Seminar „Softwareentwicklung in der Wissenschaft“

Überblick über Softwareentwicklung

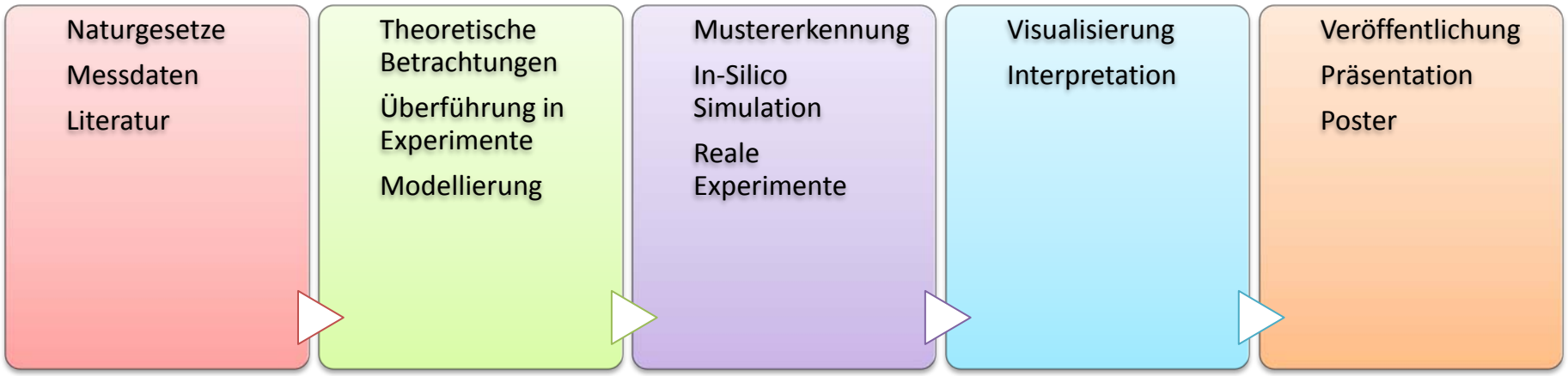
Julian Kunkel
Prof. Dr. Thomas Ludwig, Dr. Hermann Lenhart, Petra Nerge

Gliederung

- Wissenschaftlicher Erkenntnissgewinn
- Kurze Einführung in Softwareengineering
- Bewertung von Code Qualität
- Einige relevante Begriffe
- Fragestellungen für die Interviews

Wissenschaftlicher Erkenntnissgewinn

Werkzeuge/Methoden



Finanzierung

- Forschung an Universitäten
  - Doktoranden
  - Professoren
  - Über Projekte (Kollaborationen)
  - *Veröffentlichungen stehen im Vordergrund*
- Grundlagenforschung in Industrie
  - *Vermarktung steht im Vordergrund*
- *Widerspruch: Code-Qualität vs. erwartete Ergebnisse!*

Überführung der Fragestellung in Code

- Modellierung
  - Abstraktion des Problems.
  - Hypothesenbildung
  - Gleichungen
  - Parametrisierung?
- Algorithmen
- Numerik
  - Genauigkeit/Korrektheit der Ergebnisse?
- Codierung einer Programmiersprache

- Auf welchen Systemen läuft der Code?
  - Hochleistungsrechnen!

- Wissenschaftliche Probleme nicht in 5 Zeilen codiert!
  $\Rightarrow$ Softwaretechnik ist wichtig!

Software Engineering (SWE)

„Zielorientierte Bereitstellung und *systematische* Verwendung von *Prinzipien*, *Methoden* und ***Werkzeugen*** für die arbeitsteilige, ingenieurmäßige Entwicklung und Anwendung von umfangreichen Softwaresystemen.„ (Helmut Balzert/2001)

Einige Fakten zur Softwareentwicklung

- ca. 23 % aller Softwareprojekte *erfolgreich*.
- ca. 53 % aller Softwareprojekte über Budget und/oder über Zeit.
- ca. 24 % aller Softwareprojekte abgebrochen.

