Verlässliche Echtzeitsysteme

Lehrveranstaltungskonzept

Fabian Scheler

Friedrich-Alexander-Universität Erlangen-Nürnberg
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)

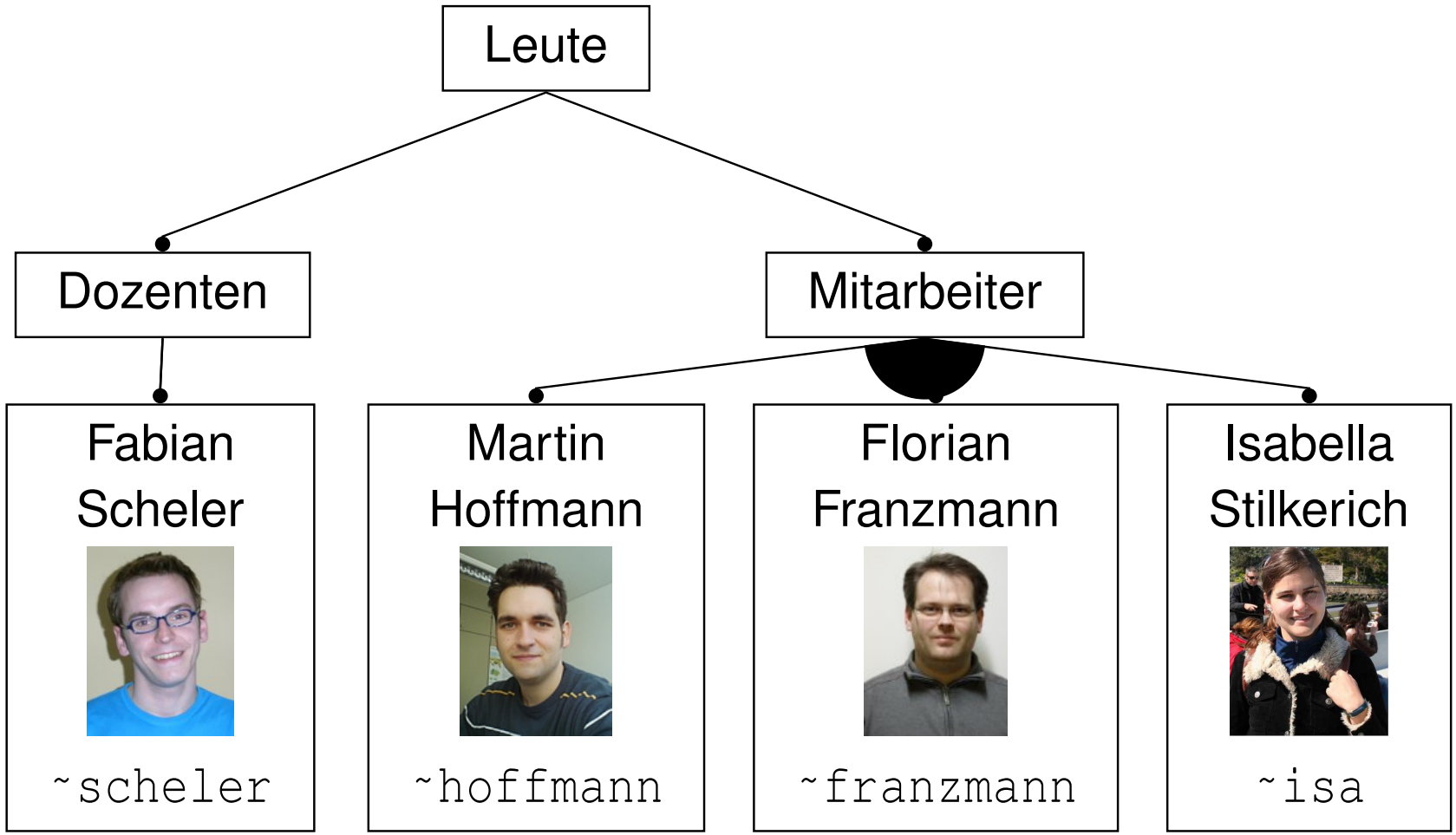
`www4.informatik.uni-erlangen.de`

17. April 2012

Gliederung

1. Vorstellung
   - Dozenten
   - Vorlesung
   - Lernziele
2. Einordnung
   - Studiengänge
   - Merkmale
   - Lehrkanon
3. Ablauf
   - Vorlesung und Übung
   - Leistungsnachweise

www4.informatik.uni-erlangen.de/*



Verlässliche Echtzeitsysteme – Motivation

⇝ Persönliches Interesse!

