

Einführung in die Softwaretechnik 0

Übersichtsvortrag für das 1. Sem. SWT

Jochen Ludewig, ISTE, Abt. Software Engineering

Inhalt

- Software-Qualität
- Erste Übersicht zur Softwaretechnik, Vorschau EST
- Der aktuelle Studienplan
- Ratschläge für werdende Softwaretechnikerinnen
- Wer ist zuständig?

Software-Qualität

Schlechte Software haben fast alle Menschen erlebt (erlitten).

Gute Software sieht man dann und wann auch.

Wie kommt sie zu Stande? Welche **Hindernisse** gibt es? Welche **Verfahren und Werkzeuge** sind nützlich? Wie ist die Arbeit zu **organisieren**? Welche **Techniken** gibt es? Wie können wir abschätzen, welcher **Aufwand** und wieviel **Zeit** nötig sind? Wie können wir unsere Ergebnisse **prüfen**?

Das sind die Fragen, um die es in Ihrem Studium geht.

Wenn Sie (und wir) erfolgreich sind, werden Sie nach dem Abschluss nicht nur ein Stück Papier besitzen, sondern auch eine Qualifikation, die recht selten und darum wertvoll ist: Sie haben ein **tiefes Verständnis** dafür entwickelt, was **gute Software** ist und wie man sie macht. Das ist **die Idee des Studiengangs Softwaretechnik**.

Erste Übersicht zur Softwaretechnik

Thema der Vorlesung EST I: Ein Gang durch das Software-Projekt

Software (Merkmale)	Softw. Engineering (Geschichte)
Planung	Analyse
Begriffslexikon	Spezifikation
Kostenschätzung: COCOMO, FPs	CASE
Qualitätssicherung	Reviews
Lokalität, Information Hiding	Kopplung und Zusammenhalt
Entwurf	Handbuch
Metriken	Implementierung und Test

Anschließend: PE, SoPra, SEfST, Studienprojekt, Fachstudie

Software, Geschichte des Software Engineerings

Software umfaßt nach IEEE Std 610.12 *Programme, Abläufe, Regeln, auch Dokumentation und Daten, die mit dem Betrieb eines Rechnersystems zu tun haben.*

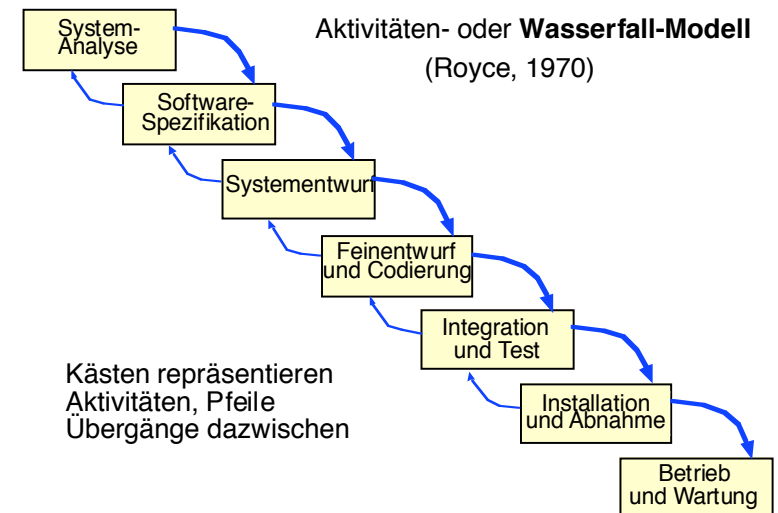
Software gehört zu den technischen Produkten, hat aber einige sehr spezielle Eigenschaften und Merkmale.

F.L. Bauer, NATO-Konferenz in Garmisch, 1968:

*The whole trouble comes from the fact that there is so much tinkering with software. It is not made in a clean fabrication process, which it should be. What we need, is **software engineering**.*

Software Engineering (= Softwaretechnik) strebt nach dem **globalen Kosten-Minimum** (oder nach dem globalen Nutzen-Maximum).

Planung, Life Cycle, Software-Prozess



Der Software-Prozess ist das **Schema**, nach dem Projekte ablaufen.

Qualitätssicherung, Reviews

Qualität wird uns vor allem dann bewusst, wenn es daran **mangelt**. **Ungenügende Software-Qualität** kostet viel Zeit, Geld, Nerven, immer öfter auch Leben und Gesundheit der Menschen.

Darum ist Software Engineering untrennbar mit der **Software-Qualitätssicherung** verbunden.

