

# K 1 Einführung in die Prozessdatenverarbeitung(PDV) und Begriffe

## 1.1 Einführung und Geschichte der PDV

*Geschichtlicher Überblick im Zeitraffer bis heute*

*Hinweis: Betrachtung nur der RZ-Geschichte ohne SPS*

### Geschichte der PDV

#### 1. Beginn in den 50er Jahren:

- Erste Realzeitrechner (PR) - Prozessrechner (PR)- zur PDV Mitte bis Ende der 50er Jahre
- Entstanden aus kommerziellen und wissenschaftlichen Rechnern
- Besonders an PR: Erweitert um Komponenten zur Prozesskommunikation
- ► damals: Kosten 1-4 Mio. DM, ► Großindustrieeinsatz
- Erste Einsatzgebiete: chemische, petrochemische und Stahlindustrie zur Überwachung von Anlagen
- Probleme:
  - PR sehr teuer, also wurde versucht viele Funktionen parallel zu implementieren ► Überforderung der Leistungsfähigkeit
  - mangelnde Integrationsfähigkeit mit bestehenden regelungs- und steuerungstechnischen Komponenten

#### 2. Mitte 60er Jahre:

- Aufkommen von Kleinrechnern als PR
- Start der systematischen und stürmischen Entwicklung der Prozessleittechnik
- Meilensteine:
  - Konzepte zur Modularisierung und hierarchischen Aufteilung der Prozessleit- und Regelaufgaben
  - Auf der prozessnahen untersten Ebene Einsatz von Kleinrechnern, obere Ebene Großrechner
- Probleme:
  - brauchbare Realzeitsprachen und Betriebssysteme nur für große PR
  - Assemblerprogrammierung und "aus dem Vollen" entwickeln dominiert



#### 4. Heute:

- Mikrorechner:= Mikroprozessor + Speicher + Peripherie ist der typische PR heute => der heutige PR unterscheidet sich von klassischen EDV-Rechnern nur noch geringfügig. d.h.: EDV hat sich in Richtung PR entwickelt.
- "klein" PR zur Anlagenautomatisierung vergleichbar zu 60er Jahre: Kosten heute deutlich unter 10 T€, typisch 2-10 T€, => Technik: Standard-32bit/64bit Micro als Kern, hohe Taktraten (3 Ghz und mehr möglich) , 0,5 -16 Gbyte RAM, Festplatte >>=500 GB, Graphiksubsystem, Maus, Tastatur, vernetzt über (Gb)LAN (Ethernet), Ankopplung der Peripherie über Feldbus / Sensor-Aktorbus oder Sternverkabelung
- Typische Kennzeichen von PR der "untersten" prozessnahen Ebene im Fabrikeinsatz:
  - CPU: Mikroprozessor (16) 32, 64 bit, hohe Taktraten
  - Arbeitsspeicher: dynamische RAM im Ausbau 2(und mehr) Gbyte
  - Universalbus : PCIe PC-BUS (VME-BUS) ....
  - Standardperipherie:
    - LAN-Ankopplung 10/100 Mbit 1/10Gbit (TCP/IP) zu übergeordnetem Leitrechner
    - lokaler Massenspeicher vorhanden z.B.: Platte 500-... GByte
    - entweder (sinkende Tendenz): Peripherieansteuerung über Einsteckkarten und Sternverkabelung zum Prozess
    - oder aktuell: Sensor/Aktor-Bus, Feldbus (PROFINET, CANOpen, EtherCAT, ...)
  - oft: Standardbürorechnerausführung eingebaut im klimatisiertem und staub- und spritzwassergeschützten 19"Schrank vor Ort in Prozessnähe in der Fabrikhalle
- **Wachstumsmarkt:** wirtschaftlicher Einsatz PR bis hinunter zu der Familie der "embedded Systems"