„Softwareentwicklung ist in besonderem Maße geprägt von Fehleinschätzungen. Sehr häufig wird der Zeitbedarf zu kurz geschätzt, sowohl für die Projektorganisation als auch für den Kommunikationsbedarf und die Programmierdauer.“

[Quelle: http://www.torsten-horn.de/techdocs/sw-dev-process.htm]

- Lines-of-Code pro Programmierer:
- 10 LOC pro Arbeitstag laut [Mayr 2005]
- 16 LOC pro Arbeitstag laut [Ludewig/Lichter 2006]

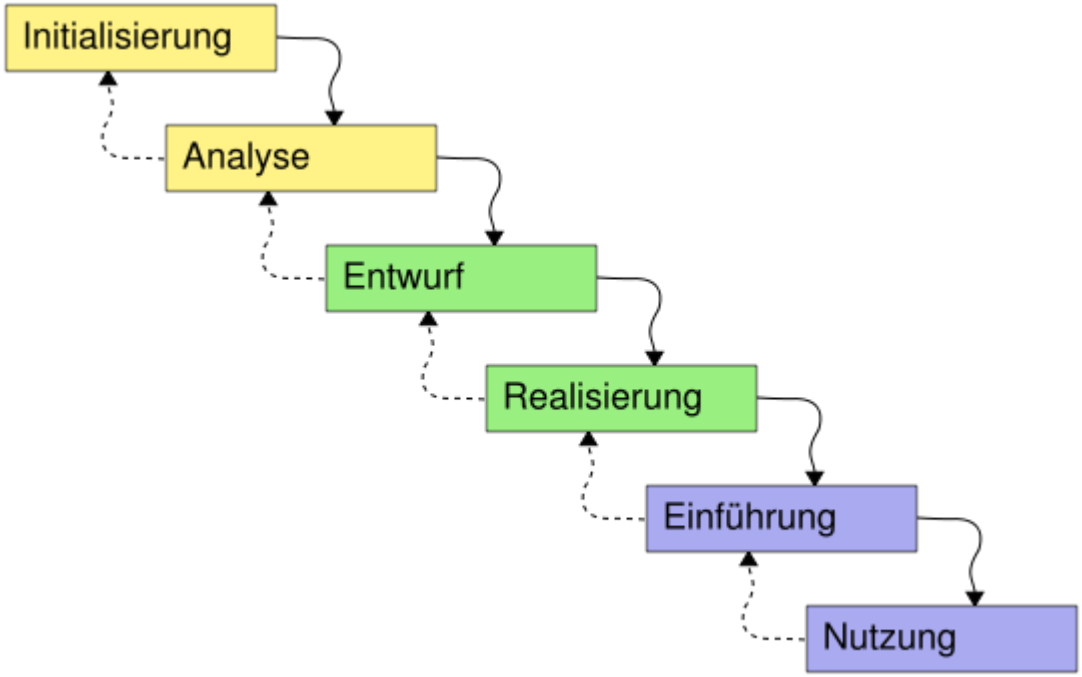
Teilgebiete des SWE

- **Projektmanagement**
- **Qualitätsmanagement**
- Risikomanagement
- Anforderungserhebung
- Systemdesign/technische Konzeption
- Implementierung
- **Softwaretest**
- **Softwareeinführung**
- **Wartung/Pflege**

(Mit den Fett gedruckten Bereichen werden wir uns beschäftigen.)

Software-Entwicklungsprozess

- Herangehensweise an die Entwicklung
- z.B. Wasserfallmodell



[Quelle: http://de.wikipedia.org/wiki/Wasserfallmodell]

Bewertung von Software ISO/IEC 9126

- Funktionalität
  - Angemessenheit, Richtigkeit, Interoperabilität, Sicherheit, Ordnungsmäßigkeit
- Zuverlässigkeit
  - Reife, Fehlertoleranz, Robustheit, Wiederherstellbarkeit, Konformität
- Benutzbarkeit
  - Verständlichkeit, Erlernbarkeit, Bedienbarkeit, Attraktivität, Konformität
- Effizienz
  - Zeitverhalten, Verbrauchsverhalten, Konformität
- Änderbarkeit
  - Analysierbarkeit, Modifizierbarkeit, Stabilität, Testbarkeit
- Übertragbarkeit
  - Anpassbarkeit, Installierbarkeit, Koexistenz, Austauschbarkeit, Konformität

