

---

---

# [ Graphs-Visualization-Service ]

---

---

STUDIENARBEIT

ABTEILUNG INFORMATIK  
HOCHSCHULE FÜR TECHNIK RAPPERSWIL

AUTOREN  
[ MICHAEL WIELAND ]  
[ MURIÈLE TRENTINI ]

BETREUER  
[ THOMAS LETSCH ]  
DOZENT FÜR INFORMATIK AN DER  
HOCHSCHULE FÜR TECHNIK RAPPERSWIL

AUDITOR  
[ NAME ]

ZEITRAUM: 18.09.2017 - 22.12.2017

## Abstract

# Management Summary

Ausgangslage

Vorgehen, Technologien

Ergebnisse

Ausblick

# Technischer Bericht

Ausgangslage & Problembeschreibung

Lösungskonzept

Umsetzung

Ergebnisdiskussion mit Ausblick

## Danksagungen

Wir danken folgenden Personen für Ihre Unterstützung während der Studienarbeit:

- Thomas Letsch für die Betreuung unserer Studienarbeit.
- Jessica Martin für die technische Unterstützung beim Logo Design.

# Inhaltsverzeichnis

<b>1</b>	<b>[ Anforderungsanalyse ]</b>	<b>1</b>
1.1	[ Ausgangslage ] . . . . .	1
1.2	[ Mehrwert ] . . . . .	1
1.3	[ Aufgabenstellung ] . . . . .	1
1.4	[ User Stories ] . . . . .	1
1.5	[ Use Cases ] . . . . .	1
1.5.1	Brief . . . . .	1
1.5.2	Fully Dressed . . . . .	1
1.6	[ Domainanalyse ] . . . . .	1
<b>2</b>	<b>[ Realisierung ]</b>	<b>2</b>
2.1	[ Architektur ] . . . . .	2
2.2	[ UI Design ] . . . . .	2
2.2.1	Logo . . . . .	2
2.2.2	Konzept . . . . .	3
2.2.3	Icons . . . . .	3
2.2.4	Farben . . . . .	3

2.2.5	Wireframes . . . . .	3
2.3	[ Testing ] . . . . .	3
<b>3</b>	<b>[ Projektmanagement ]</b>	<b>4</b>
3.1	[ Projektverwaltung ] . . . . .	4
3.1.1	Iterationsplanung . . . . .	4
3.1.2	Meetings . . . . .	4
3.1.3	Schätzungen . . . . .	5
3.1.4	Zeitauswertung . . . . .	5
3.2	[ Artefakte ] . . . . .	6
3.3	[ Git Repositories ] . . . . .	6
3.4	[ Entwicklungsumgebung ] . . . . .	6
3.5	[ Continuous Integration ] . . . . .	6
3.6	[ Qualitätsmanagement ] . . . . .	6
3.6.1	Metriken . . . . .	6
3.6.2	Review . . . . .	6
3.7	[ Risikomanagement ] . . . . .	6
3.7.1	Backups . . . . .	6
<b>Anhang</b>		
<b>Glossary</b>		<b>8</b>
<b>Literatur</b>		<b>9</b>
<b>A Eigenständigkeitserklärung</b>		<b>12</b>
<b>B Vereinbarung</b>		<b>13</b>
<b>C Aufgabenstellung</b>		<b>14</b>

---

<b>D Zeitauswertung</b>	<b>17</b>
<b>E Testspezifikation</b>	<b>18</b>
<b>F Meeting Protokolle</b>	<b>19</b>



# Kapitel 1

## [ Anforderungsanalyse ]

### 1.1 [ Ausgangslage ]

### 1.2 [ Mehrwert ]

### 1.3 [ Aufgabenstellung ]

### 1.4 [ User Stories ]

### 1.5 [ Use Cases ]

#### 1.5.1 Brief

#### 1.5.2 Fully Dressed

### 1.6 [ Domainanalyse ]

# Kapitel 2

## [ Realisierung ]

### 2.1 [ Architektur ]

### 2.2 [ UI Design ]

#### 2.2.1 Logo

Das Logo wurde von den Eigenschaften des Kraken [5] inspiriert. Kraken sind bekannt dafür, dass sie viele Irrgarten-Probleme effizient lösen können. Dies ist eine Anspielung an die Algorithmen, die vom Graphs-Visualization-Service (GVS) unterstützt werden. Ebenfalls wurden die Saugnäpfe des Kraken als Graph visualisiert und auf der Stirn ist ein binärer Baum zu erkennen.



Abbildung 2.1 – Graphs-Visualization-Service Logo

### **2.2.2 Konzept**

### **2.2.3 Icons**

### **2.2.4 Farben**

### **2.2.5 Wireframes**

## **2.3 [ Testing ]**

# Kapitel 3

## [ Projektmanagement ]

### 3.1 [ Projektverwaltung ]

Als Projektmanagement Software wird Jira [4] eingesetzt. In Jira werden alle Requirements als Issues erfasst und in den Product Backlog eingepflegt. Pro Iteration sollen jeweils so viele Issues eingeplant werden, wie unter Berücksichtigung von administrativen Aufgaben abgearbeitet werden können. Dabei spielt der Teamspeed eine grosse Rolle, welcher sich über die Projektdauer einpendeln soll.