Hohe Qualität lässt sich nur erzielen, indem

- sehr kompetent und sorgfältig entwickelt wird (**konstruktive Qualitätssicherung**) und
- Fehler, die trotzdem entstehen, hartnäckig aufgespürt und beseitigt werden.

Für die Fehlersuche ist die Inspektion, das **Review**, die wichtigste Technik. In EST werden Reviews unterrichtet und geübt.

Analyse, Begriffslexikon, Spezifikation

Die **Analyse** dient dazu,

- den **Ist-Zustand** beim Kunden zu erfassen und
- die **Anforderungen** des Kunden zu ermitteln.

Im **Begriffslexikon** werden die für das Projekt wesentlichen Begriffe, also Wörter und Bedeutungen, dokumentiert.

Die (Anforderungs-) **Spezifikation** dokumentiert die bereinigten und geprüften **Resultate der Analyse**. Die Spezifikation wird als **Vorgabe** für fast alle weiteren Tätigkeiten der Software-Entwicklung benötigt (Entwurf, Realisierung, Tests und andere Prüfungen, Benutzerhandbuch, ...) und bildet darum das **zentrale Dokument** der Software-Entwicklung.

Kostenschätzung

Die Kostenschätzung dient nicht nur der Kosten-Prognose, sondern der Planung und Organisation allgemein. **COCOMO** wird als ein bekanntes Verfahren der Kostenschätzung vorgestellt.

Entwurf, CASE

Für den **Software-Entwurf** gibt es (bis jetzt) keine wirklich griffigen Methoden. Wenigstens gibt es aber einige **Grundregeln** und **Kriterien**, nach denen Entwürfe beurteilt werden können.

CASE (Computer Aided Software Engineering) steht für die Idee, in der Software-Entwicklung Werkzeuge, sogenannte **CASE-Tools**, einzusetzen, die die **Produktivität** erhöhen und die **Qualität** verbessern sollen. Im SoPra werden Sie ein CASE-Tool einsetzen.

Handbuch

Das Handbuch, auf Papier oder elektronisch, ist eine wichtige Komponente der Software. Seine Herstellung gehört darum zum Software Engineering.

Metriken

Als Metriken bezeichnet man Verfahren zur Erhebung von Daten (**Zahlen**) über den Verlauf des **Projekts** (z.B. die Aufwände der Entwickler) und über sein **Resultat** (z.B. den Umfang in Lines of Code oder die Zahl der entdeckten Fehler). Metriken bieten die Möglichkeit, die **Wirkung von Änderungen** zu erkennen.

Configuration Management, Change Management

Jedes komplexe Produkt, das aus einer großen Zahl von Komponenten besteht, bedarf einer sorgfältigen **Verwaltung der Teile und des Ganzen**. Das gilt um so mehr für die immaterielle Software.

Die **Organisation und Dokumentation aller Änderungen** (Change Management) ist ein spezieller Aspekt des Software Configuration Managements.

Implementierung und Test

Die **Codierung** wird in der Softwaretechnik nur gestreift, weil sie in anderen Lehrveranstaltungen vermittelt wird (vgl. PE).

Der **systematische Test** ist dagegen ein weiterer Schwerpunkt der Vorlesung.

Der Studienplan SWT

Studienplan und Prüfungsordnung sind brandneu und sicher noch nicht ausgereift. Bitte rechnen Sie damit, dass es dann und wann irgendwo klemmt oder knackt.

Trost: Auch in Studiengängen, die seit vielen Jahren angeboten werden, gibt es immer wieder Probleme. Das Leben ist eine Baustelle, und die Universität ist pralles Leben.

Sie können aber auch damit rechnen, dass wir, die Anbieter der Studiengänge, uns in der Umstellung noch mehr als sonst bemühen werden, **Schwierigkeiten und Mängel** rasch zu beseitigen.