- statt komplexes PR komplettes Automatisierungs-/Steuerungssystem auf einer Platine (embeddedRZ) ► Bestreben eine "Ein-Chip-Lösung" zu erreichen. i.d.R. keine Festplatte, Tastatur, Bildschirm ...
- ► ASICs, FPGAs, SystemOnAChip(SOC), Microcontroller 4/8/16/32/64 Bit, der beinhaltet auf Chip: CPU, FLASHRom, RAM, digitale I/O, Timer, A/D, D/A, PWM, Kommunikationsbausteine... ► gute EMV-Eigenschaften, da oftmals nur noch Leistungselektronik außerhalb des Microcontrollers auf der Platine zu finden ist ► keine externen ROM/RAM Bausteine!!
- typische Hardwarekosten im 0,5-100 € Bereich, 8-32(64)bit Microcontroller mit ROM/RAM -teilw. extra. Einsatz von ASICs in denen alle Sonderfunktionen auf einem Chip integriert sind.
  - Stückzahlen min. 0,5-x Mio. pro Sorte pro Jahr
- embeddedRZ innerhalb von Großserienhaushaltsgeräten z.B.: Waschmaschine, Fotoapparat, Videorecorder, CD/DVD-Player etc.
  - Kosten im Euro Bereich, 8-bit/16-bit/ 32-bit Microcontroller mit ROM/RAM auf dem Chip, Mio.-Stückzahlen/Jahr.
- embeddedRZ innerhalb Automobilelektronik u.a. sog. Steuergeräte w.z.B. Motronik, Airbag, Bordcomputer....
  - Steuergeräte hochgradig vernetzt. Datenaustausch während der Fahrt. Einsatz von Feldbussen: z.B. CAN (► Erfinder Bosch), FlexRay
  - SOCs mit Microcontroller 8/16/32 Bit, 4-128 K RAM on-Chip A/D, D/A, I/O , Feldbus, Spezialperipherie , RAM und FlashROM.
  - Randbedingungen:
    - höchste Zuverlässigkeit bei widrigen Umweltbedingungen w.z.B. hohe Temperaturschwankungen, Vibration und Nässe
    - EMV ► Abstrahlung und vor allem Unempfindlich gegen Einstrahlungen. ESD (Elektrostatikfest)

- Sehr kostensensitiv ► Einsparung von "Centbeträgen" an der Serienhardware, daher individuelle Entwicklungen
- Standardisierung von Echtzeit-Betriebssystemeigenschaften und Vernetzungsfunktionen ► OSEK, AUTOSAR um die Entwicklungszeiten und -kosten zu minimieren.
- embeddedRZ innerhalb Telekommunikation
  - SOC Handy's GSM ein Chip Lösungen/ UMTS drei Chip Lösungen (noch ohne Betriebssystem, Generation UMTS aber teilweise mit Realzeitbetriebssystem) kosten zwischen 8-16 Euro/Chip
  - SOC's Netzwerkkomponenten: Router, Switches....
- embeddedRZ innerhalb Multimedia
  - Set-Top-Boxen des digitalen Fernsehens
  - Video/DVD/Blue-ray-Player/Streaming-Boxes.....
- **Heute immer noch: statt Hardwarekostenproblem nun Softwarekosten- und Testkostenproblem:**
  - Es gibt mittlerweile mehr als genügend leistungsfähige und kostengünstige Echtzeit-Betriebssysteme (>200 verschiedene) und (Realzeit)-Programmiersprachen für PR
  - **Aber:**
    - Seit Mitte der 80er Jahre steigen die Kosten der Erstellung von Software bei sinkender Hardwarekostentendenz ► typische PR-Kosten eines mittleren Automatisierungsprojektes: 50 T€ Software und 10 T€ Hardware
    - Der all-round Automatisierer der 80er musste sich anwendungsnah spezialisieren um quasi-Standard-Anwendungs-Software erstellen zu können damit die Kosten reduziert werden konnten.
    - seit Anfang 90er bis heute verlangt der Anwender (► der Kunde der Automatisierer) nach Softwareengineering-Tools und Softwarehilfsmittel die es ihm gestattet **selbst und wenn möglich ohne Programmierkenntnisse** Automatisierungssysteme zu projektieren, konfigurieren, in Betrieb zu setzen aber vor allem selbst zu pflegen.

