der Graphen Editor soll es dem User ermöglichen, Graphen zu zeichnen und zu editieren. Dazu soll der User ein Zeichnungsfeld und eine Auswahl an Tools haben, die es ihm ermöglichen, Vertices und Edges zu selektieren, verschieben, erstellen und zu löschen.

1.2 Graphen speichern, laden und löschen

Der Graphen Editor soll es dem User erlauben, Graphen zu speichern, zu laden oder gespeicherte Graphen zu löschen. Dabei soll der User die Möglichkeit haben, Graphen unter einem von ihm selbst definierten Namen an einem vordefinierten Ort abzuspeichern und später unter demselben Namen wieder zu laden oder zu löschen.

Ausserdem soll beim Speichern die Unterscheidung gemacht werden, ob der User den Graphen zum ersten Mal, unter demselben Namen oder als neue Datei unter einem anderen Namen speichern will. Speichert der User den Graphen zum ersten Mal, soll der Graph Editor vom User einen Namen verlangen. Bei jedem weiterem Mal wird der Graph automatisch unter demselben Namen gespeichert. Will der User den Graphen unter einem neuen Namen speichern, soll der Graphen Editor wiederum einen Namen verlangen.

Zusätzlich soll der User davor gewarnt werden, wenn er versucht, einen Graphen unter einem bereits existierenden Namen zu speichern. Ebenso soll der User nochmals gewarnt werden, bevor er einen Graphen löscht.

1.3 Attribute für Vertex und Edge erstellen

Der Graph Editor soll dem User die Möglichkeit bieten, für jeden Vertex und jedes Edge eine Auswahl an vordefinierten Attributen zu erstellen und mit eigenen Werten zu füllen. Dabei soll der User einem Vertex die Attribute Name und String und einem Edge die Attribute Weight und String vergeben können. Das Attribut String soll als Platzhalter für Attribute dienen, die der User selber definieren kann.

1.4 Attribute des Graphen darstellen

Die Attribute, die der User für Vertices und Edges erstellen kann, sollen vom User ein- und ausgeblendet werden können. Werden die Attribute vom User eingeblendet, sollen sie direkt im Graphen neben den entsprechenden Vertices oder Edges platziert als Text angezeigt werden.

1.5 Aktionen rückgängig machen und wiederherstellen

Der User soll im Graphen Editor alle Aktionen, die er gemacht hat, rückgängig machen oder wiederherstellen können. Als Aktionen sind folgende Tätigkeiten definiert: Vertices oder Edges verschieben, Vertices oder Edges erstellen, Vertices oder Edges löschen, Attribute erstellen und die Werte von Attributen ändern.

2 Algorithmen Visualisierung

2.1 Gekennzeichnete Algorithmen finden

Der User soll die Möglichkeit haben, eigene Algorithmen im der GraphExamples Klasse zu programmieren und mit der vom Tool vorgegeben Annotation zu kennzeichnen. Damit soll der Algorithmus Editor dazu fähig sein, mit der vorgegeben Annotation gekennzeichnete Algorithmen zu finden und im Algorithmus Editor unter dem entsprechenden Namen als Auswahl für den User aufzulisten.

2.2 Algorithmen abspielen und stoppen

Der Algorithmus Editor soll es dem User erlauben, die aufgelisteten Algorithmen als Animation abzuspielen und die Animation zu kontrollieren. Startet der User die Animation, soll sie selbstständig in einem bestimmten Tempo ablaufen und die einzelnen Schritte, die der Algorithmus beim Graphen macht, direkt am Graphen darstellen. Der User soll die Möglichkeit haben, die Animation zu jedem Zeitpunkt, an dem sie läuft, zu stoppen.

2.3 Algorithmen schrittweise abspielen

Mit einer Auswahl an Tools soll der User die Animation pausieren und schrittweise in der Animation vorwärts oder rückwärts gehen können.

2.4 Abspieltempo einstellen

Der User soll das Tempo, mit dem ein Algorithmus automatisch animiert wird, individuell innerhalb vordefinierter Grenzen auswählen können.

2.5 Ergebnisse der Algorithmen an Graphen anwenden

Der User soll die Ergebnisse, die durch das Anwenden von einem Algorithmus an einem Graphen, gefunden werden direkt am Graphen anwenden oder verwerfen können. Dabei wird der Graph vom Tool so angepasst, dass sie direkt den Ergebnissen des Algorithmus entsprechen.

