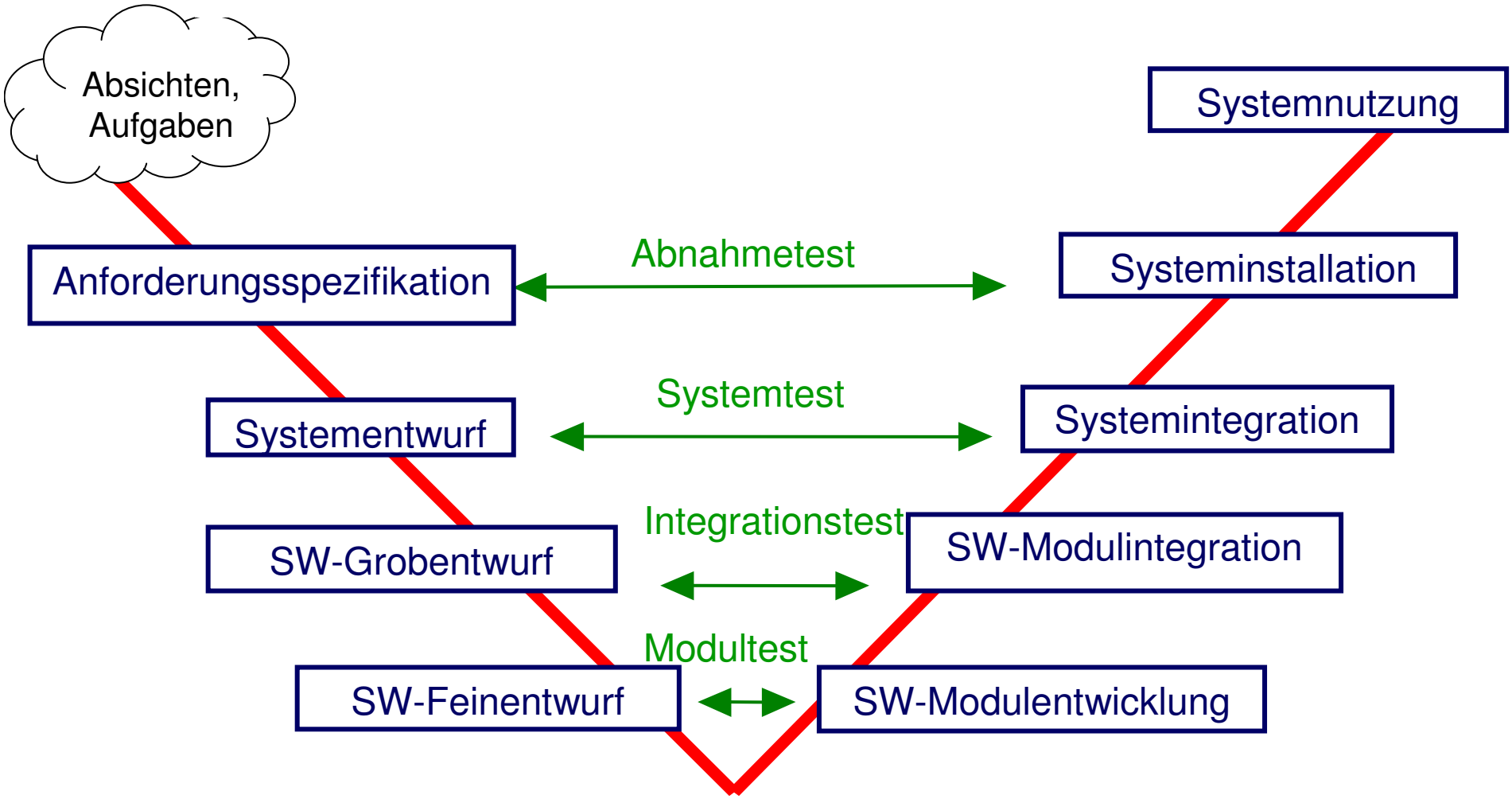
Praktikum Software Engineering: Verfahren und Werkzeuge

