

Systemprogrammierung

Lehrveranstaltungs-konzept

Wolfgang Schröder-Preikschat

Lehrstuhl Informatik 4

17. April 2012

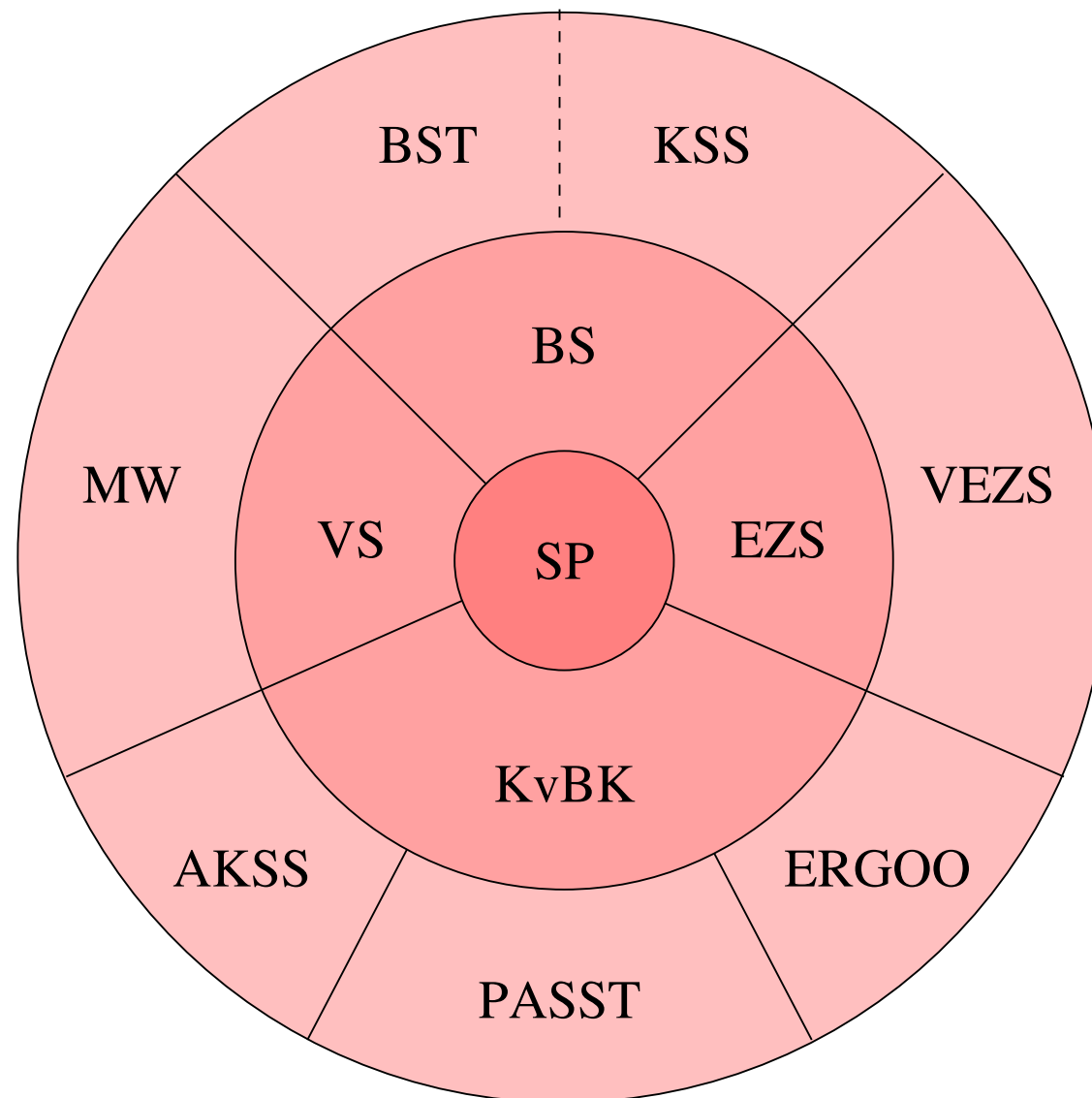
Gliederung

1 Konzept

- Lehrkanon
- Lehrziele
- Vorkenntnisse
- Veranstaltungsbetrieb
- Leistungsnachweise