#### 3.1.1 Iterationsplanung

Die Iterationsplanung orientiert sich grob am Rational Unified Process (RUP) und ist unterteilt in eine Inception-, Elaboration-, Construction- sowie eine Transition-Phase. Eine Iteration wird nach SCRUM organisiert und deshalb auch Sprint genannt. Ein Sprint beginnt und endet jeweils am Freitag und dauert 2 Wochen.

#### 3.1.2 Meetings

Über die gesamte Projektdauer findet jeweils am Mittwoch um 17:15 ein wöchentliches Standortmeeting statt. Die Beschlüsse aus den Meetings werden protokolliert und bis spätestens 24h später an alle Teilnehmer versendet. Allfälliges Feedback wird nachträglich eingepflegt und versioniert abgelegt.

Phase	Beschreibung
Inception	Projektsetup (Jira, Workflow, IDE), Einarbeitung in die bestehenden Sourcen
Ellaboration 1	Anforderungsspezifikation, Domainmodell
Ellaboration 2	Architektur Modell, Guidelines für Quellcode und Tests, Wireframes
Construction 1	
Construction 2	
Construction 3	
Transition	

**Tabelle 3.1** – Iterationsplanung

### 3.1.3 Schätzungen

Issues werden auf Basis von Story Points geschätzt. Dieses Vorgehen hat sich mit SCRUM etabliert. Die Nutzung von Story Points führt dazu, dass nicht die individuell unterschiedliche Bearbeitungszeit, sondern die Komplexität einer UserStory geschätzt wird. Dies vereinfacht und homogenisiert die Schätzungen und hilft den Teamspeed zu bestimmen.[6, 7]

### 3.1.4 Zeitauswertung

Für die Zeitauswertung wird das Jira Plugin Tempo [1] verwendet. Dieses bietet umfassende Auswertungsmöglichkeiten, sowie Exports nach MS Excel. Die Auswertungen dieser Arbeit befinden sich im Anhang D.

## 3.2 [ Artefakte ]

## 3.3 [ Git Repositories ]

## 3.4 [ Entwicklungsumgebung ]

## 3.5 [ Continuous Integration ]

## 3.6 [ Qualitätsmanagement ]

### 3.6.1 Metriken

### 3.6.2 Review

## 3.7 [ Risikomanagement ]

### 3.7.1 Backups

Zur Minimierung von allfälligen Datenverlusten wird wie folgt vorgegangen:

1. Täglich automatisiertes Backup aller Projektdaten im JIRA [4] (Zeiterfassung, erstellte Issues, Jira Konfiguration)
2. Die Projektdokumentation sowie der Programmcode wird in den vier Github Repositories der Github Organisation [3] versioniert abgelegt. Für die Projektmitglieder gilt der Grundsatz "Commit early and often".
3. Weitere Artefakte werden auf Dropbox [2] abgelegt und dort automatisch gesichert und versioniert.

# Anhang

# Glossar

**GVS** Graphs-Visualization-Service: Titel des vorliegenden Produktes 2, 10

6, 10

7, 10

**RUP** Rational Unified Process: Vorgehensmodell in der Software Entwicklung, dass ursprünglich von IBM entwickelt wurde. 5, 10

**SCRUM** Vorgehensmodell zur agilen Softwareentwicklung 5, 10



# Literatur

- [1] *Atlassian Marketplace: Tempo Timesheets for Jira*. URL: <https://marketplace.atlassian.com/plugins/is.origo.jira.temp-plugin/cloud/overview>.
- [2] *Dropbox*. URL: <https://www.dropbox.com>.
- [3] *Github Repositories*. URL: <https://github.com/Graphs-Visualization-Service>.
- [4] *Jira*. URL: <https://project.redbackup.org/projects/GVS/>.
- [5] *Kraken*. URL: <https://de.wikipedia.org/wiki/Kraken>.
- [6] *Story Points verständlich erklärt*. URL: <http://www.ksimons.de/2011/06/story-points-verstandlich-erklart/>.
- [7] *The secrets behind story points and agile estimation*. URL: <https://www.atlassian.com/agile/estimation>.

# Abbildungsverzeichnis

2.1	GVS Logo . . . . .	2
-----	--------------------	---

# Tabellenverzeichnis

3.1	Iterationsplanung . . . . .	5
-----	-----------------------------	---

# Eigenständigkeitserklärung

Der/Die Verfasser/in erklärt hiermit, dass er/sie die vorliegende Arbeit selbständig, ohne fremde Hilfe und ohne Benutzung anderer als die angegebenen Hilfsmittel angefertigt hat. Die aus fremden Quellen (einschliesslich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind ausnahmslos als solche kenntlich gemacht. Durch Copyright geschützte Materialien wurden in dieser Arbeit nicht in unerlaubter Weise genutzt. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

.....