- Herausforderung an den Automatisierer: Die zugestandenen Entwicklungszeiten werden immer kürzer, die Entwicklungskosten müssen permanent reduziert werden und die Automatisierungsaufgaben werden immer komplexer.
- **Ursache:**
  - der Anwender kennt seinen Prozess selbst am genauesten, weiß ihn bestens zu optimieren
  - die Automatisierung findet immer mehr schritthaltend mit der Optimierung und Produktverbesserung des Gesamtprozesses statt ► Know-how nur beim Anwender der das Produkt herstellt,
  - ► Es dauert zu lange bis jemand das erklärt bekommen hat.
- **Wirkung:**
  - Der typische Anbieter von Automatisierungstechnik wird zukünftig der Standard-Hardware/Softwaretoolslieferant und hilft beim Projektieren des Erstsystems ► Zwang zur **ISO-Standardisierung**, um Mehrfach verkaufen zu können, denn die Entwicklungskosten zahlt längst nicht mehr nur 'ein' Kunde.
  - Der Projektingenieur beim Anwender projiziert, realisiert und wartet die Automatisierungssysteme selbst .
- **Trend:**
  - Seit spätestens der Jahrtausendwende zeichnet sich ab, dass die Multimediaanwendungen für Jeden und Anwendungen der Telekommunikation, w.z.B Handy, den Entwicklungsfortschritt im Echtzeitbereich extrem beeinflussen.
  - Es existieren mittlerweile ausgefeilte (teure) Toolketten, die es gestatten komplette Applikationslösungen integriert vom Entwurf über die Implementierung bis hin zum Test in einem 'Guss' zu entwickeln. (► Rhapsody von IBM Rational...).

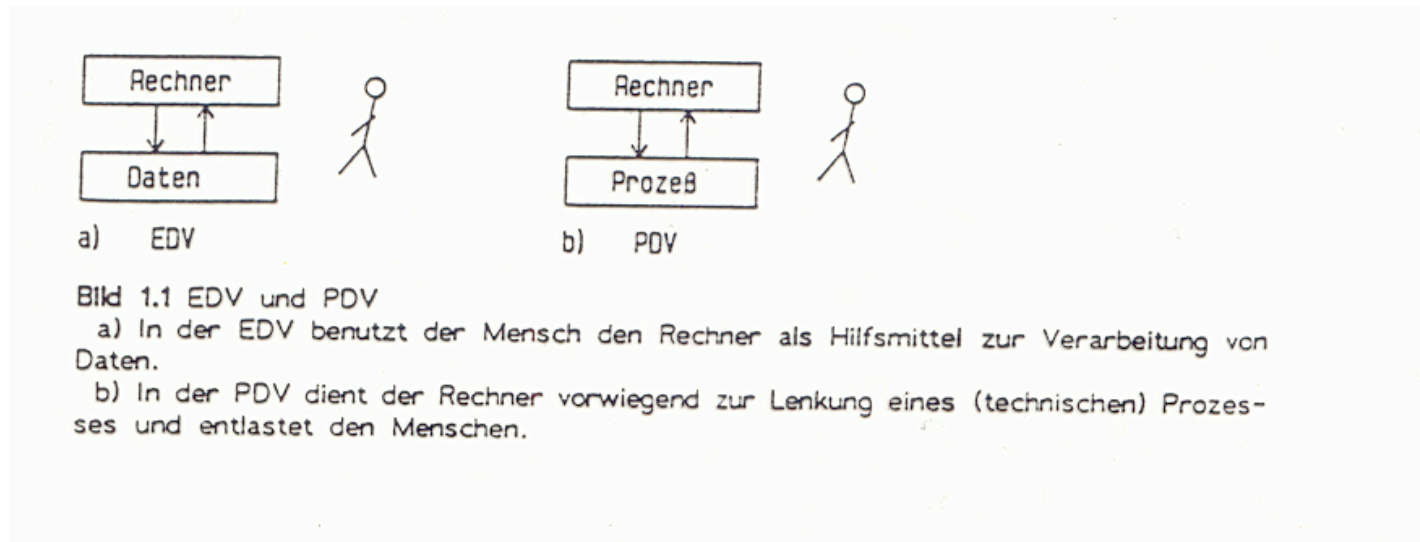
*Fazit:*

*Der RZ von heute unterscheidet sich in der Hardware von der Standard-EDV Rechnern nur in wenigen Punkten.*

*Heute ist die Softwareerstellung und der Nachweis der Funktionstüchtigkeit der Hauptkostenfaktor ► Es ergibt sich deshalb die Notwendigkeit zur Standardisierung von Softwarekomponenten und -systemen mit dem Ziel -ähnlich der Hardware- zu kommerziell verfügbaren preiswerten mehrfach einsetzbaren Bausteinen zu gelangen. Diese werden nach dem Baukastenprinzip nur noch "zusammengestopelt" -gegebenenfalls durch Nichtprogrammierer!! und/oder prozessnahen Leuten-.*

