**Projekt 1: Visualisierugssoftware für Graphen und Graphenalgorithmen**

Notizen

Es soll - aufbauend auf einer vorgegebenen Graphen-Library - eine Software erstellt werden, welche Graphen darstellen kann, bzw. es erlaubt, Graphen mit einem graphischen Editor zu spezifizieren.

Gleichzeitig soll die Software der Visualisierung typischer Graphen-Algorithmen dienen. Ein Algorithmus, wie etwa derjenige von Dijkstra soll mit diesem Werkzeug so auf einfache Weise visualisierbar werden. Das Werkzeug soll sich als didaktisches Hilfsmittel bzw. als Debugging Tool für beliebige Graphen-Algorithmen eignen.

**17.2.2015**

Dienstag vormittag treffen

Pflichtenheft – alle Anforderungen

Kurze Beschreibung

(Ergänzende Information, Überlegungen festhalten)

Dozent nicht unbedingt Stakeholder

Dozent von Projekt überzeugen

Journal führen, ganz spartanisch

(17.2 Brainstorming)

Zuerst Domain Model

Gewisse Teile gut testen

Gewisse Teile im Prototypstadium

Code von aussen erlaubt

Angeben von wem Code stammt, auch wenn abgeändert

Repository erstellen (GitHub, Maven)

**17.2.2015**

Swing oder JavaFX?

Tutorials anschauen

Graphen speichern, öffnen

Objekte serialisieren/deserialisieren?

Graphen zeichnen

Wie Graphen zeichen? Wie Edges zeichnen?

Gerichtete und ungerichtete Graphen

Gewichtete und ungewichtete Graphen

Ganze Graphen verschieben

Graphen editieren, einzelne Vertex und Edges anwählbar

Vertex und Edges löschen?

Composite-Pattern?

Algorithmus visualisieren

Welche Algorithmen visualisieren?

Additional Attribute für Vertex, Edges setzen

Was alles visualisieren? Was alles sichtbar machen?

Wie verschiedene Attribute visualisieren? Vertex kann verschiedene Attribute je nach Algo haben

Alle Attribute anzeigen lassen pro Vertex

Algorithmus läuft von alleine ab, start/stop

Zusätzlich Optionen für zurück zum Start, schrittweise vorwärts/rückwärts gehen

State-Pattern?

Verschiedene Zustände für bevor/nach Algorithmus

Wenn Algo zum Beispiel einen minimal tree findet, dass man Graphen zum minimal tree abändern kann?

Topologisch nummerieren, wie visualisieren?

Undo/Redo im Sinne von Vorwärts/Rückwärts gehen

Command-Pattern

**26.2.2015**

GUI bekommt Graph als Objekt

Vertex als Kreis anordnen (etwas besseres finden?)

Methoden in Algorithmen einbauen

Schrittweise durch Algorithmus laufen

Pro Schritt soll ganzer Graph immer bekannt sein (also auch gefundene Edges durch Algorithmus), also wie eine Option geben, gefundener Graph anzuwenden oder zusätzlich als Kopie abspeichern

Transaktionen auch Graphen-weise speichern (so dass auch veränderte Graphen rückgängig gemacht werden können), nicht nur einzelne Schritte innerhalb eines Algorithmus

Algorithmus zuerst durchlaufen lassen und Ergebnis speichern

Erst danach anhand des Ergebnis Algorithmus schrittweise zeichnen

Customizable machen, Benutzer verschiedene Optionen geben wie er Graph/Algorithmus sichtbar gemacht werden muss

Benutzer erlauben Farben selber auszuwählen

Was ist sinnvoll darzustellen?

(Es muss nichts alles dynamisch sein, wäre natürlich vorteilhaft)

Gewichtete Kanten anzeigen

Gewicht der Kanten auch veränderbar machen

Gewicht der Kanten auch während des Algorithmus verändern, anpassen (zum Beispiel bei Algorithmus zuerst alle Kanten unendlich machen, dann je nach dem etwas setzen)

Beim Verschieben der Vertices auch das Verschieben sichtbar machen, während des Verschiebens zeichnen updaten

Attribute bei Vertices anzeigen (zumindest solche die als String darstellbar sind)

Zum Beispiel Attribut Distance bei Djikstra

Menüauswahl bei MouseOver für Vertices, Edges mit Editieren, Löschen, Umbenennen, ect

Algorithmen im GUI finden

Mit Annotations (dynamisch nachschauen ob es eine Klasse mit einer bestimmten Annotation gibt, Metasprache)

Annotationen können auch Parameter übergeben werden, die Methoden brauchen (Methode mit bestimmten Wert aufrufen)

Was soll das Programm alles können?

Wie setzen wir es um? (Swing, JavaFx)

**5.3.2015**

Zwei verschiedene anwählbare Tools für Vertex zeichnen, für Edges zeichnen

Vertex anwählbar, verschiebbar, löschbar

Menge von Vertices anwählbar, verschiebbar, löschbar

Graph als Ganzes anwählbar, verschiebbar, löschbar

Attribute einblenden, ausblenden

Graphen speichern, laden