Softwareentwicklung am CliSAP Arbeitsbereich Meereisfernerkundung

Hagen Peukert

Arbeitsbereich Wissenschaftliches Rechnen
Fachbereich Informatik
Fakultät für Mathematik, Informatik und Naturwissenschaften
Universität Hamburg

2015-06-23

Gliederung (Agenda)

Einleitung

- Motivation
  - SE-Methodenwissen außerhalb der Informatik wichtig
  - Interessant, welche Unterschiede/Gemeinsamkeiten es gibt
- Ziel
  - Wie wird Software in den Naturwissenschaften entwickelt?
  - Welche *Best Practices* bilden sich dort heraus?
- Methodik
  - Qualitatives Interview
  - strukturiert, semi-standardisiert (Interviewleitfaden)
  - Arbeitsbereich Meereisfernerkundung
  - 1,5 h

Interviewpartner

- Maciej Miernecki (Doktorand)
- Lehrstuhl: Prof. Dr. Lars Kaleschke
- Forschungsgebiet B1: Arktische und Permafrost Gebiete Integrated Climate Systems and Analysis Prediction (CLiSAP)
- untersucht die "Rauhigkeit" des Eises und wie diese die Vorhersagekraft vorhandener Modelle verändert
- dazu ist die automatisierte Verabeitung von Satellitendaten mittels Software unabdingbar
- Eigenentwicklung aufgrund Themenspezifizität

Theoretischer Hintergrund

- Softwareentwicklung in der Informatik
  - Definition
  - Phasen der Softwareentwicklung
- Arbeitsbereich Meereisfernerkundung am CliSAP
  - Forschungsfragen und Hypothesen
  - Modelltheoretische Annahmen

Definition

Zielorientierte Bereitstellung und systematische Verwendung von Prinzipien, Methoden und Werkzeugen für die **arbeitsteilige**, **ingenieurmäßige** Entwicklung und Anwendung von **umfangreichen** Softwaresystemen. [Bal98, 36]