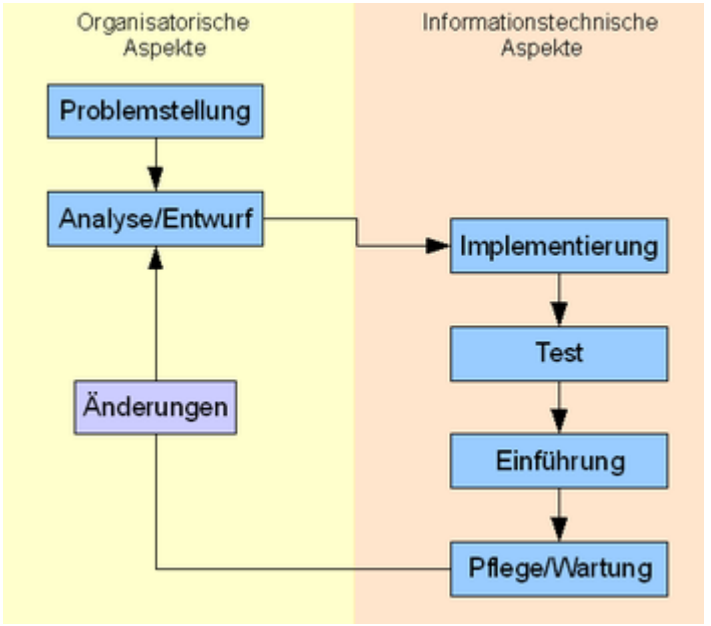
[Quelle: http://de.wikipedia.org/wiki/ISO/IEC_9126]

Weitere Begriffe

- Software wird typischerweise wiederverwendet:
  - In der Wissenschaft => verwandte Fragestellungen.
  - Software-Lebenszyklus
  - Release-Management
  - Versionsverwaltung
- Change-Management
  - Umgang mit: Neuen Features, Bugs, Portabilität auf neue Systeme

Software-Lebenszyklus

- „Ein Software-Lebenszyklus beginnt vor der Markteinführung einer Softwarelösung und besteht aus den Phasen Problementstehung, Entwicklungsprozess, Implementierung und Nutzung und wird durch die Ablösung der Software durch ihren Nachfolger abgeschlossen.“



[Quelle: http://de.wikipedia.org/wiki/Software-Lebenszyklus]

Release-Managment

- Aufgaben des Release Managments:
  - Festlegung des *funktionellen Umfangs*
  - Festlegung des genauen *Zeitplans* einer Release-Freigabe in Abstimmung mit dem Change- bzw. Produktmanagement
  - *Qualitätskontrolle* zur Überwachung der Einhaltung der Kriterien, die im Rahmen des Change- bzw. Produktmanagements für eine Release-Erstellung festgelegt wurden
  - *Dokumentation* des Umfangs und der Änderungen, dabei insbesondere Beschreibung der für die Rückwärtskompatibilität relevanten Eigenschaften
  - Verwaltung der *Versionshistorie* (Versionierung), damit Sicherstellung der Reproduzierbarkeit

[Quelle: http://de.wikipedia.org/wiki/Release_Management]

Versionsverwaltung

- Änderungen an Dokumenten protokollieren.
  - Wer hat was geändert und wann.
- Wiederherstellen beliebiger Versionen.
  - Zustand vor 3 Monaten, letztes Release…
- Koordinierung des Zugriffs von Entwicklern.
  - Methodik zur Nutzung des Systems?
- Parallele Entwicklungszweige.
  - Erweiterung des Programms in viele Richtungen.

Software-Architektur

- Beschreibt Komponenten und Zusammenspiel
- Komponente:
  - „Sinnvolle“ Funktionseinheit.
  - Nutzt Schnittstellen und stellt bereit.
- Beispiel:
  - Einlesekomponente, Rechenkomponente, Ausgabekomponente

Qualitätskontrolle

- Fragen:
  - Berechnet das „Programm“ das Richtige? - Validierung
  - Rechnet das „Programm“ richtig? - Verifikation

- Qualität sicherstellen mittels:
  - Softwareentwicklungsprozesse
  - Code Qualität:
    - Tests
    - Dokumentation
  - Abschätzungen der Modelkorrektheit - andere Frage.

Softwaretest

- Softwaretest misst Qualität
- Fehler aufdecken
- Nicht: Beweis der Programm-Fehlerfreiheit
- Nachweis, dass das Programm fehlerfrei ist, ist schwer
  - „Verifikation“: formale Korrektheit beweisen.

Weitere relevante Aspekte für die Interviews

- Werden SWE-Methoden direkt oder Vergleichbares implizit angewendet? Wie genau?
- Kollaboration(Konsortiumgröße)
- Programmiersprachen
- Werkzeuge zur Entwicklung (Editor? IDE?)
- Abhängigkeiten: Software-Bibliotheken ...
- Automatisierung der Prozesse
- Genutzte Hardwaresysteme
- Reproduzierbarkeit der Ergebnisse
- Eingabedaten (Größe, Ursprung)
- Ausgabedaten (Antwort auf Frage?)
- Datenmanagement (Umgang mit Daten)
- Anforderungen an die technische Umsetzung

- Historie des Projektes
- Erfahrungen während Entwicklung, Systeme?