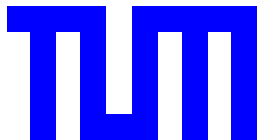
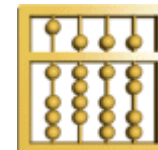

Zur Informatikausbildung im Grundstudium

Konzepte, Inhalte und Qualität

Manfred Broy



Technische Universität München
Institut für Informatik



Ziele der Informatikausbildung im Grundstudium

- Begriff von Informatik als Disziplin vermitteln

- ◊ Informatik als Disziplin

- Grundlagen- und Erkenntniswissenschaft
- Ingenieurwissenschaft

- ◊ Informatik in der Praxis - wirtschaftliche Bedeutung

- ◊ Informatik in den Anwendungsgebieten

- ◊ Rolle der Informatik in der Gesellschaft

- ◊ Das Grundlegende digitaler Technologie vermitteln

Problem: Selbstverständnis der Informatik als Wissenschaft noch im Fluss

- Überblick über wichtigste Teilgebiete der Informatik geben und Kerninhalte vermitteln

Problem: Welche Inhalte sind wie wichtig

- Zentrale Fertigkeiten in der Programmentwicklung vermitteln

Problem: Wie stark in heutige Techniken einführen

Im Vordergrund: Verständnis

Thesen zur Informatik im Grundstudium

- Ausbildung im Grundstudium ist die spannendste Herausforderung für den Dozenten
- Die Diskussion um die Informatik im Grundstudium spiegelt das Ringen der Informatik um ihr Selbstverständnis wieder
 - ◇ Grundlagen oder Hilfswissenschaft
 - ◇ Ingenieur- oder erkenntnisorientiert
 - ◇ Kern- oder Anwendungsfach
- Die Informatik kann sich oft nicht entscheiden, ob sie wissenschaftlich oder berufsorientiert ausbilden will
 - ◇ Konzentration auf Konzepte und Grundlagen
 - ◇ Schlagworte vermeiden, alles umfassend und genau erklären
- Anwendungsaspekte sollten im Grundstudium
 - ◇ höchstens exemplarisch
 - ◇ übersichtsartig (um einen Eindruck von der Anwendungsbreite der Informatik zu geben)behandelt werden
- Halbwertszeit der Inhalte kein wirkliches Problem

Beobachtungen zu den Studienanfängern Informatik im Eignungsfeststellungsverfahren

- Zahl der Anfängerstudenten insgesamt zu gering
 - ◇ Wirtschaft braucht höhere Zahl hochqualifizierter Informatiker
- Eignung sehr unterschiedlich
 - ◇ Beste Abiturienten gehen zur vereinzelt in die Informatik
 - ◇ Intellektuelle Herausforderung offenbar nicht erkennbar dargestellt
 - ◇ Kritischer Teil von „Computer-Freaks“
- Vorstellungen vom Informatikstudium sehr vage
- Kaum realistische Vorstellungen vom Beruf Informatiker
- Motivation
 - ◇ Gute Berufsaussichten
 - ◇ Faszination Computer

Beobachtungen zur Informatikstudenten im Grundstudium

- Erfolgsquote zu gering
 - ◇ Hohe Anzahl an Studienabbrechern (überwiegend gezwungenermaßen - nicht etwa weil das Studium nicht gefällt) im Grundstudium
 - ◇ Kaum Scheitern im Hauptstudium

Offensichtliche Frage:

- ◇ Wird zu streng ausgesiebt
- ◇ Ist unsere Didaktik angemessen
- Klassische Probleme am Ende vom Grundstudium
 - ◇ Ungenügende Fertigkeiten im Programmieren
 - ◇ Defizite in Kerninhalten

Was sollte der Student am Ende des Grundstudiums können

- Einfache bis mittelschwere Programmieraufgaben fehlerfrei systematisch lösen (Programme bis 300 Zeilen - Code gut strukturiert)
 - ◇ Spezifizieren und Strukturieren
 - ◇ Strukturiertes Programmieren
 - ◇ Analysieren (Performanz, Berechnungskomplexität)
 - ◇ Verifikation
- Zentralbegriffe der Informatik beherrschen
 - ◇ Information und ihre Darstellung
 - ◇ Wichtigste Programmierkonzepte (funktional, prozedural, OO, nebenläufig, ereignisgesteuert, kommunizierend)
 - ◇ Grundlegende Datenstrukturen (Keller, Schlange, Baum, ...)
 - ◇ Algorithmen (Sortieren, Suchen, effiziente Datenstrukturen, ...)
 - ◇ Hardware (Schaltungslogik, Rechnerstrukturen, maschinen nahe Programmierung)
 - ◇ Zentrale Modelle der Informatik kennen und verstehen
 - ◇ Berechenbarkeit
 - ◇ Berechnungskomplexität
 - ◇ Formale Sprachen (Zusammenhang zwischen Kompliziertheit der Beschreibungsmittel und der Ausdrucksmächtigkeit verstehen)
 - ◇ Modelle verteilter, nebenläufiger Systeme
 - ◇ Wichtigste Bestandteile der Systeminformatik (Betriebssystem, Compiler, Interpreter, Protokoll, Middleware, Kommunikationsnetz, Datenbank)

