

Motivation & Inhaltsübersicht

Vorlesung Verteilte Systeme Wintersemester 2002/03 Universität Salzburg

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

© Copyright Wolfgang Pree, All Rights Reserved

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.



Context

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

Verteilte Programmierung im Überblick

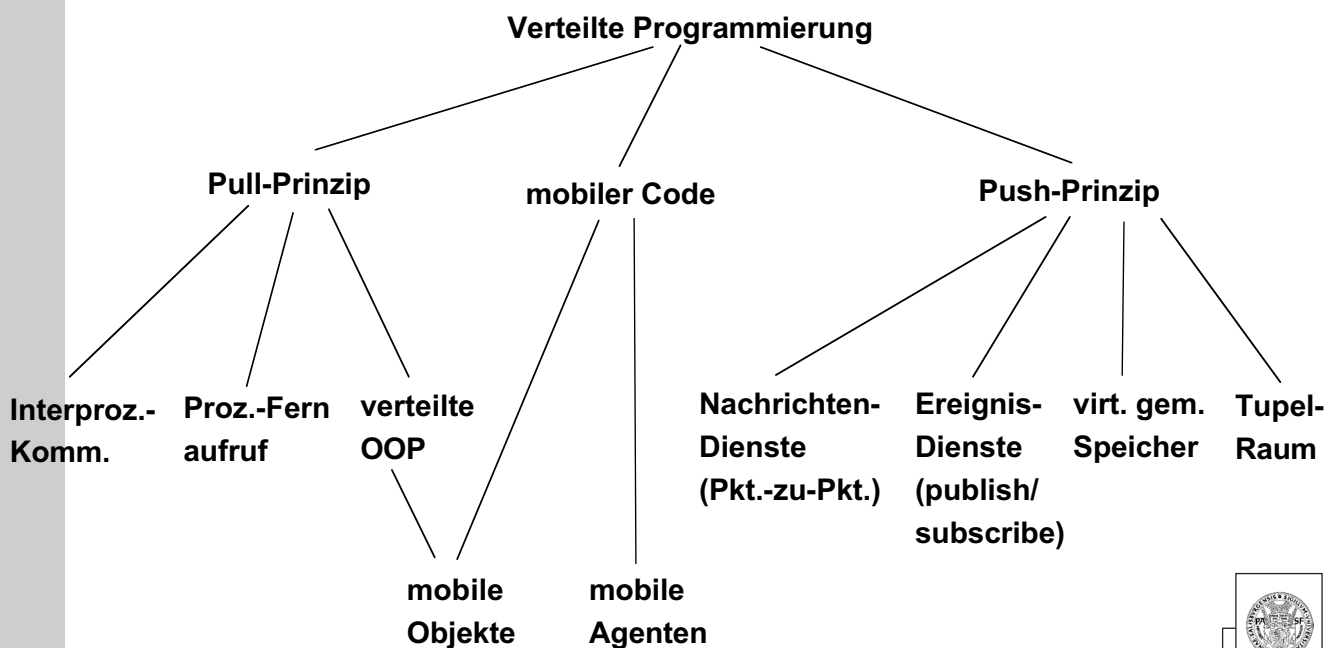
Definition

- 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

Ausprägungen verteilter Programmierung



Beispiel für verteilte Programmierung

Automatisierte Entwicklung von verteilter Software zur Steuerung von Echtzeitsystemen

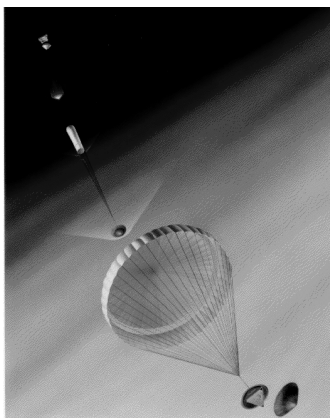
7

© 2002, W. Pree

SOFTWARE RESEARCH LAB



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



Satelliten-, Flugzeugsteuerungen

ABS, X-by-Wire, etc.



