Softwareentwicklung und **das Projekt**

Lars Wechsler

12. April 2023

Inhaltsverzeichnis

1 Projektorganisation

1.1 Grundlagen der Softwaretechnik

In diesem Kapitel soll es nicht mehr um Projekte gehen, die jeder Einzelne alleine angehen kann, sondern um das „Programmieren im Großen“ - sei es in einer (internationalen) Firma oder „privat“ in einem Verbund mit anderen Programmierys.
Gerade in großen Projekten ist eine qualitativ hochwertig Zusammenarbeit unerlässlich. Leider gibt es hier immer wieder große Reibungsverluste, neben den fachlichen Inhalten ist die Fähigkeit des „Teamwork“ - gerade heutzutage - ein immens wichtiger „soft skill“.
In der Informatik gibt es verschiedene Modelle, die die Projektorganisation strukturieren und z.T. veranschaulichen sollen (dazu gibt es ganze eigene Vorlesungen im Bereich der **Softwaretechnik**!).
Im Allgemeinen geht man grob von den folgenden Phasen im Leben eines Softwareprojekts aus:

1. Anforderungen und Spezifikationen
2. Planung
3. Entwurf und Design
4. Implementierung und Integration
5. Betrieb und Wartung
6. Stilllegung

Zur Beschreibung dieser Phasen, deren Interaktion und zeitlicher Abfolge gibt es verschiedenste Vorgehensodelle, von denen in folgenden Kapiteln einige vorgestellt werden sollen.
Zunächst aber noch Details zu den einzelnen Phasen

1.2 Der Lebenszyklus von Software

Anforderungen und Spezifikationen
Grundlegende Fragen, die eine Anforderungsanalyse zu Beginn eines Projektes beantworten sollte, sind z.B.:

- welches Problem soll konkret gelöst werden?
- welche Leistung soll das geplante Projekt erbringen?
- welche Personen müssen mit einbezogen werden?
- gibt es sich widersprechende Anforderungen verschiedener Personen/Gruppen?

Insbesondere die letzten beiden Punkte sollen verdeutlichen, dass gerade aus Kundensicht nicht immer Einigkeit über den Funktionsumfang oder die genauen Spezifikationen eines Programs herrscht. Eine genaue Spezifikation der zu erbringenden Leistung ist deshalb wichtig, um Unzufriedenheit diesbezüglich vorzubeugen.
Die Anforderungsanalyse beschränkt sich aber nicht nur auf konkrete Funktionalität, sondern bildet ein ganzes Spektrum an Faktoren ab:

- **Funktionale Anforderungen**: das was man zunächst erwarten würde - welche Funktion soll das System haben, wie soll es sich verhalten, etc.
- **Nichtfunktionale Anforderungen**: hier geht es eher um den eigentlichen Betrieb der Software, wie leistungsfähig soll Sie sein - d.h. wie groß sind die Anforderungen an Speicher/Prozessor im laufenden Betrieb, ist die Software skalierbar, etc.
- **Designbedingungen**: gibt es bereits weitere technische Bedingungen, das könnte z.B. bereits existierende Software sein, zu der Schnittstellen vorhanden bzw. eine Kompatibilität hergestellt werden muss.
- **Prozessbedingungen**: dies sind eher interne Anforderungen der Entwickler - wie viele Personen sind notwendig, wie soll die Vorgehensweise bei der Entwicklung sein (zwar intern zu sehen, aber insbesondere auch der Kontakt mit dem Kunden - Ablieferung eines „fertigen“ Produkts oder Zwischenstände z.B.).

Übliche Verfahren, die zur Ermittlung der Anforderungen verwendet werden sind z.B. Fragebögen, Interviews, Simulationsmodelle, Workshops, etc.
Die Ergebnisse müssen natürlich verschriftlicht werden, der Auftraggeber fasst alles im sogenannten **Lastenheft** zusammen, das möglichst konkret die Gesamtheit aller Anforderungen beschreibt - kurz gesagt: **Was soll erstellt werden?**.
Das **Pflichtenheft** dagegen beschreibt - ebenfalls möglichst konkret - wie der Auftragnehmer die Anforderungen des Auftraggebers lösen möchte, kurz: **Wie und womit wird umgesetzt**. Erst nach Akzeptanz des Pflichtenhefts beginnt die eigentliche Umsetzung.
Hinweis: Diese Beschreibung entspricht in der Realität natürlich nicht immer den Gegebenheiten, sondern eher eine etwas akademische Sicht auf die Dinge. Gerade Software-Entwicklung ist ein sehr dynamischer Prozess und lässt sich selten gut in statische Kategorien verpacken. Die Schule kann hier kein sicheres aktuelles Bild vermitteln, da Regeln und Abläufe ggf. in den Betrieben bereits überholt sind ⇒ Praktika deutlich nützlicher, um zu lernen „wie der Hase läuft“.

2 Nebenläufige Prozesse

3 Entwurfsmuster - MVC

4 Das Projekt - Anforderungen