## 1.1.1 Ziel der Prozessdatentechnik (PDV)-Vorlesung, oder was ist Prozessdatentechnik?

- Es gibt kein homogenes klar abgegrenztes Gebiet der Prozessdatentechnik (PDV).
- Die PDV ist ein Spezialgebiet der Datenverarbeitung (DV) , bei der als Besonderheit die **Lenkung** eines **technischen Prozesses** im Vordergrund steht



- Der technische Prozess erweist sich als Quelle und Senke von Daten.
- Dabei gilt: Die Erfassung und die Erzeugung von zeitlichen Abfolgen von Daten und die Einhaltung von strikt vorgegebenen Bearbeitungszeiten ist hierbei besonders wichtig:
  - ▶ Das wird auch als **Realzeitbedingungen** bezeichnet.  
*Beispiel: Abfüllung einer Bierflasche in Abfüllanlage: Nicht zu früh und nicht zu spät öffnen des Hahns*

**Die Einhaltung der zeitlichen Prozessgegebenen Abfolgen definiert die Leistungsanforderungen, den Umfang und Ausprägung der PDV mit der Maßgabe die Verarbeitung schritthaltend mit dem Prozess und die Einhaltung der gegebenen Anforderungen der Betriebssicherheit zu gewährleisten.**

1. Schritt: Wie kann man den technischen **Prozess** näher fassen? ► alles das in Kapitel 1 der Vorlesung

- Existieren gemeinsame Merkmale von unterschiedlichen Prozessen ► gleiche Merkmale, gleiche technischen Probleme zu lösen?
  - Wie lassen sich Prozesse beschreiben? ► SDL, SADT, Zustandereignisdiagramme....
  - Stichworte: Zuverlässigkeit, Fehlertoleranz, Sicherheit
- 
- Die **Steuerung** des Prozesses erfolgt in PDV durch einen **Prozessrechner(PR)** mit **Realzeitbetriebssystem**.
    - **'embedded' Systeme und Maschinensteuerungen** ► keine SPSEN, eigene Vorlesung
    - **keine Prozessperipherie** ► Vorlesung: Messtechnik, Sensortechnik
    - **keine Microcontroller** ► Vorlesung: Microcontrollervorlesung + Assemblerprogrammierung
  - **Schwerpunkthema: Arbeiten mit einem Realzeitbetriebssystem (RBS) und Programmieren von Echtzeit-Anwendungen.** ► E Kapitel und Praktikum
    - Wozu braucht man RBS? ► Was versteht man unter Echtzeitverarbeitung?
    - Multitasking ► quasi zeitparalleles Abarbeiten von Aufgaben am Rechner
    - Aufbau und Funktionen eines RBS am Beispiel OS9
    - Planung der Abarbeitung von Echtzeitaufgaben auf einem PR
    - **Voraussetzung: fundierte C-Programmierenkenntnisse**



## 1.1.2 Allgemein: Automatisierung von Prozessen

### 1. Zunächst landläufiges Vorurteil: "**Automat ersetzt Menschen**"

- :- ( Mit dem Begriff der Automatisierung wird oft die Vorstellung verbunden: der Mensch soll durch Automaten **ersetzt** werden.
  - ▶ *Der Automat ist in dieser Vorstellungswelt oft eine mehr oder minder komplizierte Maschine.*
- ☺ **Über den Einsatz von Automaten zu Unterstützung der Menschen reden**, denn der Automat kann verglichen zu den menschlichen Fähigkeiten doch nur einen sehr begrenzten Einsatzraum abdecken.
- :- ( Beim Automat schlecht ausgeprägt oder gar nicht vorhanden sind:
  - freie Entscheidungsmöglichkeiten
  - Kreativität zur Verbesserung
  - Spontanität
  - Mut zum Risiko
  - Taktile Fähigkeiten komplexe Dinge zu handhaben
  - Mustererkennungseigenschaften
  - Abstraktionsfähigkeiten.
- ☺ Beim Automaten gut ausgeprägt: Schnelligkeit und Zuverlässigkeit,
  - ▶ *dort wo Arbeiten endlos wiederholt und quasi immer gleich ablaufen sollen.*