- Neue Verfahren und Architekturen zu entwickeln, ist spannend!
  - Mikrokerne schotten Programme voneinander ab
  - sporadisches Zusteller begrenzen den zeitlichen Einfluss
    - den nicht-periodische Aktivitäten auf periodische Arbeitsaufträge haben
  - Verschlüsselungsalgorithmen garantieren sichere Kommunikation
  - ...

⇝ Das ist aber nur die halbe Miete!

- Diese Verfahren und Architekturen müssen korrekt arbeiten!
  - erfordert möglichst fehlerfreie Implementierungen
  - die Implementierung muss mit Laufzeitfehlern umgehen können
    - Fehler rühren nicht nur von Software-Bugs ...

Fokus: Software

Zuverlässig Software entwickeln

- Wie kommt man zu einer möglichst fehlerfreie Implementierung?
- Welche Werkzeuge helfen mir dabei?
  - Was tun dieser Werkzeuge eigentlich?
  - Welche Grenzen haben diese Werkzeuge demzufolge?
- Hier „lernen“ wir zusammen mit euch ;-)

Zuverlässige Software entwickeln

- Wie geht man zur Laufzeit mit Fehler um?
  - Wie erkenne und toleriere ich solche Fehler?
- Wie testet man, ob man korrekt mit solchen Fehlern umgeht?
- Hier „forschen“ wir (hoffentlich auch zusammen mit euch :-))

Forschung und Lehre funktioniert nur mit Studenten!

Wanted: Studenten/Innen für/als

- Bachelor-, Master-, Studien- und Diplomarbeiten
- Bachelor-Praktikum und Master-Projekte
- studentische Hilfswissenschaftler (Hiwis)
- wissenschaftliche Mitarbeiter

Lernziele

Zuverlässig Software entwickeln

- typische Laufzeitfehler in C/C++-Programmen suchen und finden
  - Nullzeiger, Ganzzahlüberläufe, nicht initialisierte Speicherstellen, ...
  - mithilfe statischer Analysewerkzeuge
- *Design-by-contract*: statische, werkzeug-gestützte Verifikation
  - Formulierung/Verifikation von Nachbedingungen für kleine C-Programme
  - *ANSI C Specification Language* (ACSL) und *Frama-C*
- *Testüberdeckung*: Wie gut hat man getestet?
  - die Testüberdeckung für ein gegebenes Programm messen
  - Gibt es Zusammenhänge zwischen der Testüberdeckung, der Testfallanzahl und anderen Metriken?

☞ Vorurteile gegenüber formalen Methoden abbauen

- sie sind keine unverwendbaren Monster mehr
  - sie vollbringen aber auch keine Wunder
  - ihre Anwendung ist noch immer mühsam, aber sie lohnt sich

Lernziele (Forts.)

Zuverlässige Software entwickeln

- Maskieren von Fehlern durch redundante Ausführung
  - räumlich und zeitlich redundante ausführung
  - homogene und heterogene Redundanz
- Härtung von Datenstrukturen und Kontrollfluss
  - Fehlererkennung und -korrektur innerhalb
    - von Daten mithilfe von z.B. Prüfsummen
    - in Berechnungen/Kontrollfluss mithilfe erweiterter arithmetischer Kodierung
- proaktive Methoden zur Maskierung von Fehlern
  - Neustarts, die zyklisch oder im Fehlerfall durchgeführt werden

☞ Anknüpfungspunkte für den praktischen Einsatz aufzeigen

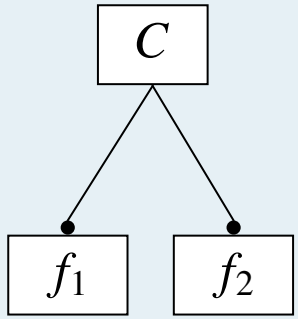
- Niemand braucht das 1001 Fehlertoleranzprotokoll!
  - das den gegenwärtigen Stand der Kunst nicht reflektiert
  - und obendrein vielleicht fehlerhaft ist

Gliederung

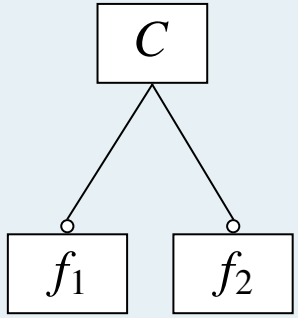
Vorwort

Die Lehrveranstaltung ist grundsätzlich für alle Studiengänge offen. Sie verlangt allerdings gewisse Vorkenntnisse. Diese müssen nicht durch Teilnahme an den Lehrveranstaltungen von I4 erworben worden sein.