2 Kontakt

Lehre@I4



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

Aufbau

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

Spezialisierung

BST	Betriebssystemtechnik	V/Ü
KSS	Konfigurierbare Systemesoftware	V/Ü
MW	Middleware (Rechnerwolken)	V/Ü
VEZS	Verlässliche Echtzeitsysteme	V/Ü
PASST	Praktikum angewandte Systemsoftwaretechnik	PR
AKSS	Ausgewählte Kapitel der Systemsoftware	HS
ERGOO	Erlangen Research Group on Distributed Objects and Operating Systems	KO

Integrierte Lehrveranstaltung: 10 ECTS $\sim$ 8 SWS

Systemprogrammierung $\sim$ geteiltes Modul (vgl. S. 9)

- i Systemprogrammierung 1 (SP1) $\mapsto$ Vorlesungsteile A und B 5 ECTS
- ii Systemprogrammierung 2 (SP2) $\mapsto$ Vorlesungsteil C 5 ECTS

- beide Modulkhälften werden **semesterweise zugleich** angeboten
 - sie sind jedoch verschiedenen „Generationen“ zugeordnet
 - „Studienbeginn WS“ einerseits und „Studienbeginn SS“ andererseits
 - SP1 im „2. Fachsemester“ || SP2 im „3. Fachsemester“
 - der Stoff von SP2 ist „kausal abhängig“ vom Stoff von SP1 (vgl. S. 11)

$\Rightarrow$ von der Belegung beider Hälften im selben Semester wird abgeraten

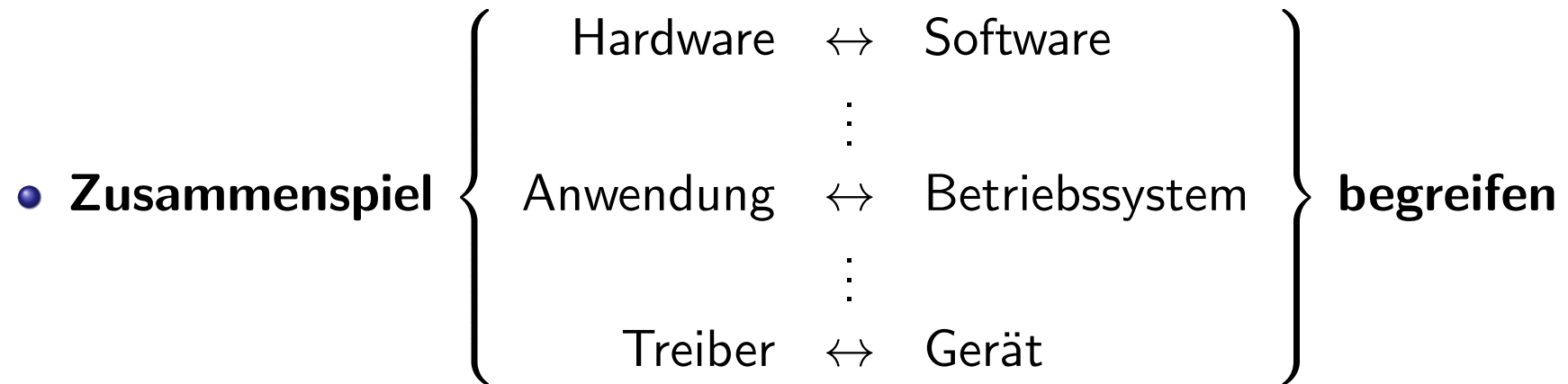
$$\text{SP}[1\&2] \equiv \left\{ \begin{array}{ll} \text{Vorlesung} & 2 \\ \text{Tafelübung} & 1 \\ \text{Rechnerübung} & 1 \end{array} \right\} \equiv 4 \times 2 = 8 \text{ SWS} = 6 \text{ Präsenzstunden}$$

Studiengänge und Zuordnung der Modulhälften

Abschluss	Studiengang	SP1	SP2
Bachelor	Informatik	×	×
	Informations- und Kommunikationstechnik	×	×
	Computational Engineering	×	×
	Wirtschaftsinformatik	×	×
	Mathematik, Nebenfach Informatik	?	
	Technomathematik (angerechnet als SPiC)	?	
Lehramt	Informatik, Gymnasium	×	×

Lernziele

Vorgänge innerhalb von Rechensystemen **ganzheitlich** verstehen



Grundzüge imperativer Systemprogrammierung (in C)

- im Kleinen praktizieren $\leadsto$ Dienstprogramme
- im Großen erfahren $\leadsto$ Betriebssysteme

Lehrinhalte

Vorlesung — Vorstellung und detaillierte Behandlung des Lehrstoffs

- Organisation (der Systemsoftware) von Rechensystemen
- Grundlagen von Betriebssystemen
- maschinennahe Programme

Übung — Vertiefung, Besprechung der Übungsaufgaben, Tafelübungen

- Systemprogrammierung in C
- Systemprogramme, -aufrufe, -funktionen von UNIX

Praktikum — Rechnerarbeit: Programmierung, Fehlersuche/-beseitigung

- UNIX (Linux), CLI (*shell*), GNU (gcc, gdb, make), vi. . .

