# Analyse u. Design: Phasen der Softwareentwicklung

Wir betrachten Software "im Großen", im kommerziellen, professionellen Umfeld.

Ein Auftraggeber schreibt einen Auftrag aus,

worauf sich Firmen mit einem Kostenvoranschlag bewerben können.

Die Firma, die den Zuschlag bekommt,

trifft sich mit dem Auftraggeber oder dessen Vertretern.

Softwarefirmen arbeiten im Team mit einem Projektleiter.

In der Softwareentwicklung ist Teamfähigkeit deshalb eine zunehmend wichtige Voraussetzung.

In der ersten Phase, der Analyse,

trifft sich ein speziell dafür geschulter Mitarbeiter mit dem Kunden.

Der Kunde legt seine Wünsche dar,

die die Software erfüllen soll.

I.d.R. ist der Kunde kein IT-Spezialist,

was Probleme mit sich bringt:

* Kunde kann die Forderungen nicht fachgerecht formulieren.
* Kunde stellt unrealistische Forderungen.
* Beschreibungen sind auf unterschiedlichem Abstraktionsniveau.

In der Analysephase wird von technischen Randbedingungen abstrahiert,

wie der Frage der zu verwendenden Programmiersprache,

der Hardware, der Typ der Datenbank o.ä.

Pflichtenheft:

Es wird ein "Pflichtenheft" erstell,

indem genau spezifiziert wird,

was die Software leisten soll sowie Preis, Fertigstellungsdatum,

Wartungsarbeiten u. genehmigte Ausfallzeiten u.v.m.

Die genaue Spezifikation ist juristisch von Bedeutung,

da es oft um große Beträge geht

u. Betrugsversuche von beiden Seiten denkbar sind:

* Der Kunde könnte, nachdem die Software wie vereinbart geliefert wurde, behaupten,  
  dass ein größerer Leistungsumfang vereinbart war  
  u. mit dieser Begründung die Zahlung verweigern.
* Die Softwarefirma könnte eine Software mit nur einem Teil des vereinbarten Funktionsumfangs abliefern u. behaupten,  
  mehr sei nicht vereinbart gewesen.

Der Prototyp

Ein Prototyp in der Softwaretechnik ist die grafische Oberfläche,

wie sie der Kunde bedienen wird,

ohne die dahinter implementierte Funktionalität.

Warum wird ein solcher erstellt?

Um dem Kunden in einer frühen Phase der Softwareentwicklung

die Möglichkeiten zu geben, Änderungswünsche zu äußern.

Würde dies erst geschehen,

wenn die Implementierung schon vollständig abgeschlossen ist,

wäre der Änderungsaufwand viel größer.

Designphase:

In der Designphase werden die Informationen des Analytikers aus der Analysephase

in ein Modell umgesetzt, z.B. in UML.

Dabei wird schon recht genau spezifiziert,

wie die Klassen aussehen, welche Attribute sie haben,

wie das Zusammenspiel der Klassen sein soll u.a.

Direkt programmiert wird jedoch noch nicht.

Vergleich:

Ein "Häuslebauer" hat, nachdem er lange gespart hat, ein Grundstück erworben

u. kann es kaum abwarten, dort sein Haus zu bauen.

Emotional nachvollziehbar ist es,

dass er möglichst schnell anfangen möchte

um konkrete Ergebnisse zu sehen.

Analytisch betrachtet ist die Vorgehensweise jedoch falsch:

Bevor der erste Stein in die Hand genommen wird,

müssen sehr detailliert sämtliche Aspekte geplant werden

wie Führung von Leitungskanälen, Wasser- u. Gasleitungen,

Größe u. Position von Fenstern u. Türen u.v.m.

Muss auf einem Papierplan eine Änderung vorgenommen werden,

ist dies mit Bleistift u. Radiergummi mit geringem Aufwand möglich.

Wären hingegen schon Baumaßnamen gestartet,

wäre der Aufwand groß,

bereits gebaute Mauer wieder einreißen zu müssen.

Der Architekt soll vor Baubeginn sehr genau alle Aspekte berücksichtigen

u. erst danach zu bauen anfangen.

Zurück zur Software:

Der (naive) Kunde würde sich wünschen,

möglichst schnell Quelltext auf dem Papier zu sehen.

Dies ist aus Aspekten der Softwaretechnik jedoch kontraproduktiv.

Wie beim "Häuslebau" wäre die Wahrscheinlichkeit groß,

dass nachträglich unter großem Aufwand prinzipielle Strukturen geändert werden müssen,

weil sich nicht zu Beginn hinreichend durchdacht waren.

Dies würde die Kosten für das Projekt in die Höhe treiben.

Deshalb hier wie beim Haus:

Erst in der Designphase sehr genau alles überdenken u. genau spezifizieren,

bevor die erste Zeile Quelltext geschrieben wird.

Ein weiterer Aspekt ist der,

dass, wenn die Designphase sehr gründlich durchlaufen wurde,

die Modelle bereits so genau sind,

dass halbautomatisch der Quelltext daraus schon maschinell generiert werden kann.

Der Programmierer arbeitet deshalb mehr im Modell,

was seinem Denken näher kommt als das Erstellen langer Quelltexte.

So wird seine Arbeit "auf ein höheres Niveau gehoben"

mit den Vorteilen, dass sie leichter, schneller u. mit geringerer Fehlerquote

verlaufen kann.

Implementierungsphase:

Erst zum Schluss wird konkreter Quelltext erzeugt,

in der Zwischenzeit oftmals maschinell von Tools aus dem Modell generiert.

Die Phase verliert gegenüber den beiden vorherigen dadurch an Bedeutung.