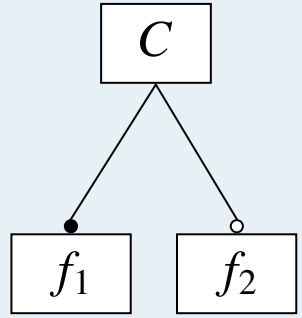
Lehre@I4 – Beschreibungshilfsmittel



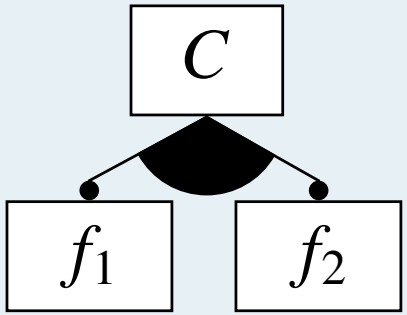
- Verpflichtungen
- $f_1 \cdot f_2$



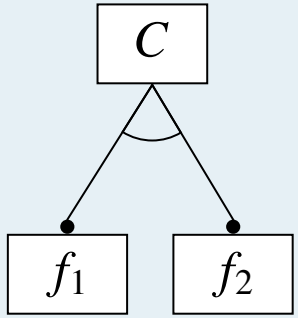
- Optionen
- $\emptyset, f_1, f_2, f_1 \cdot f_2$



