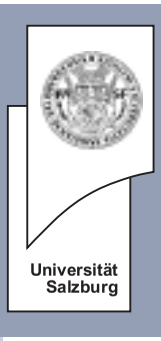



# Motivation & Inhaltsübersicht

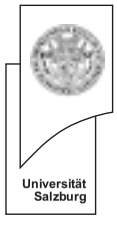
## Vorlesung Verteilte Systeme
## Wintersemester 2003/04
## Universität Salzburg

**O.Univ.-Prof. Dipl.-Ing. Dr.techn. Wolfgang Pree**




# Context

- Verteilte Programmierung im Überblick
- Konzeption der LVA
  Verteilte Systeme/Distributed Systems

# Verteilte Programmierung im Überblick




## Definition

- Ein verteiltes (Datenverarbeitungs-)System besteht aus mehreren autonomen Prozessor-Speicher-Systemen, die mittels Datenaustausch kooperieren (Mühlhäuser 2002; Informatik-Handbuch, Hanser)
  - häufig sind die Prozessor-Speicher-Systeme vollständige Rechner
  - Mehrprozessorsysteme mit gemeinsamem Speicher sind KEINE verteilten Systeme
  - ebenso eng gekoppelte Mehrrechnersysteme, die die durch zentrale oder verteilte Koordination die Autonomie der Teilsysteme einschränken