2.6 Graphen speichern und laden

Der User soll im Algorithmus Editor die Möglichkeit haben Graphen zu speichern oder zu laden. Das Speichern und das Laden sollen dabei so funktionieren, wie es bereits unter Punkt 1.2 definiert ist.

# Qualitätsanforderungen

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nr. | | Kurzbezeichnung | Status | P | V | K | R | Quelle | Datum | Ziele |
| 1 |  | Darstellung |  | | | | | | | |
|  | 1.1 | Einfache Bedienung des Interfaces zur Erstellung und Bearbeitung von Graphen | geplant |  |  |  |  | CS |  | T6 |
| 1.2 | Einfache Bedienung des Interfaces zum Abspielen von Algorithmen | geplant |  |  |  |  | CS |  | T6 |
| 2 |  | Performanz |  | | | | | | | |
|  | 2.1 | Erstellen und Bearbeiten von Graphen in angemessener Zeit | geplant |  |  |  |  | CS |  | T1 |
| 2.2 | Durchlaufen von Algorithmen in angemessener Zeit | geplant |  |  |  |  | CS |  | T2  T5 |

1 Darstellung

1.1 Einfache Bedienung des Interfaces zur Erstellung und Bearbeitung von Graphen

Der Graph Editor soll dem User ein möglichst simples Interface zur Erstellung und Bearbeitung bieten. Das Interface soll dabei dennoch leicht zu verstehen sein.

1.2 Einfache Bedienung des Interfaces zum Abspielen von Algorithmen

Der Algorithmus Editor soll dem User ein möglichst simples Interface zum Abspielen von Algorithmen zur Verfügung stellen. Dabei soll das Interface dennoch leicht zu verstehen sein.

2 Performanz

2.1 Erstellen und Bearbeiten von Graphen in angemessener Zeit

Der User soll im Graph Editor das Erstellen und Bearbeiten von Graphen in möglichst angemessener Zeit ausführen können. Die Zeit zwischen Aktion auslösen und Aktion darstellen soll dabei die Toleranzgrenze eines Users nicht überschreiten.

2.2 Durchlaufen von Algorithmen in angemessener Zeit

Der Algorithmus Editor soll es dem User erlauben, Algorithmen in angemessener Zeit abzuspielen und zu kontrollieren. Die Zeit zwischen Aktion auslösen und Aktion darstellen soll dabei die Toleranzgrenze eines Users nicht überschreiten.

# Randbedingungen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | | Kurzbezeichnung | Quelle | Datum |
| 1 |  | Java |  | |
|  | 1.1 | Umsetzung des Tools mit der Programmiersprache Java | CS |  |
| 1.2 | Umsetzung des Tools auf Basis der GraphExamples Library | CS |  |

# Glossar

|  |  |
| --- | --- |
| Algorithmus Editor | Der Algorithmus Editor ist der Teil des Tools, der es dem User erlaubt, Algorithmen auszuwählen und abzuspielen. |
| Annotation | Eine Annotation ist eine Java-Klasse und dient dazu, eine Anmerkung zu machen. |
| Attribut | Attribute sind Zahlenwerte oder Zeichenketten, die Vertices und Edges vergeben werden können. |
| Edge | Edges sind Bestandteil eines Graphen und verbinden Vertices miteinander. Sie werden graphisch als Linie oder als Pfeile repräsentiert. |
| Graph | Graphen sind Strukturen, aufgebaut aus Vertices und Edges. |
| Graphen Editor | Der Graphen Editor ist der Teil des Tools, der es dem User erlaubt, Graphen zu erstellen und zu bearbeiten. |
| Interface | Ein Interface ist eine graphische Oberfläche, die es dem User ermöglicht, verschiedene durch das Interface definierte Aktionen durchzuführen. |
| Tool | Als Tool wird die ganze Software zur Erstellung und Bearbeiten von Graphen sowie zum Animieren von Algorithmen definiert. |
| User | Als User des Tools sind Studenten und Dozenten im Informatikbereich gedacht. |
| Vertex, Vertices | Vertices sind Bestandteil eines Graphen und werden graphisch als Punkte repräsentiert. |