Lehrstuhl für Software Engineering

(Informatik 11)

Software Engineering



Praktikum Software Engineering

- **Projekt-Management**
  - Erfassung relevanter Prozessgrößen (Meilensteine, Ressourcen, etc)
  - Koordination des Personalseinsatzes
- **Anforderungsanalyse / Spezifikation**
  - Modellierung und Simulation des spezifizierten Systemverhaltens
  - Nachweisen bzw. Widerlegen relevanter Sicherheits- bzw. Lebendigkeitseigenschaften
- **Objektorientierte Analyse und Design**
  - Statische Modellierung der Architektur (Klassendiagramme)
  - Dynamische Modellierung der Aufrufsequenzen (Interaktionsdiagramme)
- **Implementierung / Wartung**
  - Generierung von Programmskeletten
  - Verwaltung aufeinanderfolgender Versionen
  - Erfassung und Bearbeitung eingehender Fehlermeldungen bzw. Änderungswünsche
- **Testen / Verifikation**
  - Erfassung des Fortschreitens funktionaler Tests (Anteil getesteter Anforderungen)
  - Erfassung des Fortschreitens struktureller Tests (Codeinstrumentierung → Überdeckung)
  - Verifikation durch interaktive Korrektheitsbeweise
- **Quantitative Bewertung des Produkts**
  - Ermittlung quantitativer Komplexitätsindikatoren (Softwaremetriken)

Praktikum Software Engineering

- Aufgrund der wachsenden logischen Komplexität
  - systematische Vorgehensweisen nicht manuell zu realisieren
  - mühsam und fehleranfällig
- Lösung
  - industrieller Einsatz von **Werkzeugen**, d. h. unterstützender Programme, die entsprechende Schritte des Software Engineering zu automatisieren erlauben
- Praktikum
  - individuelle praktische Erprobung der vorgestellten Werkzeuge unter möglichst realen Randbedingungen