Inhaltsüberblick: Kapitelzuordnung und -folge

I. Lehrveranstaltungsüberblick

Teil A ~ C-Programmierung

II. Einführung in C

III. Programm $\mapsto$ Prozess (UNIX)

Teil B ~ Grundlagen

IV. Einleitung

V. Rechnerorganisation

VI. Abstraktionen (UNIX)

VII. Betriebsarten

VIII. Zwischenbilanz SP1

XIV. Fragestunde SP1 & SP2

Teil C ~ Vertiefung

IX. Prozessverwaltung

- Einplanung

- Einlastung

X. Koordinierung

- Synchronisation

XI. Betriebsmittelverwaltung

XII. Speicherverwaltung

- Adressräume

- Arbeitsspeicher

XIII. Dateisysteme

- Speicherung

- Fehlererholung

Voraussetzungen zum Verständnis des Lehrstoffs

Obligatorisch

Algorithmen & Datenstrukturen $\mapsto$ Grundlagen der Programmierung

- Datentypen, Kontrollkonstrukte, Prozeduren
- statische und dynamische Datenstrukturen
- „Programmierung im Kleinen“

Wünschenswert, jedoch nicht zwingend erforderlich...

Technische Informatik $\mapsto$ Grundlagen der Rechnerorganisation

- „Von-Neumann-Architektur“
 - Operationsbefehle, Befehlsoperanden, Adressierungsarten
 - Unterbrechungssteuerung (Pegel kontra Flanke)
 - Assemblerprogrammierung
- CPU, DMA, FPU, IRQ, MCU, MMU, NMI, PIC, TLB

Abhängigkeiten zwischen den Vorlesungsteilen

Systemprogrammierung 1

- Teil A
- setzt grundlegende Programmierkenntnisse voraus
 - vermittelt Grundlagen der **Programmierung in C**
- Teil B
- setzt grundlegende Programmierkenntnisse in C voraus
 - vermittelt **Operationsprinzipien von Betriebssystemen**

Systemprogrammierung 2

- Teil C
- setzt Kenntnisse erwähnter Operationsprinzipien voraus
 - vermittelt die **interne Funktionsweise** von Betriebssystemen

Erlangung der benötigten Vorkenntnisse

- (i) durch Vorlesungsteilnahme
- empfohlene sequentielle Belegung der Vorlesungsteile
- (ii) durch Lehrbuchlektüre, aus anderen Lehrveranstaltungen, ...

Vorlesungsbetrieb und Lehrmaterialien

Vorlesungstermine ab KW 16

SP1	Donnerstag	08:00 – 09:30	00.915
SP2	Montag	14:15 – 15:45	H4

C/UNIX-Einführung: SP1

- 4 Vorlesungstermine + 1
- ab heute: KW 16 – 18

Ausfalltermine

- SP1** ● Christi Himmelfahrt (17.05.), Fronleichnam (07.06.)
- SP2** ● Pfingstmontag (28.05.)

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

- ① Austeilung von Folienkopien vor der Vorlesung
- ② www4.informatik.uni-erlangen.de/Lehre/SS12/V_SP

Fachbegriffe der Informatik (Deutsch ↔ Englisch)

- www.babylonia.ork.uk
- www.inf.fu-berlin.de/inst/ag-ss/montagswort
- www.aktionlebendigesdeutsch.de

Ergänzende Literatur (nicht nur) zum Vorlesungsstoff

- [1] KERNIGHAN, B. W. ; RITCHIE, D. M.:
The C Programming Language.
Prentice-Hall. Inc., 1988. –
ISBN 0–131–10362–8
- [2] NEHMER, J. ; STURM, P. :
Systemsoftware: Grundlagen moderner Betriebssysteme.
dpunkt.Verlag GmbH, 2001. –
ISBN 3–898–64115–5
- [3] SCHRÖDER-PREIKSCHAT, W. :
Betriebssysteme — Grundlagen, Entwurf, Implementierung.
Springer, 2012 (geplant; Manuskriptauszüge erhältlich)
- [4] SILBERSCHATZ, A. ; GALVIN, P. B. ; GAGNE, G. :
Operating System Concepts.
John Wiley & Sons, Inc., 2001. –
ISBN 0–471–41743–2
- [5] TANENBAUM, A. S.:
Structured Computer Organization.
Prentice-Hall, Inc., 1979. –
443 S. –
ISBN 0–130–95990–1