Softwareentwicklung in der Informatik

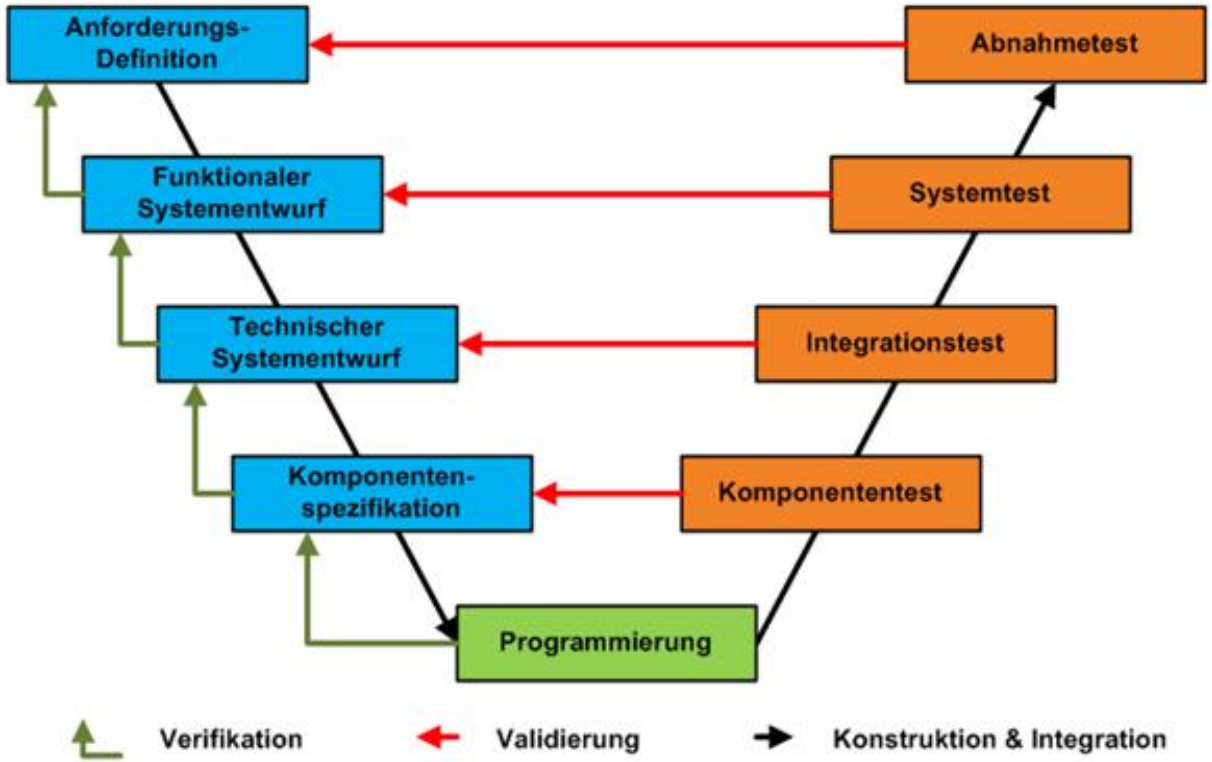
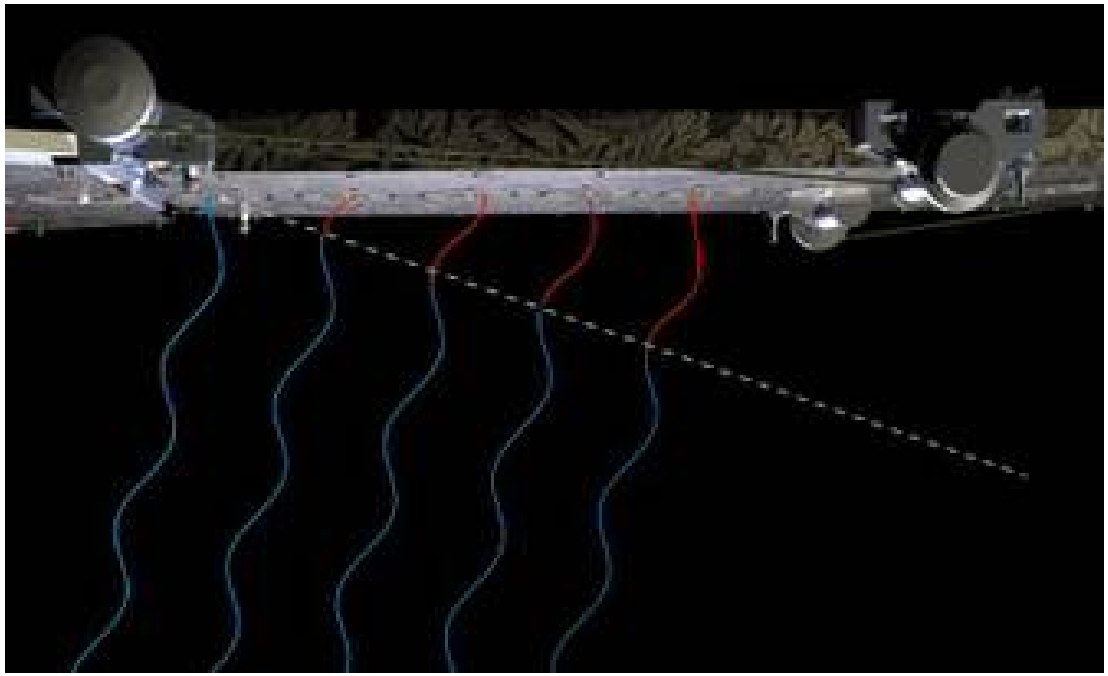


Figure: Das V-Modell der Softwareentwicklung [Bal98, 99]

Arbeitsbereich Meereisfernerkundung

- Forschungsfragen und Hypothesen
- Modelltheoretische Annahmen



Forschungsfragen und Hypothesen

- Grundsätzlich: Vermessen des Eises mit SMOS (**S**oil **M**oisture and **O**cean **S**alinity) Satelliten Daten (Dicke, Größe, Anzahl der Rinnen)
- Probleme:
  - extrem hohe Variabilität der Meßdaten durch Störgrößen (Wolken, Strahlung)
  - nicht alle Informationen sind in Satellitendaten enthalten
  - *Gold Standard* nicht vorhanden
- Methodik analog zur Signalverarbeitung

Modelltheoretische Annahmen

- physikalische Größen (z.B. Einfallswinkel des Lichts)
- Strahlungs- und Transfermodelle
  - Emmissivität ist Funktion der Temperatur, des Alters
  - Höhe
  - in Wassernähe kälter
- Annahme:
  - 50 x 50 km Flächen sind gleich dick
  - Physikalische Modelle widerspruchsfrei anwendbar
  - betrachtet wird immer nur ein kleiner Theorieausschnitt

Datenmanagement

- Umgang mit Projektdaten
- Versionierung
- Dokumentation

Umgang mit den Projektdaten I

- Archivierung in einfacher Verzeichnisstruktur
  - Rohdaten werden nicht abgelegt
  - spiegelt die "Filterung"/Bearbeitung der Daten wider
  - erlaubt das Arbeiten in verschiedenen Abstraktionsstufen
  - geordnet nach Ort (Gitter 50 x 50), physikalischen Einheiten
- Daten sind nach Validierung öffentlich (http://www.icdc.zmaw.de)