8

© 2002, W. Pree

SOFTWARE RESEARCH LAB



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

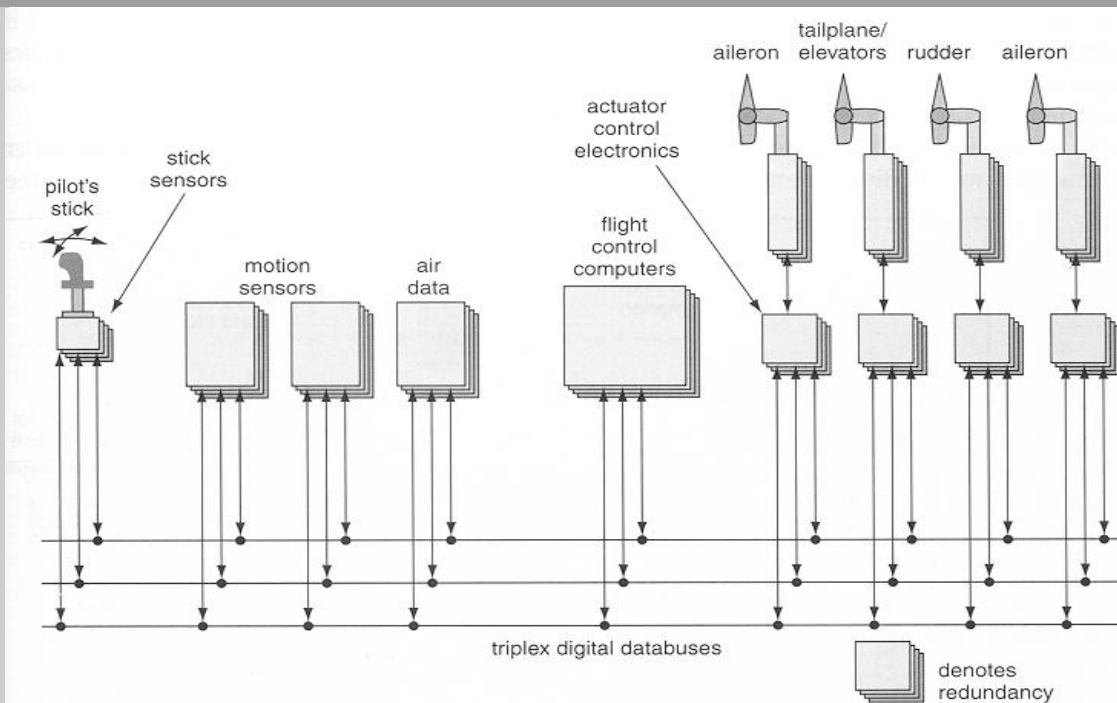
Beispiel: Helicopter Control System (I)



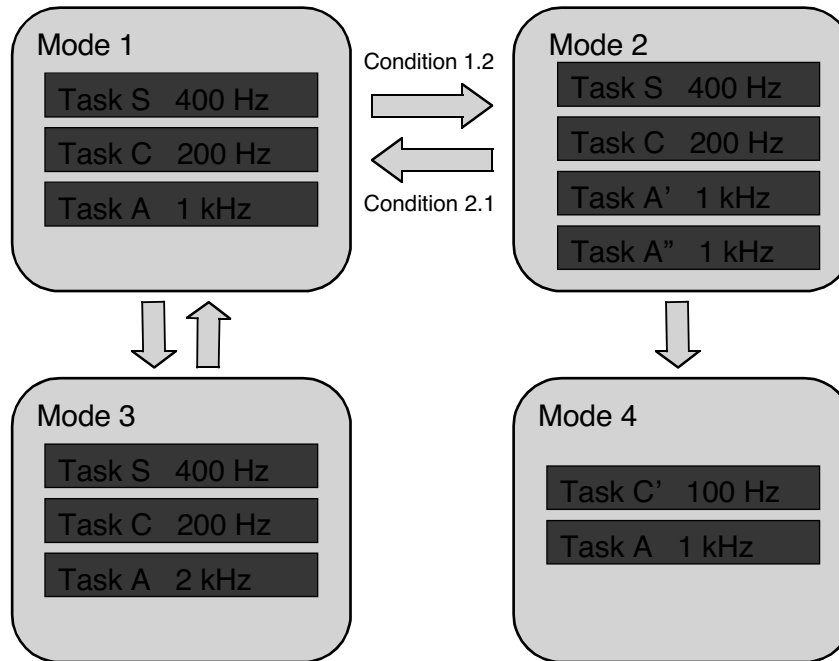
Henzinger, Kirsch, Pree, Sanvido (UC Berkeley)
Schaufelberger, Wirth (ETH Zürich)