Praktikum Software Engineering

- Ziel

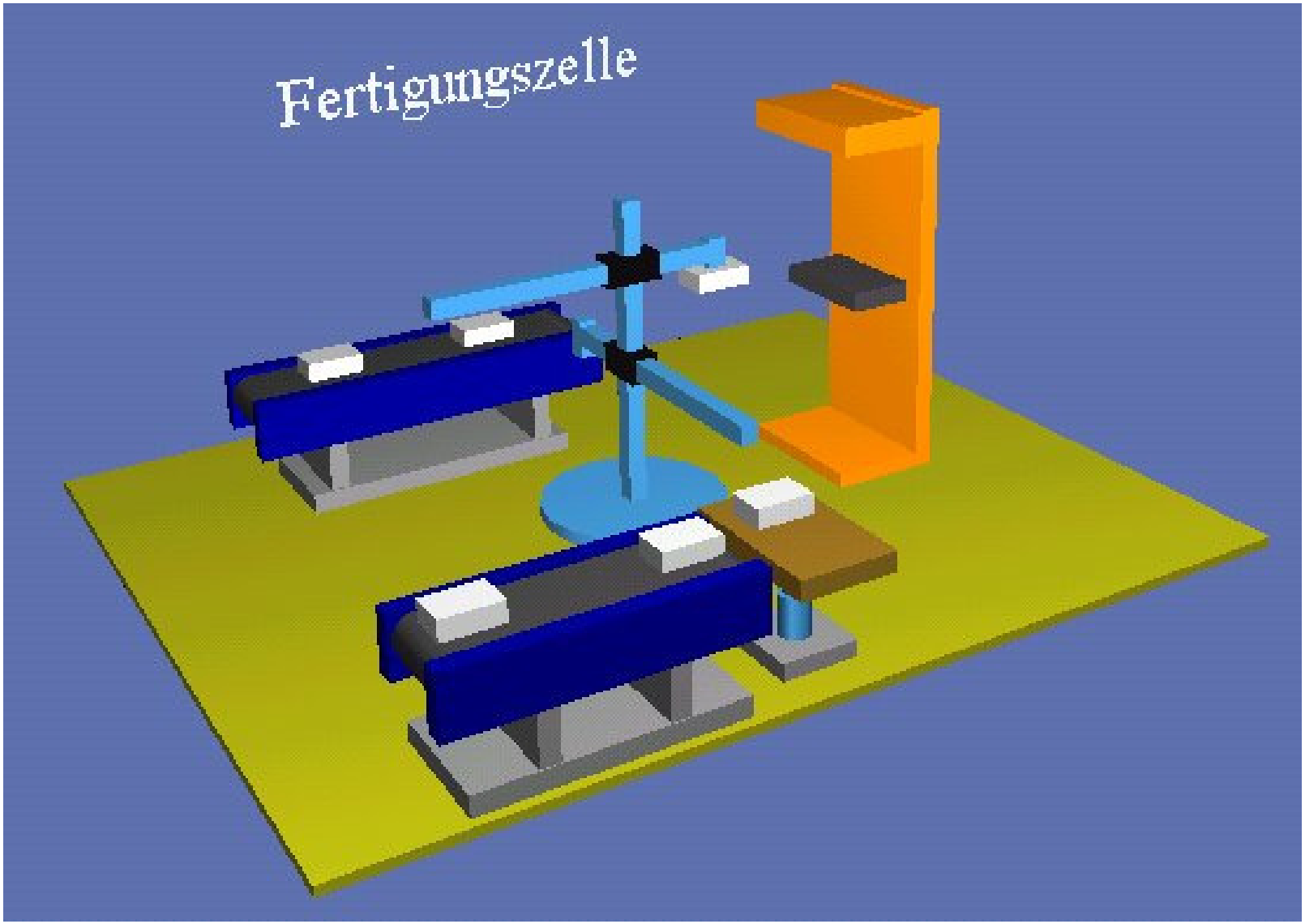
  Potenzial und Grenzen unterschiedlicher Werkzeuge zur Unterstützung softwaretechnischer Tätigkeiten werden durch deren praktischen Einsatz bei Analyse, Entwurf, Implementierung, Testen und Projektmanagement vermittelt.
  - Erkennen der Notwendigkeit bzw. der Vorteile des Werkzeugeinsatzes bei der Software-Entwicklung
  - Kennenlernen der Stärken und Schwächen verschiedener Werkzeuge

- Wie?
  - Werkzeuge zur Entwicklung und zur Analyse komplexer Software werden vorgestellt
  - deren praktische Einsetzbarkeit anschließend von den Teilnehmern anhand realer Aufgabenstellungen in ausgewählten Phasen des Software-Lebenszyklus erprobt wird

Verfahren und Werkzeuge

- Projektmanagement
  - Meilensteine (Trac)
  - Workflows (Trac)
- Spezifikation
  - Anforderungserfassung und –analyse (Requisite Pro)
  - Petri-Netze (WinPetri, TINA)
  - Model Checking (NuSMV)
- Objektorientierte Analyse und Design
  - Erstellung der UML-Diagramme (Eclipse, Borland Together)
- Implementierung / Wartung
  - Codierung / automatische Codegenerierung (Eclipse, Borland Together)
  - Statische Analyse (FindBugs, Checkstyle)
  - Versionskontrolle (Subversion)
  - Bug Tracking (trac)
- Testen / Programmverifikation
  - Funktionales Testen (JUnit)
  - Strukturelles Testen (DjUnit, Clover)
  - Programmbeweis (KIV)
- Quantitative Bewertung
  - Softwaremetriken (Together)

Die automatische Produktionszelle



Fallbeispiel: automatische Produktionszelle