Umgang mit den Projektdaten II

- Bearbeitung der Daten heisst, sie werden
  - in ihrer Anordnung verändert (rearranged)
  - komprimiert
  - für die entsprechende Hypothese aufbereitet (reprocessing)
- Verwendete Standardformate
  - hdf5, netcdf, .h5, tiff, qGis
  - binaries, ascii
  - xml zur Metadatenbeschreibung (headerfile)

Versionierung

- keine Versionierung während des Entwicklungsprozesses
  - Programme werden kontinuierlich in einer Datei fortgeführt
  - Sicherheitskopien von zu letzt lauffähigen Versionen existieren in den Anwenderverzeichnissen
- nach Fertigstellung des Programm
  - Upload in das Repositorium einer Projektmanagement-Software (Service ZMAW)
  - Anpassung bzw. Weiterentwicklung von anderen Forschern und andere (ähnliche) Projekte dann möglich
  - Vorlagen (Python, Toolbox) sind vorhanden

Dokumentation

- erfolgt überwiegend auf den Seiten der Wiki der Projektmanagement-Software (insb. Festhalten der Durchschnittsvariablen)
- Gesamtdokumentation erfolgt nach Fertigstellung (nicht fortlaufend)
- d.h. keine systematische Dokumentation in Klassen, Methoden, Paketen

Dokumentation

- teilweise werden Hinweise in den Quellcodescripten zu einzelnen Routinen gegeben
- Namenskonventionen werden eingehalten
- Struktur des Programm wird im datafile festgehalten
- weitere Informationen werden im Logfile weitergegeben

SE-Outlines

- (Use Case)
- Architektur und Design
- Implementierung
- Testen
- Wartung

SE-Prozess I

- Design und Architektur
  - läuft unterbewusst ab
  - aufgrund der kleinen Forschungsprojekte sind Softwarelösungen überschaubar
- Implementierung
  - Entwicklung auf dem Cluster (SNOW) mit Python
  - Objektorientierte Entwicklung
  - schrittweise Vervollständigung der Funktionen
  - keine Teamentwicklung, da Projektumfang auf das Dissertationsprojekt einer Person beschränkt

SE-Prozess II

- Testen
  - "Versuch und Irrtum" – Prinzip
  - schrittweise von einfach zu komplexen, d.h. einzelne Python Funktionen werden allein, dann in größer werdenden Verbünden getestet
  - Plausibilitätsprüfungen, z.B. mit Standardabweichungen (bekannt – unbekannt)
- Wartung
  - findet nicht statt oder nicht vom ursprünglichen Entwickler
  - Probleme werden bei Umstellung auf Python 3.0 erwartet

SE-Prozess III: Beispiel Salzprofil im Eis

- Frage: Funktioniert Modell, macht es die Vorhersagen, die "üblich" sind
- Fehler ergeben sich aus diesem Prozess
- Test zuerst für dünnes Eis (hier gibt es weniger Variablen), dann dicke der Eisschicht wird erhöht
- Prüfung nach jeder Steigerung der Eisdicke, ob die Ausgaben realistisch sind
- "hope-the-best": Ausgaben werden auf Widerspruch geprüft (hinsichtlich des Modells), d.h. gibt es Brüche (cuts) in den homogenen Feldern?

Zusammenfassung I

- Umgang mit Forschungsdaten
  - Ordner- und Verzeichnissystem auf dem Cluster
  - Archivierung in verschiedener Detailtiefe
- Dokumentation
  - Direkt im Quellcode (zum Verständnis der Implementierung)
  - Metadatenbeschreibung der Programmstruktur im "Data-File"
  - Allgemeine Beschreibung im Wiki

Zusammenfassung II

- Versionierung
  - Repositorium vorhanden
  - Fertige SE-Projekte werden eingestellt
- SE-Prozess
  - "ergebnisorientiert"
  - SE-Phasen unbewusst (Design und Architektur)
  - Testen nach Plausibilitätsannahmen
  - keine systematische Wartung oder kontinuierliche Verbesserung

Fazit I

- Gretchenfrage der Informatik sekundär
  - Laufzeit – Effizienz – Lesbarkeit
  - Skripte benötigen bis 13h, um durchzulaufen
  - keine Parallelisierung angedacht
- Modell der agilen SE eher zutreffend als V-Modell
  - kurze Entwicklungszyklen
  - schnelle Änderungen möglich
  - fortlaufendes Testen von Teillösungen
  - fertiges Programm entsteht nach mehreren Zyklen
  - nicht alle Prinzipien zutreffend (Team nicht gegeben)

Fazit II

Hypothese: SE-Prozess in den (Natur)Wissenschaften bildet sich selbstorganisatorisch heraus und strebt nach einem effizienzoptimalen Punkt der SE, der den Prinzipien der agilen SE stark ähnelt

Literatur


Helmut Balzert.
Lehrbuch der Software-Technik.
1998.