Beispiel: Helicopter Control System (II)

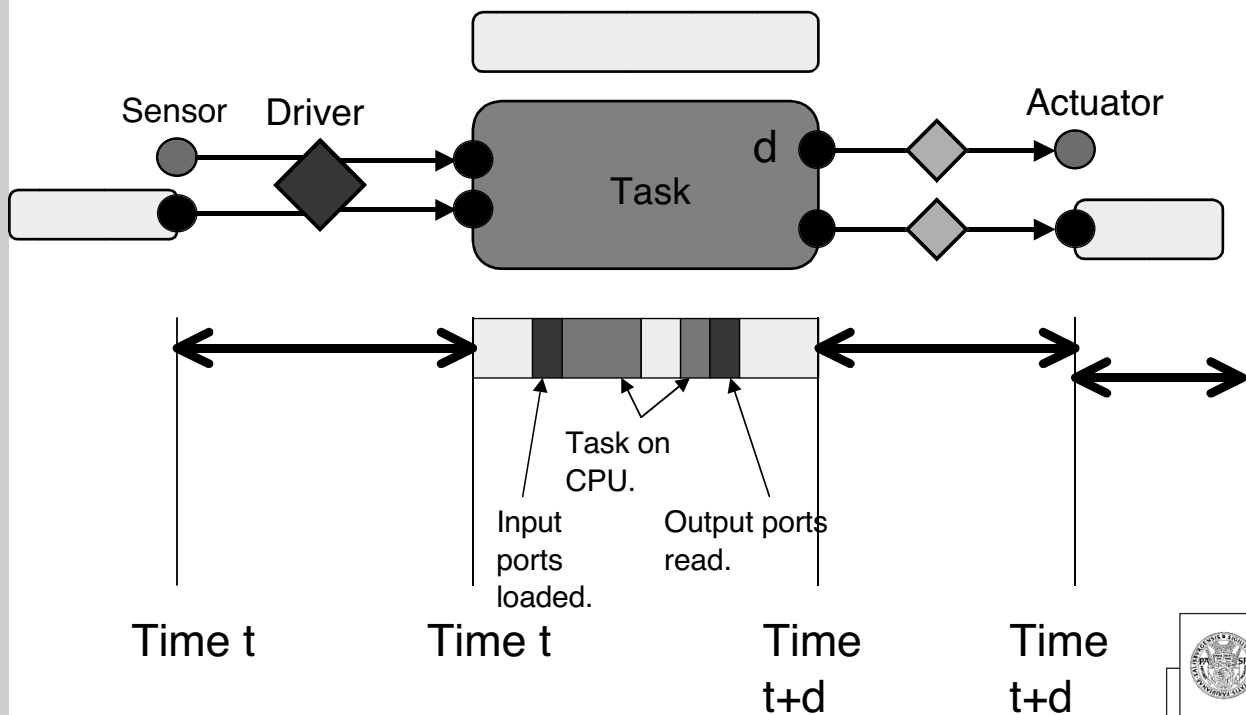


„höhere Programmiersprache“ für Zeitaspekte



=> spezif. HW-Plattform wird irrelevant

Timed Model: time determinism => value determinism + compositionality



Konzeption der Lehrveranstaltung Verteilte Systeme

15

© 2002, W. Pree

SOFTWARE
RESEARCH LAB



Ziele

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

16

© 2002, W. Pree

SOFTWARE
RESEARCH LAB



Example isn't another way to teach, it is the only way to teach

Albert Einstein

Verteilte Systeme (primär nach Tanenbaum)

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