### 2. Der Vorgang der Einführung von Automaten in irgendwelche Abläufe nennt man **Automatisierung**

### 3. Den Umfang in dem ein Vorgang oder ein Prozess automatisiert ist nennt man den **Automatisierungsgrad**.

*Beispiel: Abtauautomatik arbeitet vollständig 100% automatisch. Das Automatikgetriebe im Fahrzeug hingegen muss noch vom Menschen bedient werden, je nach Fahrsituation.*

4. Betrachtet man die automatisierten Prozesse näher so lassen sich zwei Ebenen unterscheiden:

a: **Den Vorgang,**

*der einen bestimmten Zweck oder ein bestimmtes Ziel hat:  
Die Maschine, den technischen Prozess, die Anlage selbst*

b: **Vorrichtung,** die diesen Vorgang steuert ohne dass der Mensch eingreift: die **automatische Steuerung(Automat)**

5. Wir befassen uns näher mit der automatischen Steuerung.

Nach DIN 19233 ist der Automat definiert als:

**"Ein Automat ist ein künstliches System, das selbsttätig ein Programm befolgt. Auf Grund des Programms trifft das System Entscheidungen, die auf der Verknüpfung von Eingaben mit dem jeweiligen Zustand des Systems beruhen, und Ausgaben zur Folge haben"**

► *Def. reicht von mechanischen Automaten(Maschinen) bis zur Steuerung von Prozessen durch Computer.*

• Die Definition legt zwei wesentliche Merkmale eines Automaten fest:

- **EVA Struktur:** kausale Folge von **Eingabe, Verarbeitung und Ausgabe**
- Die Verarbeitung beinhaltet **Entscheidungen zwischen verschiedenen Möglichkeiten die von der Eingabe und/oder!! vom aktuellen Zustand (Gedächtnis) des Systems abhängen.**

*Beispiel Kaugummiverkaufsautomat:*

*Eingabe: Münzeinwurf*

*Verarbeitung: Prüfung der Münzen und die Entscheidung einen Kaugummi auszuwerfen, wenn Münzen Ok. und ausreichend und Kaugummis vorhanden (Zustand des Systems) ansonsten die Münzen auszuwerfen.*

*Ausgabe: Entweder Münzen oder Kaugummi (Automat wechselt nicht)*

► *Automat wechselt nicht!!*

### 1.1.3 Ziele der Automatisierung in der PDV

- Prinzipielle Aufgabe der Automatisierung ist eine kostengünstige Vereinfachung beim Betrieb eines Prozesses.
- Es sind typischerweise vier Automatisierungsbereiche zu beobachten:
  - Automatisierung von Arbeiten, die regelmäßig und **monoton** wiederkehrend sind  
*Beispiel: Fließband, Füllen von Flaschen, Bestücken von Leiterplatten, Punktschweißen von Kfz-Teilen*
    - ▶ CNC-Automaten übernehmen hier die monotonen Arbeiten vollständig
    - ▶ automatische Anpassung des Automaten an veränderte Teile , z.B unterschiedliche Flaschengrößen in unterschiedlichen Befüllungsläufen, geschieht i.d.R. nicht automatisch. ▶ neues Programm erforderlich, individuelle Anpassung der Anlage
  - Automatisierung von Arbeiten die **hohe Anforderung an die Konzentration** stellen.
    - ▶ Aufgaben an den Menschen mit stark wechselnden Anforderungen, hier wechseln Zeiten hoher Aktivität und hoher Anforderung an die Aufmerksamkeit und mit Zeiten geringer Aktivität.  
*Beispiel: Bedienung einer Eisenbahnschranke*
      - ▶ Bei einem unregelmäßig ablaufenden Prozess sind Bedienfehler und Unfälle häufig.  
Oft können solche Prozesse nur teil-automatisiert werden, besonders dann wenn viele verschiedene möglichen Ereignisse und Aufgaben berücksichtigt werden müssen
  - Automatisierung von Arbeiten, die eine **hohe Arbeitsgeschwindigkeit** oder **Reaktionsgeschwindigkeit** erfordern.  
*Beispiel: Abfüllen von Zigaretten und Verpacken geht so schnell, dass das Auge nicht mehr mitkommt.*  
*Beispiel: Automatisches Abschalten der Drehbank bei Werkzeug- oder Werkstückbruchs*  
Hier muss zwangsläufig 100% Automatisierung erreicht werden.
  - Automatisierung von Arbeiten, die eine **große Datenflut** liefern.  
*Beispiel: Zählung von Blutkörperchen in einer Blutprobe, Computertomographie, Führung eines Flugzeugs*
    - ▶ Solche Aufgaben können oft nur durch den Einsatz von Prozessrechnern bewältigt werden.

