

Programmierkonzepte in der Physikalischen Chemie

33. Finale furioso: Zusammenfassung und Feedback

Albert-Ludwigs-Universität Freiburg

Dr. Till Biskup

Institut für Physikalische Chemie
Albert-Ludwigs-Universität Freiburg
Wintersemester 2016/17



**UNI
FREIBURG**

Zusammenfassung: Themen der Vorlesung im Überblick

Fünf Thesen

Ausblick: Wie könnte es weitergehen?

Feedback

- 1 Motivation
 - Programmierung ist essentieller Bestandteil von Forschung.
 - 2 Infrastruktur
 - notwendige Voraussetzungen für die Softwareentwicklung
 - 3 Code
 - Kommunikation erfordert Ausdrucksstärke und Lesbarkeit.
 - 4 Architektur
 - Modularität führt zu Flexibilität und Wiederverwendbarkeit.
 - 5 Datenverarbeitung und -Analyse in der PC
 - Gesamtkonzept von der Datenaufnahme bis zur Publikation
- 👉 Ziel: Software, die wissenschaftlichen Kriterien entspricht

These

Programmierung ist eine Kernkompetenz für moderne Naturwissenschaftler. Ihr Stellenwert ist viel zu gering.

- ▶ Datenauswertung ist (fast) immer rechnergestützt.
 - Programmierkenntnisse werden vorausgesetzt, sind aber selten in ausreichendem Maß vorhanden.
- ▶ Die Komplexität der Software entspricht der Fragestellung.
 - Beherrschung erfordert Kenntnis entsprechender Konzepte
- ▶ Softwareentwicklung wird nicht ausreichend gewürdigt.
 - Entsprechend gering ist die Motivation, Zeit zu investieren.

These

Wissenschaft beruht auf Nachvollziehbarkeit und Reproduzierbarkeit. Das ist in der Praxis oft nicht gewährleistet.

- ▶ Voraussetzungen: Dokumentation und Archivierung
 - Zeitskala: tendenziell Jahrzehnte
 - Stichworte: Unversehrtheit, Versionierung
- ▶ Realität sieht in vielen Fällen anders aus
 - Nachvollziehbarkeit der Auswertungen nicht gegeben
 - Reproduzierbarkeit von publizierten Ergebnissen unmöglich

These

Die primäre Aufgabe von Code ist Kommunikation.
Voraussetzung ist Ausdrucksstärke und Lesbarkeit.

- ▶ Code wird viel häufiger gelesen als geschrieben.
 - „Code for people, not computers.“
 - (Wirklich) guter Code ist offensichtlich.
- ▶ Ausdrucksstärke erfordert Abstraktionsvermögen.
 - Konzepte durchdenken, verstehen und treffend benennen
 - Gute Namen sind das Ergebnis eines Prozesses.

These

Kernaspekt aller Softwarearchitektur ist Modularität.
Das führt zu Flexibilität und Wiederverwendbarkeit.

- ▶ Ansprüche an Software ändern sich ständig.
 - Das Verständnis der Problemstellung wächst.
 - Jede Fragestellung ist neu und leicht unterschiedlich.
- ▶ Gute Software ist wie ein Legokasten.
 - kleine, durchdachte, zueinander kompatible Bausteine
 - nahezu endlose Möglichkeiten, Neues zu schaffen

These

Wissenschaftliche Datenauswertung erfordert ein Gesamtkonzept. Einzelaspekte sind (relativ) einfach umsetzbar.

- ▶ Nachvollziehbarkeit erfordert lückenlose Dokumentation.
 - von der Datenaufnahme bis zur Veröffentlichung
 - weitestgehende Automatisierung einzelner Schritte
- ▶ plattform- und medienunabhängig und modular
 - nutzerfreundlich: Fokus auf einfacher Bedienbarkeit
 - Vorteile der Nutzung müssen offensichtlich sein.

- ▶ Programmieren lernen
 - Die Programmiersprache ist (fast) egal.
 - Entscheidend ist das Bewusstsein für Konzepte.
- ▶ Anforderungsanalyse erstellen
 - Wie könnte ein System zur Datenverarbeitung aussehen?
 - Welche Aufgaben fallen immer wieder an?
- ▶ Softwareentwicklung ernst nehmen
 - Kriterien für Wissenschaftlichkeit beachten
 - Programme möglichst zukunftssicher gestalten

Tipps aus der Praxis

- ▶ schrittweise vorgehen: nicht alles auf einmal probieren
- ▶ Anfangen zu programmieren. Verbessern geht immer...



“ *Code as if whoever maintains your program is a violent psychopath who knows where you live.*

– Anonymous

Ein paar Fragen als Inspiration:

- ▶ Waren die Themen verständlich?
- ▶ Hat die Thematik für Sie eine Relevanz?
- ▶ Entsprach die Vorlesung Ihren Vorstellungen?
- ▶ Was könnte man besser machen?
- ▶ Was fehlte?
- ▶ Was war zu viel?
- ▶ Fühlen Sie sich den Ansprüchen an die Programmierung wissenschaftlicher Datenauswertung (mehr) gewachsen?
- ▶ ...