Übungsbetrieb

Tafelübung: Anmeldung mit **WAFFEL**¹ über die Startseite von SP

- die Periode der Übungswoche startet mit zweiter Vorlesungswoche
 - `www4.informatik.uni-erlangen.de/Lehre/SS12/V_SP`
- Übungsplätze werden FCFS² vergeben
 - unterbelegte Termine werden ggf. gestrichen
 - überbelegte Termine erhalten ggf. mehr Ressourcen
- Übungsaufgaben sind teilweise in Zweiergruppen zu bearbeiten
 - Übungspartner müssen für dieselbe Tafelübung angemeldet sein

Rechnerübung: Anmeldung ist nicht erforderlich

- die Teilnahme ist optional, je nach Bedarf
 - es können auch mehrere Termine pro Woche wahrgenommen werden
- Übungsleiter stehen bei Fragen zur Verfügung

¹Abk. für Webanmeldefrickelformular Enterprise Logic

²Abk. für engl. *first come, first served*, Einplanung nach Ankunftszeit

Bedeutung von Tafel- und Rechnerübungen

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

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

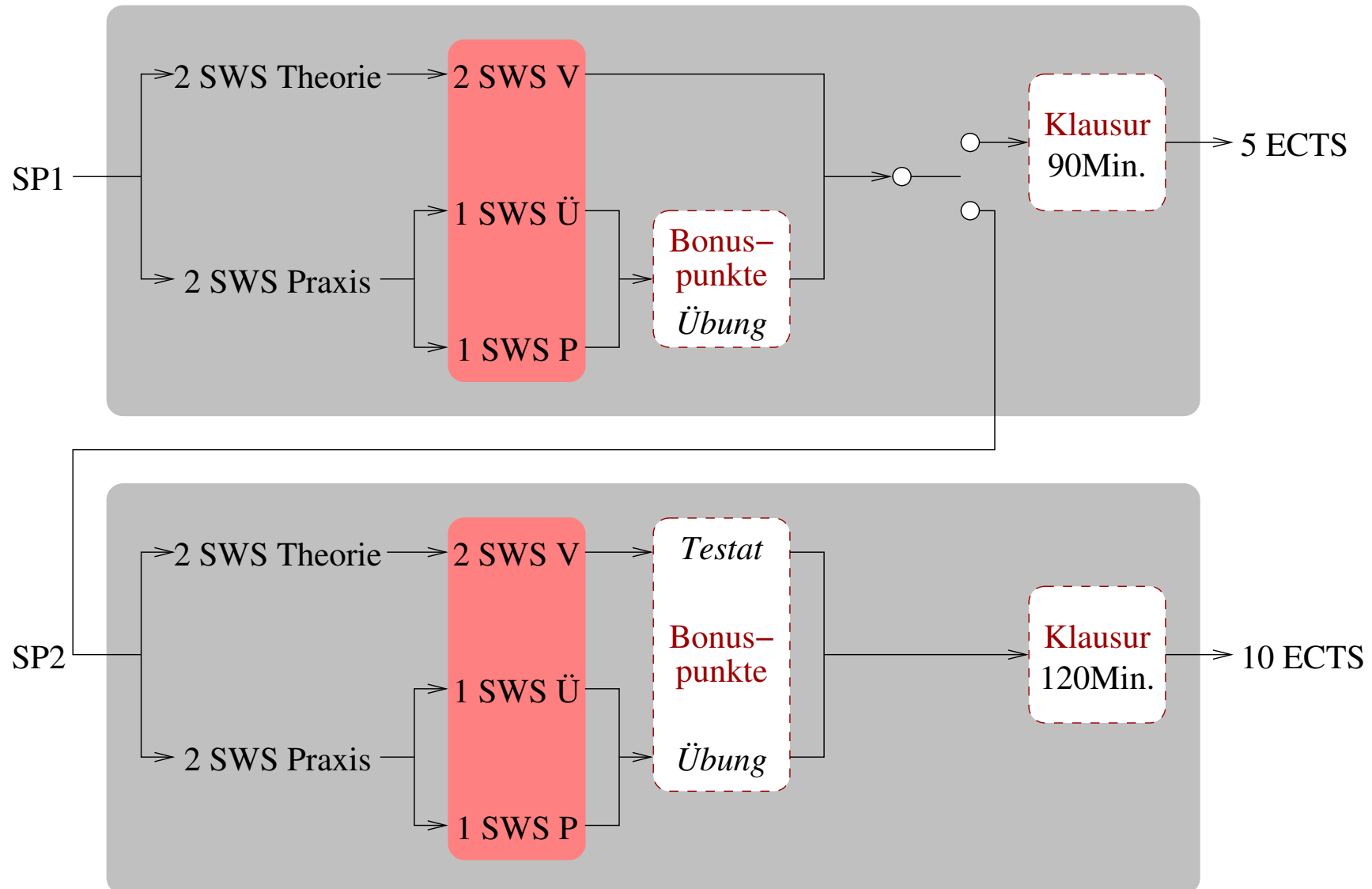
Rechnerübungen $\leadsto$ „*learning by doing*“