*Fazit: Die Prozessautomatisierung sollte zu einer Entlastung des Menschen von einfachen Arbeiten führen. Damit wird Raum geschaffen für humanere oder anspruchsvollere Arbeiten. Außerdem führt sie zu höherer Produktivität, die sich sowohl in größeren Mengen als auch in besserer Qualität bei vergleichbaren Kosten ausdrückt.*

- **Warnung: Bisherige Grenzen in der Automatisierung von Prozessen**

- **Problem der Vollständigkeit der Automatisierung.**

Ein automatisches System kann nicht angemessen auf alle real vorkommenden Ereignisse reagieren, sondern nur auf solche, die einprogrammiert wurden.

- ▶ *fatales Ereignis, Ein-Steuerung des Umkehrschubs bei A320 Airbus war nur dann möglich, wenn Fahrwerksräder sich drehen, damit es nicht versehentlich während des Flugs passieren kann (Antwort auf den Unfall der Lauda-Air in Indien!) . In Warschau war soviel Regen, dass durch Aquaplaning sich die Räder nicht gedreht haben. Der Pilot konnte den Umkehrschub nicht einschalten, fatale Folge!! Großes Unglück mit Verletzten und Toten!!*

- **Problem der Korrektheit eines Programms**

- ▶ Programmfehler können lange Zeit unentdeckt bleiben!! wenn entsprechende Zustände und Ereignisse selten sind.

- Beispiel: Der GAU eines Kernkraftwerkes kann nicht getestet werden!! nur simuliert!!*

- ▶ Aufgabe: Validierung der Software => Ungelöstes Problem!! wg. Komplexität

- **Problem der mangelhaften Erkennungs- und Erkenntnisfähigkeit des Rechners.**

- Beispiel: automatische 100% sprecherunabhängige Spracherkennung und Sprecheridentifizierung  
Bildererkennung und Bildverstehen => Führerloses Autofahren ist noch Zukunftsmusik!*

Abhilfe: Grundlagenforschung!! Es fehlen noch grundlegende in technische Systeme umsetzbare Erkenntnisse der Wirkungszusammenhänge!!

## 1.2 Begriffe und deren Bedeutung aus der PDV

### 1.2.1 Prozess allgemein, Unterscheidung nach Verarbeitungsart und Verarbeitungsgut

- *Bisher: gesprochen vom Prozess, Vorgang und Maschine ► Nun wird eine Definition benötigt, um festzulegen was in der PDT genau darunter zu verstehen ist*
- Es gibt die DIN 66 201 die besagt:
  - Ein **Prozess** ist der Vorgang zur **Umformung, zum Transport oder zur Speicherung** von **Materie, Energie oder Information**
- Veranschaulicht: Verarbeitungsart ► Vorgang und das Verarbeitungsgut ► Materie, Energie, Information