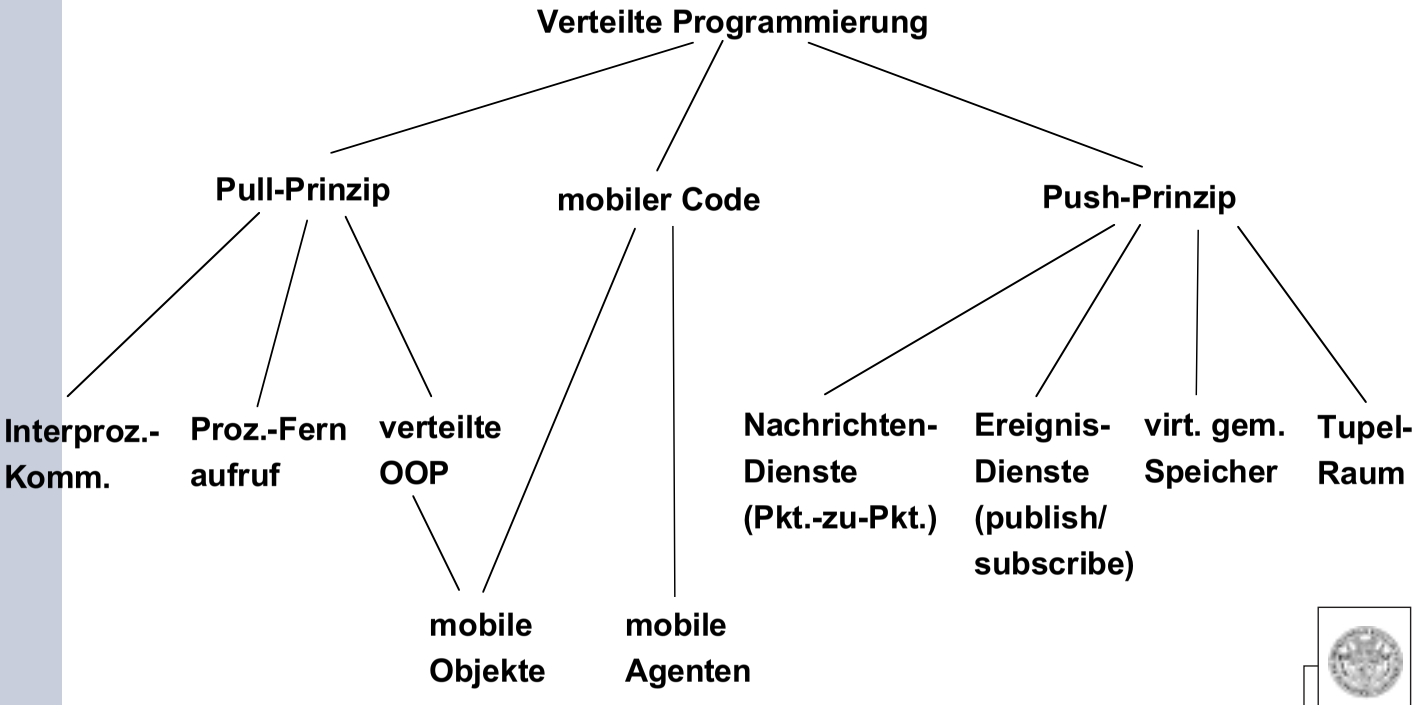
# Charakteristika

- Fehlen einer aktuellen konsistenten Sicht auf das Gesamtsystem durch Autonomie, Parallelität und räumliche Entfernung der Knoten
- Heterogenität von Rechnerarchitekturen, Leistungskenngrößen, Kommunikationsprotokollen, Betriebssystemen und Benutzerschnittstellen




SOFTWARE RESEARCH LAB
Universität Salzburg

# Ausprägungen verteilter Programmierung







SOFTWARE RESEARCH LAB
Universität Salzburg

# Beispiel für verteilte Programmierung

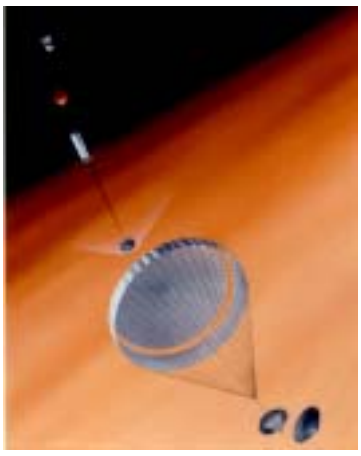
# Automatisierte Entwicklung von verteilter Software zur Steuerung von Echtzeitsystemen




SOFTWARE RESEARCH LAB

Universität Salzburg

# Beispiele für (harte) Echtzeit-(Steuer-) Systeme



Satelliten-, Flugzeugsteuerungen

ABS, X-by-Wire, etc.




SOFTWARE RESEARCH LAB

Universität Salzburg

# Charakteristika

„Olymp der Software-Entwicklung“:

- muß funktional korrekt sein
- Einhalten von Zeitlimits ist sicherheitskritisch
- typischerweise verteilte Architektur (zB wegen Fault-Tolerance): Probleme von Concurrency, Deadlocks, etc.

- eingeschränkte Ressourcen (Speicher, Prozessor)
- „Rebooting“ ist keine Option!






# internationale Kooperationsprojekte

## Partner:

- **Magna-Steyr (Graz)**
  Univ. Salzburg
- **European Space Agency (ESA):**
  Univ. Konstanz/Salzburg
- **Boeing, U.S. Dept. of Defense:**
  UC Berkeley

# Beispiel: Helicopter Control System (I)

Henzinger, Kirsch, Pree, Sanvido (UC Berkeley)

Schaufelberger, Wirth (ETH Zürich)

