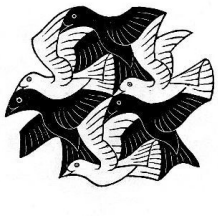
ZUSAMMENFASSUNG

PARADIGMEN DER SOFTWAREENTWICKLUNG

SOMMERSEMESTER 2025



Niklas Conrad
Fakultät Informatik
Bachelor of Science Inforamtik
April 2025

NIKLAS CONRAD

Paradigmen der Softwareentwicklung

FAKULTÄT INFORMATIK:
https://www.hs-schmalkalden.de/hochschule/fakultaeten/fakultaet-informatik

INHALTSVERZEICHNIS

1 EINFÜHRUNG

1.1 ABSTRAKTION

ABSTRAKTION ist das gezielte Unterdrücken oder Verbergen bestimmter Details eines Prozesses oder Artefakts, um andere Aspekte, Strukturen oder Eigenschaften klarer hervorzuheben.
Eigenschaften einer Abstraktion sind Prinzipien des:

- Verbergens oder Weglassens – ***Komplexitätsreduktion***
- Generalisierens – ***Wiederverwendbarkeit***
- Konzepts - Ïdee versus Realität- ***Flexibilität***

BEISPIELE

1. Die Landkarte ist nicht das Gelände
   - Eine Karte vereinfacht die Realität: Anstatt jeden Baum, jeden Stein oder jede Blume darzustellen, zeigt sie nur die wesentlichen Informationen, die man braucht.
   - Karten folgen allgemeinen Konventionen, zum Beispiel Linien mit Höhenangaben zur Darstellung der Topografie.
   - Eine Karte ist eine abstrakte Repräsentation eines Landabschnitts – ein Modell, das sich von Karte zu Karte unterscheiden kann.
2. Waschmaschine
   - „Die Waschmaschine starten“ ist ein abstrakter Vorgang. Die eigentliche Realität sind die vielen mechanischen Prozesse im Inneren der Maschine.
   - Es ist wesentlich einfacher, einen Knopf zu drücken, als manuell festzulegen, wann welche Ventile öffnen oder schließen oder wie sich die Trommel bewegen soll.
   - Auch wenn sich Waschmaschinen in ihrer Funktionsweise leicht unterscheiden (z.B. im Hinblick auf Energieeffizienz), gibt es fast immer die abstrahierte, vereinheitlichte Funktion „Start“. Diese ist eine Generalisierung über verschiedene Modelle und Marken hinweg.

Eine Anwendung ist eine Abstraktion einer Abstraktion einer Abstraktion...

- Benutzeroberfläche
- Hochsprache (z.B. JavaScript)
- Low-Level-Sprache (z.B. C)
- Maschinensprache
- Rechnerarchitektur (Register, Speicher, Recheneinheit usw.)
- Schaltungselemente (Logikgatter)
- Transistoren
- Festlörperphysik
- ...

1.2 ABSTRAKTIONSEBENEN

Abstraktionsebenen beziehen sich auf die verschiedenen Schichten oder Stufen, auf denen ein Computersystem betrachtet oder entwickelt werden kann. Jede Ebene verbirgt die Komplexität der darunterliegenden und bietet eine einfachere Schnittstelle für die darüberliegende. Damit wir, als Menschen, komplexe Softwaresysteme entwickeln können müssen wir Probleme analysieren, verstehen und kommunizieren können. Abstraktionen bieten die möglichkeit Probleme zu strukturieren.

Programmiersprachen bieten eine Reihe von Abstraktionsebenen an:

KONZEPTE beschreiben eine grundlegende Idee oder ein gedankliches Modell, das Möglichkeiten aufzeigt, wie Abstraktionen erstellt oder strukturiert werden können.
Konzepte beantworten also die Frage:

> *„**Welche** grundlegenden Möglichkeiten zur Abstraktion oder Strukturierung gibt es?“*

Typische Konzepte in der Informatik sind: *Objekt, Klasse, Schnittstelle, Vererbung, Typ, Nachricht*

PARADIGMEN definieren ein umfassendes Denkmodell oder eine allgemeine Sichtweise, wie Software strukturiert wird, um Ziele der Wartbarkeit, Erweiterbarkeit, Flexibilität, Robustheit, etc zu erreichen. Sie beantworten die Frage:

> *„**Wie** kombiniere ich Konzepte und Abstraktionen zu einem stimmigen, einheitlichen Gesamtbild?“*

Typische Programmierparadigmen sind u.a.:

OBJEKTORIENTIERTES PARADIGMA (OOP) Kombination von Konzepten wie Objekten, Klassen, Nachrichten, Kapselung, Vererbung, Polymorphie, Late Binding, um komplexe Software modular, verständlich und wartbar zu gestalten.

FUNKTIONALES PARADIGMA Kombination von Konzepten wie reinen Funktionen, Zustandslosigkeit, Immutabilität, höherwertigen Funktionen, um fehlerarme und gut testbare Software zu erzeugen.

PROZEDURALES PARADIGMA Kombination von Konzepten wie Prozeduren, Modulen, und strukturierten Anweisungen, um eine schrittweise Abfolge von Operationen klar darzustellen.

DEKLARATIVES PARADIGMA Kombination von Abstraktionen, um zu beschreiben, was getan werden soll, anstatt explizit anzugeben, wie es umgesetzt wird (z.B. SQL oder logische Programmierung).

PRINZIPIEN beschreiben eher eine allgemeine Richtlinie oder eine grundlegende Regel, nach der die Konzepte praktisch angewendet werden. Prinzipien geben Orientierung für konkrete Entscheidungen, sie beantworten also die Frage:

> *„**Wie** setze ich Konzepte und Paradigmen sinnvoll ein, unter welchen Voraussetzungen, und warum?“*

Typische Prinzipien in der Informatik sind:

- Information Hiding (Informationsversteckung)
- Separation of Concerns (Trennung der Belange)
- Single Responsibility Principle (Einzelverantwortung)
- DRY (Don’t Repeat Yourself)
- Open/Closed Principle (OCP)

2 GRUNDLAGEN VON C++

3 OBJEKTORIENTIERTES PARADIGMA

1. Objekte kommunizieren durch das Senden und Empfangen von Nachrichten (welche aus Objekten bestehen)
2. Objekte haben ihren eigenen Speicher (strukturiert als Objekte)
3. Jedes Objekt ist die Instanz einer Klasse (welche ein Objekt sein muss)
4. Die Klasse beinhaltet das Verhalten aller ihrer Instanzen (in der Form von Objekten in einer Programmliste),

Auszug aus der Definition nach Alan Kay, 1993

KERNPUNKTE

1. Messaging
2. Persistenz
3. Kapselung
4. Späte Bindung

4 TEMPLATES UND METAPROGRAMMIERUNG

5 PERFORMANCE UND EFFIZIENZ

6 FUNKTIONALES PARADIGMA

LITERATUR

Alan Kay, The Early History of Smalltalk (programming language)

1993 *definition of "object oriented"*, https://de.wikipedia.org/wiki/Objektorientierte_Programmierung.

The Valuable Dev, Article on Fundamentals

2021 *What Are Abstractions in Software Engineering with Examples*, https://thevaluable.dev/abstraction-type-software-example/.