- selbständiges Bearbeiten der Übungsaufgaben am Rechner
- Hilfestellung beim Umgang mit den Entwicklungswerkzeugen
- der Rechner ist **kein Tafelersatz**, die Betreuung verläuft eher passiv

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



Studien- und Prüfungsleistungen



Testat und Schein

Testat $\sim$ Ex $\mapsto$ **Miniklausur**: optional, **am Anfang von Teil C** (SP2)

- geprüft wird Stoff von Vorlesung und Übung, 30 Minuten
 - Fragen zu **Teil A** und **Teil B** der Vorlesung
 - Trockenübung in der Programmiersprache C
- mit Anteilen von Mehrfachauswahl (engl. *multiple choice*)

Schein: erfolgreiche Teilnahme an Tafel- und Rechnerübungen

- 50 % der Punkte aus „Übungsaufgaben + Ex“ sollten erreicht werden
- Punkte darüberhinaus gehen in die Bonuspunktberechnung ein

Bonuspunkte für die Klausur ansammeln

- max. 10 % der Klausurpunkte erreichbar $\approx$ 12 Punkte
- Erfahrungswert: $\approx$ 5.5 Klausurpunkte pro Notenstufe 0.3
- Verbesserung der Klausurnote um bis zu zwei Stufen möglich

Bonuspunkte und Klausur

Bonuspunkte: nur auf Basis des Übungsscheins **des laufenden SP-Moduls**

- beeinflussen die angelegte Notenskala nicht, werden jedoch bei bestandenen Klausuren auf die Klausurpunkte aufaddiert
- können die Note einer bestandenen Klausur verbessern, nicht jedoch den Ausschlag zum Bestehen der Klausur geben
- Erreichen der Bestehensgrenze muss also immer mit regulären Klausurpunkten erfolgen

Klausur: Termin steht noch nicht fest, Anfang vorlesungsfreie Zeit

SP1 • Struktur analog Testat (S. 17), jedoch 90 Minuten Dauer

SP • Struktur analog SP1, jedoch 120 Minuten Dauer
• zusätzlich Fragen zu **Teil C** der Vorlesung

Präsenz und **aktive Mitarbeit** machen die Klausur „leicht“

- Programme im Team entwickeln, aber selbst zum Laufen bringen

Gliederung

1 Konzept

- Lehrkanon
- Lehrziele
- Vorkenntnisse
- Veranstaltungsbetrieb
- Leistungsnachweise

2 Kontakt

`www4.informatik.uni-erlangen.de/*`



Dozenten

- Jürgen Kleinöder (~jklein)
- Wolfgang Schröder-Preikschat (~wosch)

Mitarbeiter

- Christoph Erhardt (~erhardt)
- Jens Schedel (~schedel)
- Isabella Stilkerich (~isa)

Tutoren

- | | | |
|-------------------------|---------------------|-------------------------|
| • Gabor Czilli | • Demian Kellermann | • Valentin Rothberg |
| • Hans-Peter Deifel | • Johannes Knoedtel | • Andreas Ruprecht |
| • Michael Fiedler | • Philip Kranz | • Tobias Sammet |
| • Michael Fularczyk | • Clemens Lang | • Tobias Scharpff |
| • Andreas Grillenberger | • Daniel Laffling | • Florian Schmaus |
| • Ralf Hackner | • Florian Meier | • Markus Siko |
| • Stefan Hengelein | • Simon Ruderich | • Matthias Utzschneider |

Fragen...

42