Problemfelder der Informatikausbildung im Grundstudium

- Stoffumfang in der Informatik nimmt drastisch zu
- Studenten kommen mit sehr unterschiedlichen Voraussetzungen
- Riesige Kluft zwischen den Grundthemen und den großen Anwendungssystemen
- Dozenten halten es oft für nicht nötig, die erforderliche Präzision zu bieten
- Viele Gebiete der Informatik sind nicht hinreichend wissenschaftlich erschlossen und werden phänomenologisch gelehrt
 - ◇ Beispiel: Betriebssysteme
 - ◇ Beispiel: Abstraktionsebenen, Schichtenarchitektur
- Große Unterschiede in den Vorgehensweisen
 - ◇ Programmieren in allen Einzelheiten
 - ◇ Programmieren unter Verwendung von Bibliotheken und Frameworks

Ein didaktisches Konzept zur Programmierung

- Information und ihre Darstellung
- Algorithmen und maschinenunabhängige Programmierung
 - ◇ Datenstrukturen
 - ◇ Funktionale Programmierung
- Programmierung sequentieller Maschinen (a la von Neumann)
 - ◇ Zuweisung, Speicher
 - ◇ Maschinenahe Programmierung
 - ◇ Zwangsläufig: Aufbau von programmierbaren Maschinen
- Verteilte nebenläufige Programme
- Große Softwaresysteme - Architekturen und Schnittstellen

Grundkonzepte - Systembegriff

- Schaltwerke und Schaltnetze
- Modelle von Softwaresystemen
 - ◇ Software als Zustandsmaschinen
 - ◇ Schnittstellensicht
 - ◇ Softwarearchitekturen
 - ◇ Interaktionssicht
- Software Engineering
- Eingebettete Systeme

Die „Gretchen-Frage“: Welche Programmiersprache

- Funktional
- Prozedural
- OO
- Nebenläufig
- Systemnah
- Maschinenorientiert
- Logische Programmierung

Warum Funktional zuerst kommen muss

- Sehr puristisch, wenige Konstrukte
 - ◇ Expression
 - ◇ Deklaration
 - ◇ Verzweigung
 - ◇ Rekursion
- Enge Kopplung operationelle und deskriptive Sicht
- Elementare Methoden der Spezifikation und Verifikation
- Starke Konzentration auf Algorithmik möglich
- Keine unverständlichen Rahmenkonzepte
- Enge an mathematisches Verständnis angelehnt
- Grundlage für alle anderen Programmierstile

Warum OO nicht am Anfang stehen darf

- Zentrale Begriffe nicht angemessen darstellbar:
 - ◇ Datenstrukturen: Beispiel Verkettete Liste
 - ◇ Algorithmik und Komplexitätstheorie
 - ◇ Funktionales Programmieren
- Zu viele Konzepte bis zur Unkenntlichkeit ineinander verwoben
- OO stärker auf Programmieren im Mittleren statt auf Programmieren im Kleinen ausgerichtet
- Obskure Konzepte, oft unausgegoren
 - ◇ Vererbung
 - ◇ Instanziierung

Elementares - Was beherrscht werden muss

- Spezifikation
 - ◇ Datenstrukturen
 - ◇ Beschreibung von Programieraufgaben
 - ◇ Domänenaspekte
- Strukturiertes Programmieren
- Analyse
- Verifikation
- Abschätzung Komplexität

Eine eher „konservative“ Sicht

- Grundkonzepte, -begriffe, -modelle und Theorien im Zentrum
 - ◇ Klassische Vorlesung
 - ◇ Begleitende Übungen mit Hausaufgaben und hoher Interaktion
 - ◇ Begleitende Praktika mit Arbeiten in Teams
- Vorlesung mit Schwerpunkt auch auf Motivation und Erklärung
- Informatik auf wenige Grundprinzipien reduzieren
 - ◇ Algorithmik
 - ◇ Logik und Berechnung
 - ◇ Induktion und Rekursion
 - ◇ Modelle
- Teilgebiete mit den Grundprinzipien in Zusammenhang bringen

Oft ist nicht das was, sondern das wie entscheidend

Das Beispiel Automatentheorie (und formale Sprachen)

- Klassische Form
 - ◇ Chomsky-Hierarchie
- Automaten allgemeiner
 - ◇ Zustandsmaschinen mit Ein-/Ausgabe
- Notationen der Praxis
 - ◇ Statecharts

Änderungen in der Informatik

Neue Schwerpunkte

- Verteilung
- Nebenläufigkeit
- Interaktion
- Netze
- Softwaresysteme
- Nutzerschnittstellen

Aber auch hier: Betonung auf Grundlagen

Abschließende Bemerkungen

- Schwächen der Informatik im Grundstudium sind Schwächen der Informatik als „unreife“ Disziplin
- Grundstudium sollte grundlagenorientiert sein
- Wissenschaftlicher Charakter sollte betont sein
- Im Vordergrund: Prinzipien, Konzepte, nicht Schlagworte
- Unreife, unausgeglichene Ansätze aus der Praxis sind zu vermeiden
- Informatik sollte als geschlossene Disziplin darstellbar sein/dargestellt werden
- Informatik im Grundstudium ist Herausforderung zur Flurbereinigung, zur Sauberkeit in der Begriffsbildung und zur Reduktion der Themen auf das Wesentliche in der Informatik

Wittgenstein: „Worüber man nicht sprechen kann, darüber muss man schweigen ...“