Ort, Datum

.....

Unterschrift des/der Verfassers/in

.....

Ort, Datum

.....

Unterschrift des/der Verfassers/in

## Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Studienarbeit GVS von Muriele Trentini und Michael Wieland unter der Betreuung von Thomas Letsch geregelt.

Die Urheberrechte stehen der Studentin / dem Student zu.

Die Ergebnisse der Arbeit dürfen sowohl von der Studentin / dem Student wie von der HSR nach Abschluss der Arbeit verwendet und weiter entwickelt werden

.....

Ort, Datum

.....

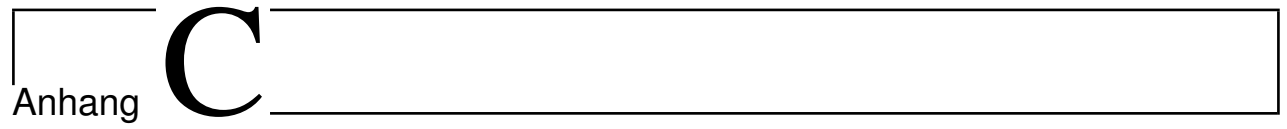
Unterschrift des/der Verfassers/in

.....

Ort, Datum

.....

Unterschrift des/der Verfassers/in



## Aufgabenstellung

Auf den nächsten Seiten befindet sich die Aufgabenstellung.

# ***Graphs-Visualization-Service GVS 2.0***

## **Studenten**

- Murièle Trentini
- Michael Wieland

## **Einführung**

Bei den Modulen "Algorithmen und Datenstrukturen 1" und "Algorithmen und Datenstrukturen 2" werden Algorithmen zur Bearbeitung von Bäumen und allgemeinen Graphen programmiert.

Dabei ist die Programmierung für die Visualisierung derartiger Algorithmen und den dazugehörigen Datenstrukturen mit relativ grossem Aufwand verbunden und nicht Bestandteil des Modul-Stoffes.

Aus diesem Grund wurde dazu im Jahre 2005 der 'Graphs-Visualization-Service' erstellt. Dieser Service ist für den Einsatz im Unterricht ausgelegt um die Studenten bei der Programmierung zu unterstützen.

Er besteht serverseitig aus einer Java-Applikation und clientseitig steht je ein API für Java und .NET zur Verfügung.

Mit einem Major-Upgrade soll das Tool nun auf die aktuellen Technologien aktualisiert werden.

## **Aufgabenstellung**

Beim aktuellen 'Graphs-Visualization-Service' soll unter Berücksichtigung resp. Beibehaltung der aktuellen Funktionalität:

- serverseitig Swing durch JavaFX ersetzt werden
- clientseitig das API um Generics erweitert werden
- punktuelle Verbesserungen vorgenommen werden

Unter Berücksichtigung von aktuellen Software-Engineering-Methoden soll ein geeigneter Entwicklungsprozess definiert und darauf basierend das Projekt realisiert werden.

## Technologien

- Java
- Enterprise Architect

## Generelles

- Die Vorgaben der Abteilung Informatik [1] sind einzuhalten, insbesondere die Anleitung zur Dokumentation [2].
- Die "Generelle Richtlinien für Studien- und Bachelorarbeiten" [3] sind einzuhalten.
- Mit dem CASE-Tool Enterprise Architect ist ein UML-Modell zu führen, welches synchron mit den Programm-Sourcen und der Projekt-Dokumentation ist.
- Ein Java-Entwickler muss mit der Projekt-Dokumentation in die Lage versetzt werden, die Applikation in Betrieb zu nehmen und weiter entwickeln zu können.

## Termine

- Montag, 18.09.17 Beginn der Studienarbeit
- Freitag, 22.12.17 17:00 Uhr Abgabe der Studienarbeit

## Betreuung

- Betreuer  
Thomas Letsch  
tlletsch@hsr.ch  
055 - 22 24 567 (HSR Büro 5.204); 055 - 214 43 50 (Geschäft)
- Besprechungen  
Wöchentliche Besprechung jeweils Mittwoch 17:15 Uhr

## Referenzen

- [1] Skripte-Server: /Informatik/Fachbereich/Studienarbeit\_Informatik/SAI
- [2] Anleitung Dokumentation BA\_SA\_170905.pdf
- [3] "Generelle Richtlinien für Studien- und Bachelorarbeiten"  
(v1.7 / 19.08.2015, Thomas Letsch)

Rapperswil, 18.September 2017



Thomas Letsch



Anhang **D**

Zeitauswertung

# Anhang **E**

## Testspezifikation

Anhang **F**

Meeting Protokolle