Bachelor Softwaretechnik

Sem.	Theoret. Informatik, div. Module	Mathematik und Wahlfächer	Programmierung, Software Engineering, Projekte		
1	3V 1Ü Logik und disk. Strukt.	4V 2Ü Mathematik für Informatiker (I)	4V 2Ü Programmieren und SW-Entwickl.	2 Ü Prgr.-kurs	2 Ü Softw.-Qual.
2	3V 1Ü Automaten & form. Spr.	4V 2Ü Mathematik für Informatiker (II)	4V 2Ü Datenstrukturen und Algorithmen	3V 1Ü Einf. in die Softw.techn.	
3	3V 1Ü Algorith. und Berechn.	3V 1Ü Wahlfach (Kat. SWT)	3V 1Ü Technische Informatik	4 P Software-Praktikum	3V 1Ü Programm-entwickl.
4	3V 1Ü Wahlfach (Kat. SWT)	3V 1Ü Wahlfach (Kat. SWT)	16 SWS (3 V, 1 Ü, 2 S, 10 P)		3V 1Ü Software Engineering
5	3V 1Ü Wahlfach (Kat. ISG)	3V 1Ü Wahlfach (Kat. SWT)	Studienprojekt inkl. Vorlesungen und Seminar		3V 1Ü Sichere und zuverl. SW
6	2 Ü Englisch	2 H Allg. Qual.	3V 1Ü Wahlfach (Kat. ISW)	4 H Fachstudie	
			8 H Bachelor-Arbeit		

Stützkurs

Bis vor einem Jahr habe ich für die Studienanfänger (SWT) den so genannten **Stützkurs** angeboten.

Er sollte all jenen, die sich im ersten Semester unsicher fühlen, die Möglichkeit geben, **beliebige Fragen** zu stellen und ihr **Hintergrundwissen** über Informatik zu ergänzen.

Also: Kein zusätzlicher Stoff, kein Lehrplan, keine Prüfung!

Wegen der unübersichtlichen Gesamtsituation durch die Umstellung auf Bachelor-Studiengänge habe ich den Stützkurs dieses Jahr **nicht** eingeplant.

Ich bin aber gern bereit, den Stützkurs spontan ins Leben zu rufen, **wenn es dafür Bedarf gibt** (und ich davon erfahre).

Keine Angst!

Der Bachelor-Studiengang ist so konzipiert, dass er in **sechs Semestern** absolviert werden kann. (Nicht muss!)

Es hat große Vorteile, **zielstrebig und effizient** zu studieren. Es hat aber auch große Vorteile, zwischendurch einmal **tief Luft zu holen** und um sich **herumzublicken**.

Wenn Sie nicht unter großem Druck stehen, nutzen Sie die Chance, **langsamer** zu studieren.

Die **Halbzeit** (nach dem 3. Semester) ist ein geeigneter Moment, um für eine Weile auszusteigen.

Machen Sie ein **Praktikum in der Industrie**, studieren oder arbeiten Sie im **Ausland**! Es gibt mehr Möglichkeiten als Nachfrage.

Achtung: Dies ist **keine Empfehlung zum Trödeln!**

Ratschläge für werdende Softwaretechnikerinnen

- Bereiten Sie Lehrveranstaltungen vor und nach, lassen Sie **Defizite** nicht anwachsen. Behalten Sie im Studium die **Initiative!**
- Sammeln Sie **bewusst** Ihre **Erfahrungen**.
Alles, was man *an* und *mit* Software erlebt, ist **SE-Erfahrung**.

Achten Sie dabei auch auf Probleme, die in der **Kommunikation** und in der **Organisation** liegen.

Beobachten und dokumentieren Sie, wenn Sie die Chance haben, **Software-Projekte** (auch Ihre eigenen) und versuchen Sie, die Probleme zu erkennen und zu verstehen. Ihre **Praktika** (die nicht formal als Praktika organisiert sein müssen) sollten in dieser Hinsicht besonders ergiebig sein.

- Nutzen Sie die **Informatik-Bibliothek**, schauen Sie sich die **Fachzeitschriften** und **Tagungsbände** an.
- Werden Sie Mitglied in einem **Berufsverband** (GI, IEEE-CS, ACM, ...)
- Besuchen Sie **Tagungen** und die vielen **Vorträge** hier im Hause, auch die **Kolloquien** der Abteilungen, Präsentationen zum Abschluss der Studienprojekte
- Schauen Sie die **Web-Seiten des SEI** an (**Links in Abt. SE**), befassen Sie sich mit dem **PSP (Personal Software Process)** und experimentieren Sie selbst damit!
- Begreifen Sie die besondere **Schwierigkeit** der Softwaretechnik, die in ihrer **Einfachheit** liegt, und nehmen Sie sie ernst!
- Lernen Sie **sprechen!** Lernen Sie **schreiben!** Sie werden beides brauchen.

Wer ist zuständig?

Keiner in der Fakultät ist **für alles** zuständig!

Dekan: Prof. Plödereder

Studiendekan: Prof. Diekert

Alle Fragen rund um die Prüfungsordnung:

Studienprojekte:

Industrie-Praktika und Fragen,
für die niemand zuständig scheint: Prof. Ludewig

Semestersprecher?

Fragen, Wünsche, Anregungen, ...