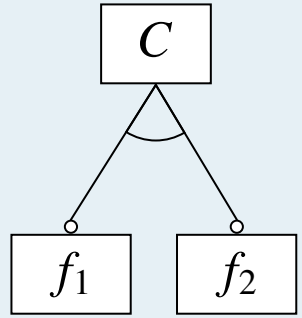
- Zusatzoption
- $f_1, f_1 \cdot f_2$



- Anhäufung
- $f_1, f_2, f_1 \cdot f_2$

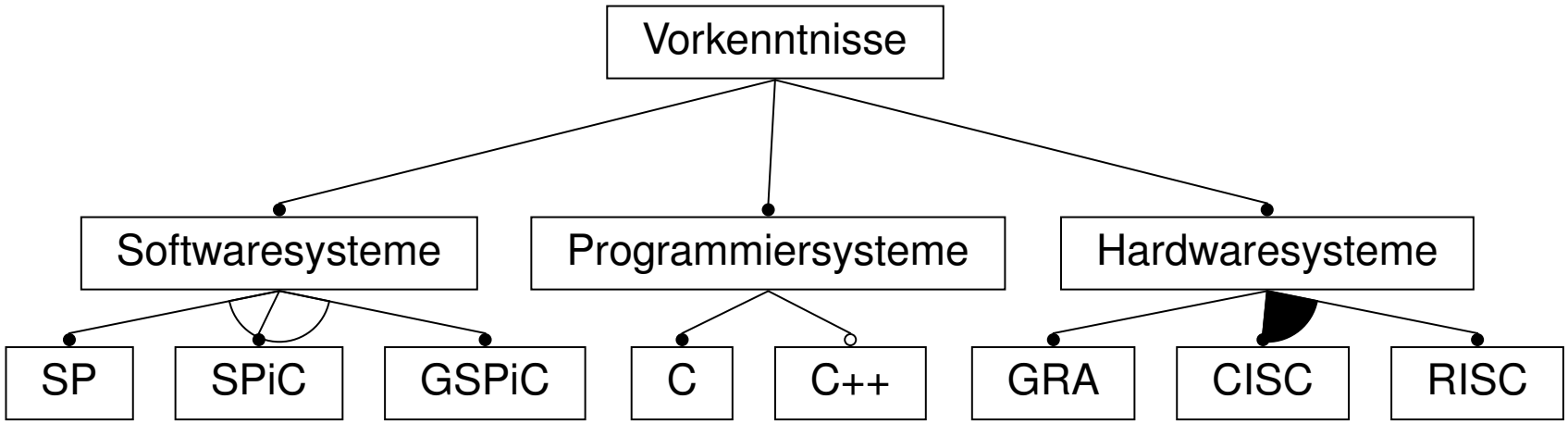


- Alternative
- f_1, f_2



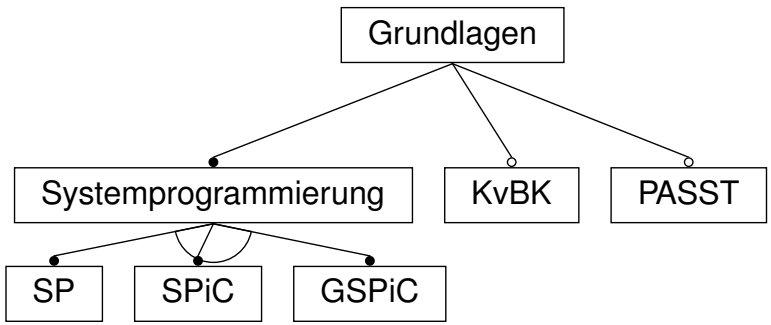
- Alternativoption
- $\emptyset, f_1, f_2$

Anforderungen

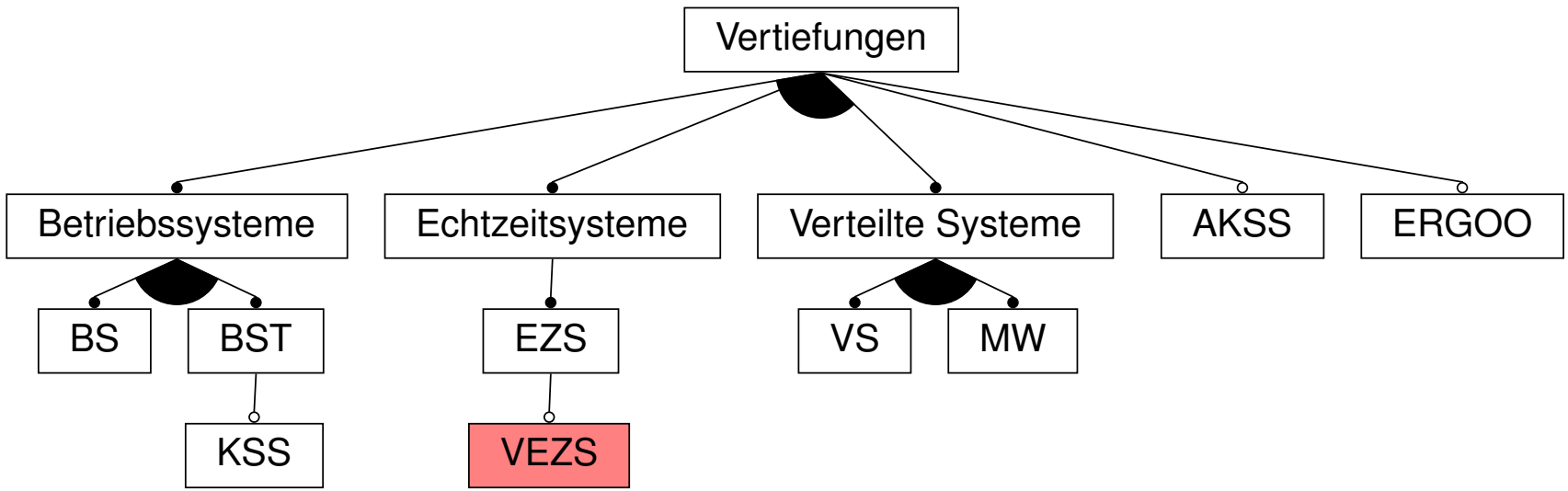


- sich an hardware-naher Programmierung erfreuen können
- „Furchtlosigkeit“ vor nur schwer erkund-/fassbaren Sachverhalten
- ein gewisses Maß an Durchhaltevermögen mitbringen und zeigen

Lehre@I4 — Überblick



Sage es mir und ich vergesse es,
zeig es mir und ich erinnere mich,
lass es mich tun und ich behalte es.
(Konfuzius, 551–479 v. Chr.)



Lehre@I4: *post* SP — Aufbau und Spezialisierung

BS	Betriebssysteme	V/Ü
VS	Verteilte Systeme	V/Ü
EZS	Echtzeitsysteme	V/Ü
KvBK	Konzepte von Betriebssystemkomponenten	PS

BST	Betriebssystemtechnik	V/Ü
BST	Konfigurierbare Systemsoftware	V/Ü
MW	Middleware	V/Ü
VEZS	Verlässliche Echtzeitsysteme	V/Ü
AKSS	Ausgewählte Kapitel der Systemsoftware	HS
ERGOO	Erlangen Research Group on Distributed Objects and Operating Systems	KO