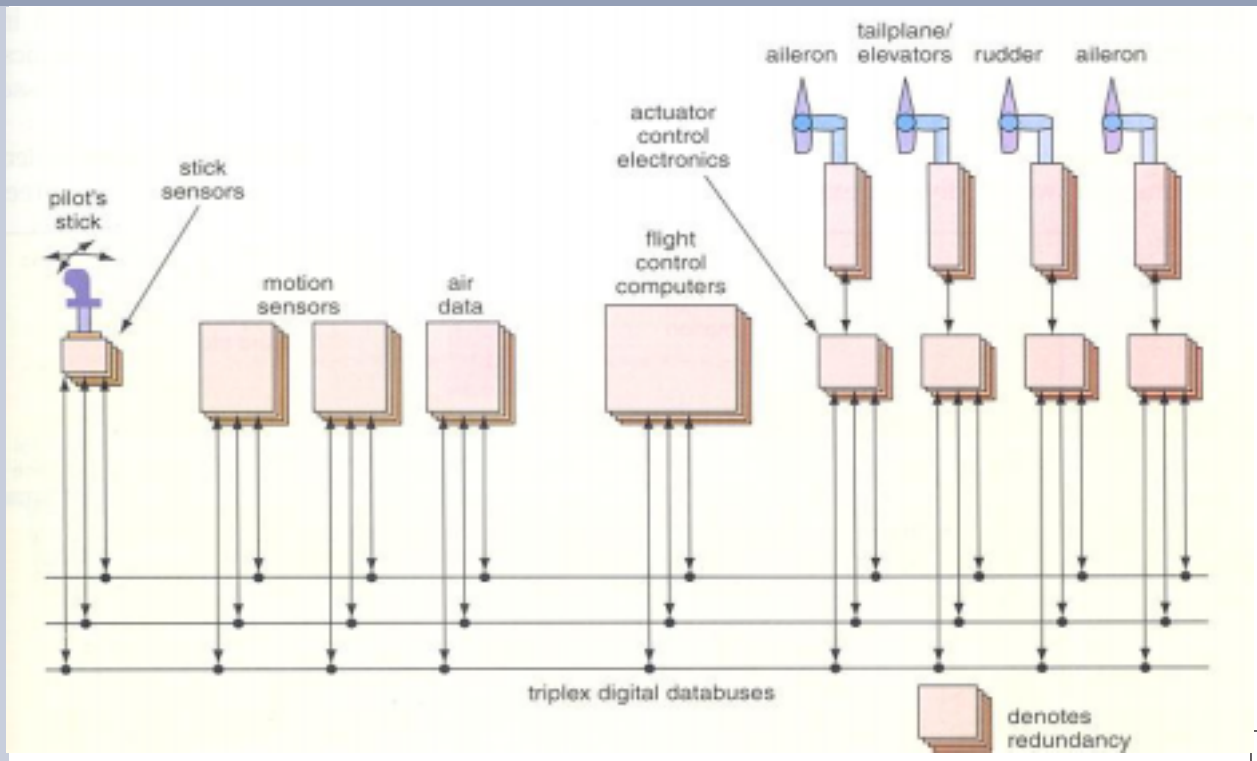




SOFTWARE RESEARCH LAB

Universität Salzburg

# Beispiel: Helicopter Control System (II)



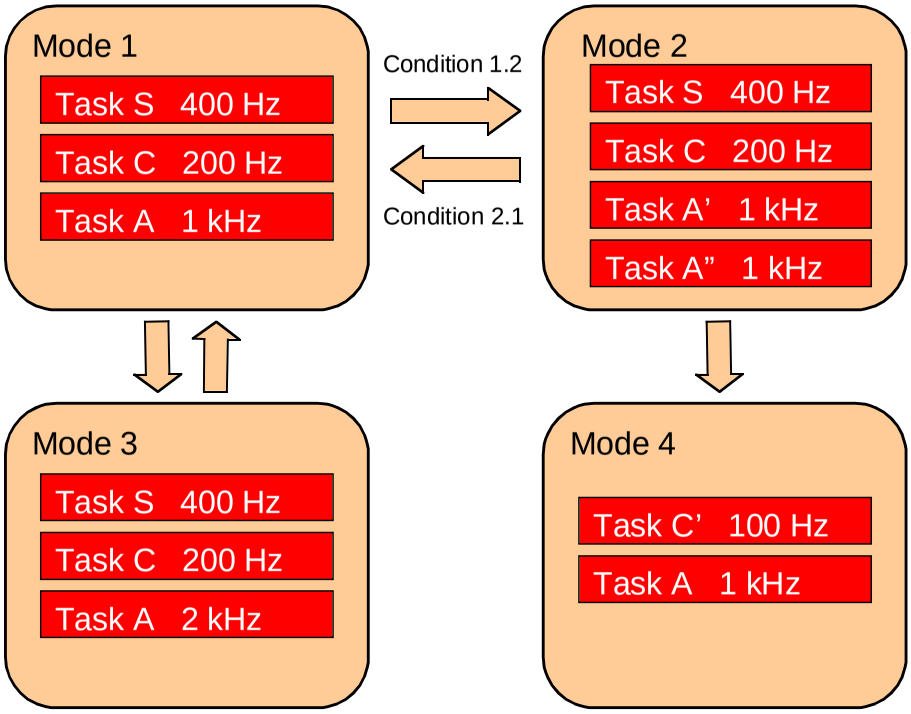





SOFTWARE RESEARCH LAB

Universität Salzburg

# „höhere Programmiersprache“ für Zeitaspekte



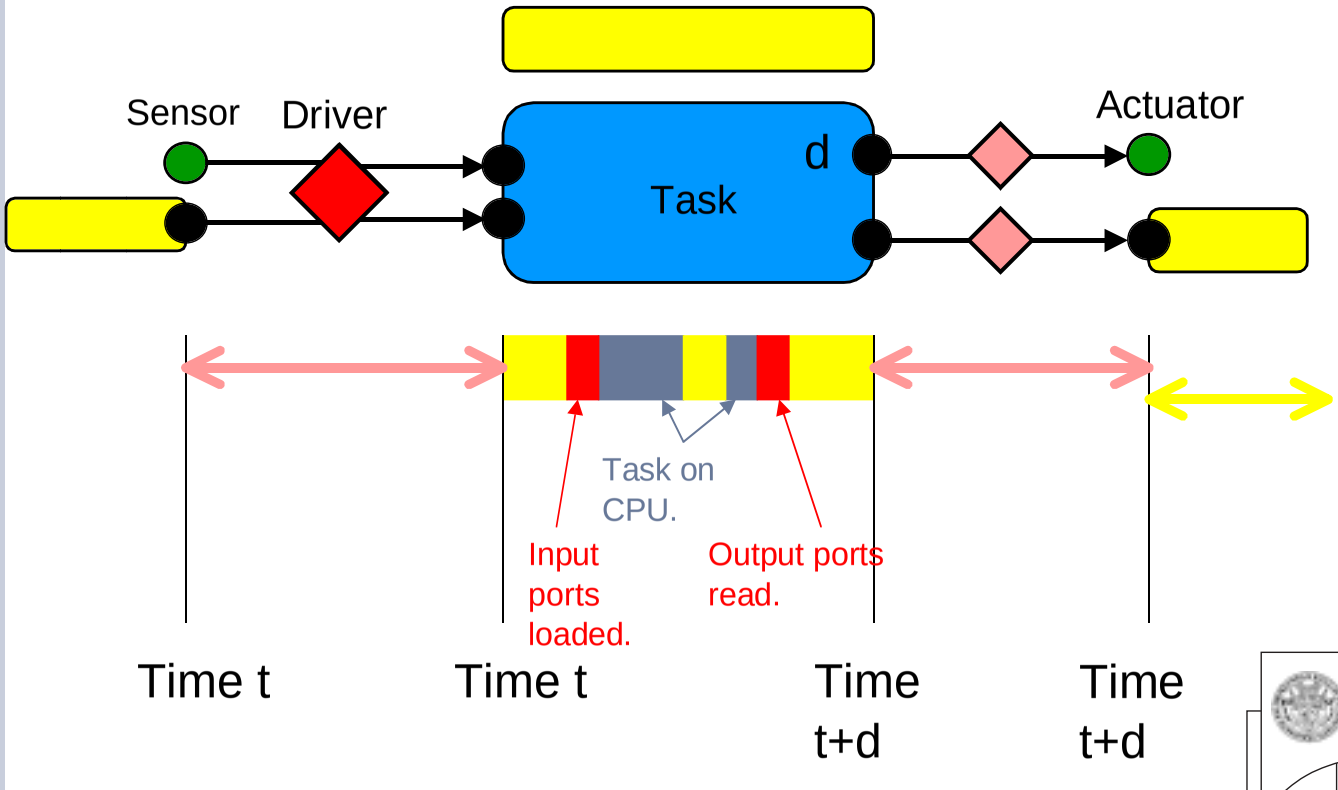

**=> spezif. HW-Plattform wird irrelevant**




SOFTWARE RESEARCH LAB

Universität Salzburg

# Timed Model: time determinism => value determinism + compositionality







SOFTWARE RESEARCH LAB

Universität Salzburg

# Konzeption der Lehrveranstaltung Verteilte Systeme




## Ziele

- Konzepte und Begriffe kennen und verstehen
- ingenieurmäßiges Herangehen an verteilte Software-Entwicklung lernen

*Example isn‘t another way to teach, it is the only way to teach*

Albert Einstein




SOFTWARE RESEARCH LAB

Universität Salzburg

# Verteilte Systeme (primär nach Tanenbaum)

- **Kommunikation**
- **Prozesse**
- **Namensgebung**
- **Synchronisation**
- **Konsistenz und Replizierung**
- **Fehlertoleranz**
- **Sicherheit**
- **Verteilte Services/Objekte**
- **Verteilte Dateisysteme**
- **Verteilte Dokument-basierte Systeme**




SOFTWARE RESEARCH LAB

Universität Salzburg