Gaida, Lässer, Schwendinger

Lerntagebuch

LV: Softwareprozesse und Softwarequalität

University of Applied Sciences: Vorarlberg

Department of Computer Science
Dipl. Inform. Bernd G. Wenzel

Dornbirn, 2014

Abstract

Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Inhaltsverzeichnis

Listings

Abbildungsverzeichnis

Todo list

1 LV 1 am 21.02.2014

1.1 Organisatorisches

Zu Beginn dieser Vorlesung wurden zuerst die organisatorischen Randbedingungen für diese Lehrveranstaltung geklärt.
Dabei wurde entschieden, dass anstelle einer Prüfung ein Lerntagebuch in Gruppen zu max. drei Personen erstellt wird. Das Lerntagebuch enthält zusätzlich ein Glossar mit den wichtigsten Begriffsdefinitionen rundum das Thema Softwareprozesse und Softwarequalität.

1.2 Motivation des Software-Engineering

Als Einstieg in das Thema dienten ein paar nicht ganz ernst gemeinte "Gesetze". Diese Gesetze sollten uns verdeutlichen, dass Software-Engineering nicht so trivial und einfach ist, wie es oft den Anschein hat. Wie schon die Folgerung von Kleinbrunner besagt

> "Wenn eine Programmieraufgabe leicht aussieht, ist sie schwer."

Es wurden unter anderem die "Gesetze"von Gutterson, Farvour, Brunk, Munbright und vielen anderen vorgestellt.

Um die Bedeutung von qualitativ hochwertiger Software zu verdeutlichen, stellte der Vortragende einige Probleme in bekannten Softwareprojekten wie beispielsweise der Verlust einer Segelyacht mit Crew vor der mexikanischen Pazifikküste aufgrund eines Softwarequalitätsfehlers.

Nach dieser Einführung wurden wichtige Begriffe wie Software, Software-Engineering, Softwareprozess usw. definiert. Die wichtigsten Definitionen, auch von zukünftigen Vorträgen, sind im Glossar angeführt. Dabei wurde auch das Vorgehensmodell und die Methode des Software-Engineerings vorgestellt.

Bei der Methode des Software-Engineerings geht es um einen strukturierten Ansatz für die Software-Entwicklung mit Ziel eine möglichst hohe Qualität der Software bei gleichzitig

geringen Entwicklungskosten zu erreichen.

Das Vorgehensmodell definiert wie der Softwareentwicklungsprozess abläuft. Bekannte Modelle sind dabei Scrum, Wasserfall oder das V-Modell. Dabei stellt sich heute oft die Frage, ob man sich für einen klassisches oder agiles Vorgehensmodell entscheidet. Der größte Unterschied zwischen agil und klassisch ist, dass beim Agilen Ansatz der Kunden viel mehr involviert ist und öfters das Feedback von ihm in die Entwicklung mit einfließt. Auch sind die Iterationen beim Agilen Vorgehen meist kürzer.

Anschließend behandelten wir das Thema der soziotechnischen Systeme. Dabei wurde der Begriff genauestens definiert, die wesentlichen Eigenschaften und der Lebenszyklus sowie die Umfeldfaktoren vorgestellt.

Nach den soziotechnischen Systemen behandelten wir das Thema der kritischen Systeme. Dabei geht es um Systeme dessen Ausfall oder Fehlfunktion sich auf die Umgebung in Form von z.B. wirtschaftlichen Verlusten, Schäden an der Umwelt auswirken. Wichtige Faktoren dabei sind die Verfügbarkeit, Zuverlässigkeit, Betriebssicherheit und Systemsicherheit. Der Sammelbegriff für diese Punkte heißt Verlässlichkeit.

Zum Schluss der Lehrveranstaltung wurden die ethnischen Herausforderungen beim Software-Engineering behandelt. Dabei geht es um Vertraulichkeit gegenüber dem Arbeitgebern und Kunden, um den Schutz von geistigem Eigentum und den Schutz vor Computermissbrauch. Es gibt auch einen Knigge für das professionelle Verhalten des Software-Engineering definiert durch ACM/IEEE.

Abschließend erhielten wir die Aufgabe bis zur nächsten Vorlesung ein Programm zu entwickeln zur Berechnung der reellen Lösungen einer quadratischen Gleichung. Ziel bei dieser Aufgabe ist es auf die Qualität des Codes zu achten.

2 Glossar

- Software
- Software-Engineering
- Informatik
- Softwareprozess
- Software-Validierung (o Ob die Software für den Kunden einen Wert hat)
- Software-Verifizierung (o Übereinstimmung der Software mit den Spezifikationen)
- Qualität
- Technisches computer-basiertes System
- Soziotechnisches System
- Funktionale Anforderungen
- Nicht-Funktionale Anforderungen

3 Zusammenfassung

Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.