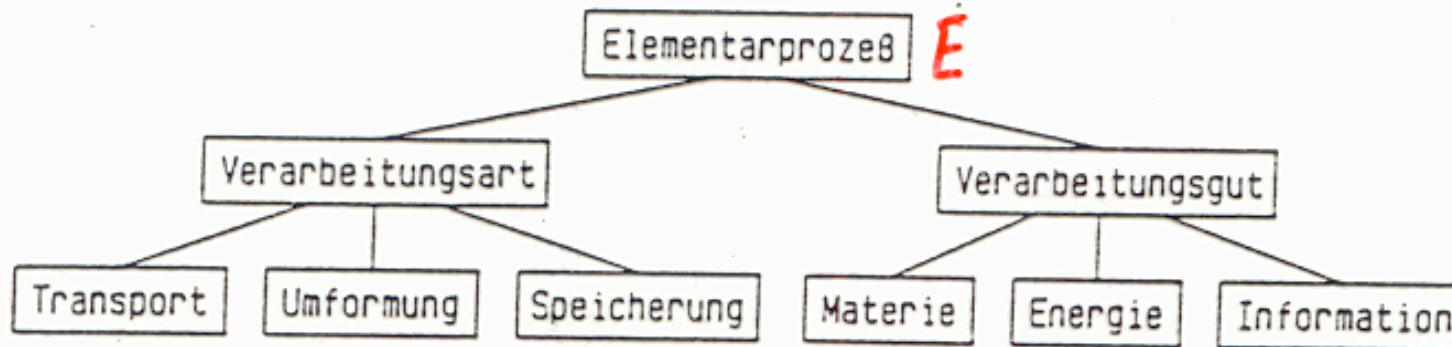


Bild 1.2 Ein Elementarprozess wird durch eine von 3 Verarbeitungsarten und eines von 3 Verarbeitungsgütern bestimmt. So ergeben sich 9 verschiedene Grundtypen.

- *Beispiele:*

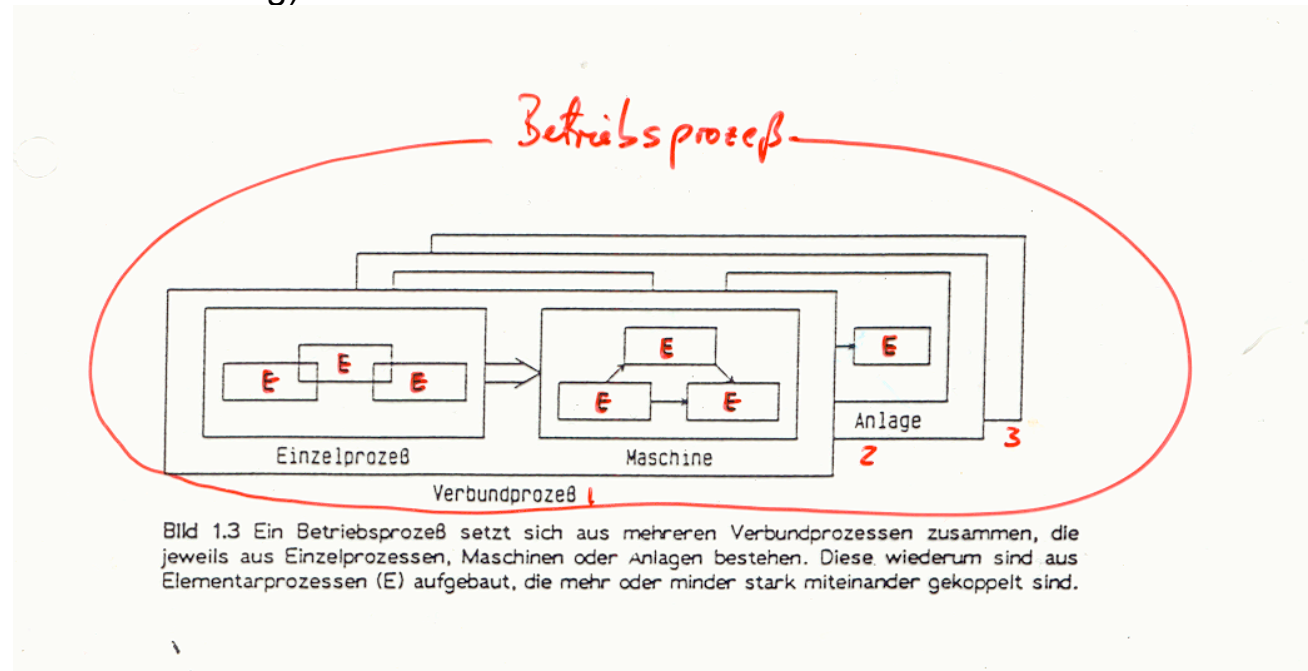
### 1. Verarbeitungsart

- **Umformung** erfolgt in Erzeugerprozessen: Drehbank zur mechanischen Bearbeitung Umformung Materie), Galvanisierbad mit Einsatz von Chemie und elektrischer Energie
- **Transport** von Gütern (Materie, Energie): Bestückung von Leiterplatten, Verkehrsnetze, Energieversorgungsnetze,
- **Speicherung** bezeichnet man als Aufbewahrungsprozesse z.B. Hochregallager in der Fabrik