Gliederung

Vorlesungsbetrieb und Lehrmaterialien

Vorlesungstermine

- wie automatisch eingeplant

Termine bis KW 29

- Dienstag, 16:00 – 17:30, 01.255-128

Ausfälle

- KW 18, 01.05.2012
- KW 20, 15.05.2012
- KW 22, 29.05.2012

- vom Dozenten erwünscht/erhofft: Verschiebung auf Mittwoch

Termine bis KW 29

- Mittwoch, 14:15 – 15:45, 01.150-128

Ausfälle

- KW 20, 16.05.2012
- KW 22, 30.05.2012

Handzettel (engl. *handout*) sind verfügbar wie folgt:

- `www4.informatik.uni-erlangen.de/Lehre/SS12/V_VEZS`
- Folienkopien werden vor der Vorlesung ausgegeben

Ausnahme: Industrievortrag am 30.04.2012

Termine KW 18

- Montag, 14:15 – 15:45, 00.031-113

- Vortragender: Bernhard Sechser
  - Method Park Software AG – Principal Consultant SPICE & Safety

- Thema: Safety-Normen in der Automobilelektronik
  - Automobile sind sicherheitskritische, verteilte Systeme auf Rädern

  ⇝ Was schreibt der Gesetzgeber vor, um Fehler zu vermeiden?

  - Thematik wird großteils von der Industrie verwendet und getragen

  ⇝ Ein industrieller Vertreter kann hier besser Auskunft geben, als ein Wissenschaftler!

Übungsbetrieb

Termine bis KW 6

- siehe Webseite VEZS

Ausfälle

- KW 22

Tafelübung

- Anmeldung über WAFFEL(URL siehe Webseite von VEZS)
- Übungsaufgaben sind bevorzugt in Gruppen zu bearbeiten

Rechnerarbeit: komplett in Eigenverantwortung

- Anmeldung ist nicht vorgesehen, keine reservierten Arbeitsplätze
  - ein Termin für eine Rechnerübung wird aber vorgesehen
- bei Fragen zu den Übungsaufgaben, Übungsleiter konsultieren
  - eMail an die Mailingliste senden
  - bei kniffligen Fragen: vorbei kommen oder Rechnerübung

Bedeutung von Tafel- und Rechnerübungen

Tafelübungen $\rightsquigarrow$ „*learning by exploring*“

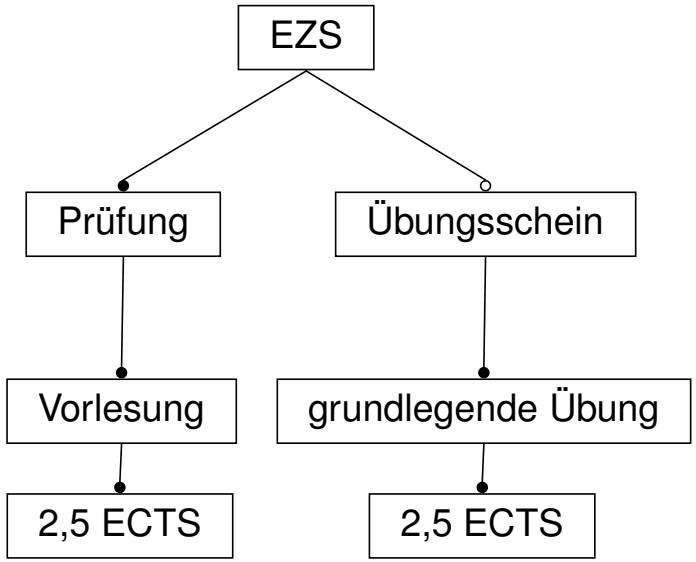
- Besprechung der Übungsaufgaben, Skizzierung von Lösungswegen
- Vertiefung des Vorlesungsstoffes, Klärung offener Fragen

Rechnerarbeit $\rightsquigarrow$ „*learning by doing*“

- selbstständiges Bearbeiten der Übungsaufgaben am Rechner
- der Rechner ist allerdings kein Tafelersatz

Der, die, das.
Wer, wie, was?
Wieso, weshalb, warum?
Wer nicht fragt, bleibt dumm!

Studien- und Prüfungsleistungen



Verwendbarkeit siehe UnivIS bzw. ihre Modulkataloge

- Wahl(pflicht)modul in diversen Studiengängen

Übungsschein bei erfolgreicher Bearbeitung aller Übungsaufgaben

Prüfung per Email an `wosch` den Termin vereinbaren

- 30-minütige mündliche Prüfung

Notengebung mündliche Prüfung + „Übungsbonus“ in Zweifelsfällen

Fragen...

42