### 2. Verarbeitungsgut

- **Materie:** geg. in fester, flüssiger oder gasförmiger Konsistenz ► Umwandlung der Aggregatzustände kann beim Bearbeiten, Verbinden, Umwandeln (Schmelzen) Transport oder Lagerung erfolgen.
- **Energie:** Erzeugung, Transport
- **Information:** Verarbeitung PDV-Anlagen
- Der **Elementarprozess** ist gekennzeichnet durch **genau eine** Verarbeitungsart **und ein** Verarbeitungsgut
  - Heizung: Umformung elektrischer Energie in Wärme*
  - Kessel: Speicherung von Materie*
  - Leitung Transport von Energie oder Material*
- Der **Einzelprozess** stellt die kleinste geschlossene Prozesseinheit dar, bei der **ein kompletter Bearbeitungslauf** durchgeführt wird. Er ist aus mehreren Elementarprozessen zusammengesetzt, die eng miteinander verknüpft sind.
  - Eine Maschine basiert auf mindestens einem/mehreren Einzelprozessen
  - Eine Anlage besteht wiederum aus mehreren Maschinen und basiert ebenfalls auf mindestes einem Einzelprozess.
- Der **Verbundprozess** besteht aus mehreren Einzelprozessen (Maschinen) die zusammenarbeiten um **eine gemeinsame Aufgabe** zu erledigen.
  - Beispiel: Fertigungsstraßen , Transportsysteme*
- Der **Betriebsprozess** umfasst verschiedene Verarbeitungsbereiche eines Betriebs: Fertigung, Lager, Versand, Buchführung, Auftragsabwicklung, Verwaltung, Entwicklung.

- Die Einbeziehung **aller** Betriebsbereiche **in einen Prozess** wird mit dem Schlagwort **CIM** (Computer Integrated Manufacturing) bezeichnet.



- ▶ Hintergrund ist die Optimierung des gesamten Ablaufs in einem Fertigungsunternehmen, um Kosten und Zeit zu sparen bei Steigerung der Produktivität. ▶ Hier gibt es zwei Vorgehensweisen:
  - globale zentrale Strategie
    - Es wird versucht -isoliert- eine Gesamtlösung auf einmal zu optimieren ▶ im Extremfall nur einmal und hat dann die optimale Lösung.*
    - ▶ Problem: Die zu betrachtenden Vorgänge sind meist unüberschaubar vielfältig und im Detail voneinander abhängig ▶ kein Einzelner überblickt mehr alles.
    - ▶ Problem: Umsetzung der globalen Optimierungsvorgaben im Detail ▶ oft verhält sich die Umsetzung nicht so optimal wie geplant, sondern eher hinderlich, weil Details nicht berücksichtigt wurden durch die zentrale Steuerung.
  - Strategie der kleinen Iterationen: Prozess der stetigen lokalen Verbesserung mit dem Ziel Zeit und Kosten einzusparen bei gleich bleibender Produktivität oder Produktivität zu erhöhen bei konstanten Kosten und Zeit.
    - Jeder Teilprozessverantwortliche optimiert (theoretisch bis er überflüssig ist) lokal und im Hinblick auf seine Prozessschnittstellen.*
    - ▶ Grundgedanke: Pragmatismus und der Glaube dass die, die mit der Materie jeden Tag zu tun haben am besten wissen was zu optimieren ist und das Teiloptimierungen sich gegenseitig optimieren bis zu dem Zeitpunkt bis ein Gesamtoptimum erreicht ist ▶ Problem ggf. lokales Minimum gefunden und nicht globales. ▶ funktioniert im Unternehmen nur dann gut, wenn die Mentalität der Betroffenen diese Vorgehensweise unterstützt. ▶ Abteilungsdenken hinderlich!!

## 1.2.2 Der technische Prozess

Wir befassen uns mit einer besonderen Sorte von Prozessen:

- Ein **technischer Prozess** ist ein spezieller Prozess ▶ dessen Zustandsgrößen werden mit technischen Mitteln erfasst und beeinflusst (gemessen, gesteuert, geregelt)
- ▶ Prozessdatenverarbeitungssysteme (PDVS) bestehend aus Hardware und Software stellen u.A. solche technischen Mittel zu Beeinflussung und Erfassung bereit:
  - Die Schnittstelle PDV und Prozess ist die elektrische und logische Schnittstelle des jeweiligen Sensors und Aktors.
  - ▶ **Mit den Grundlagen der PDVS beschäftigen wir uns hier und gehen davon aus, dass die technischen Prozesse und deren Automatisierungsanforderungen bekannt und gegeben sind.**

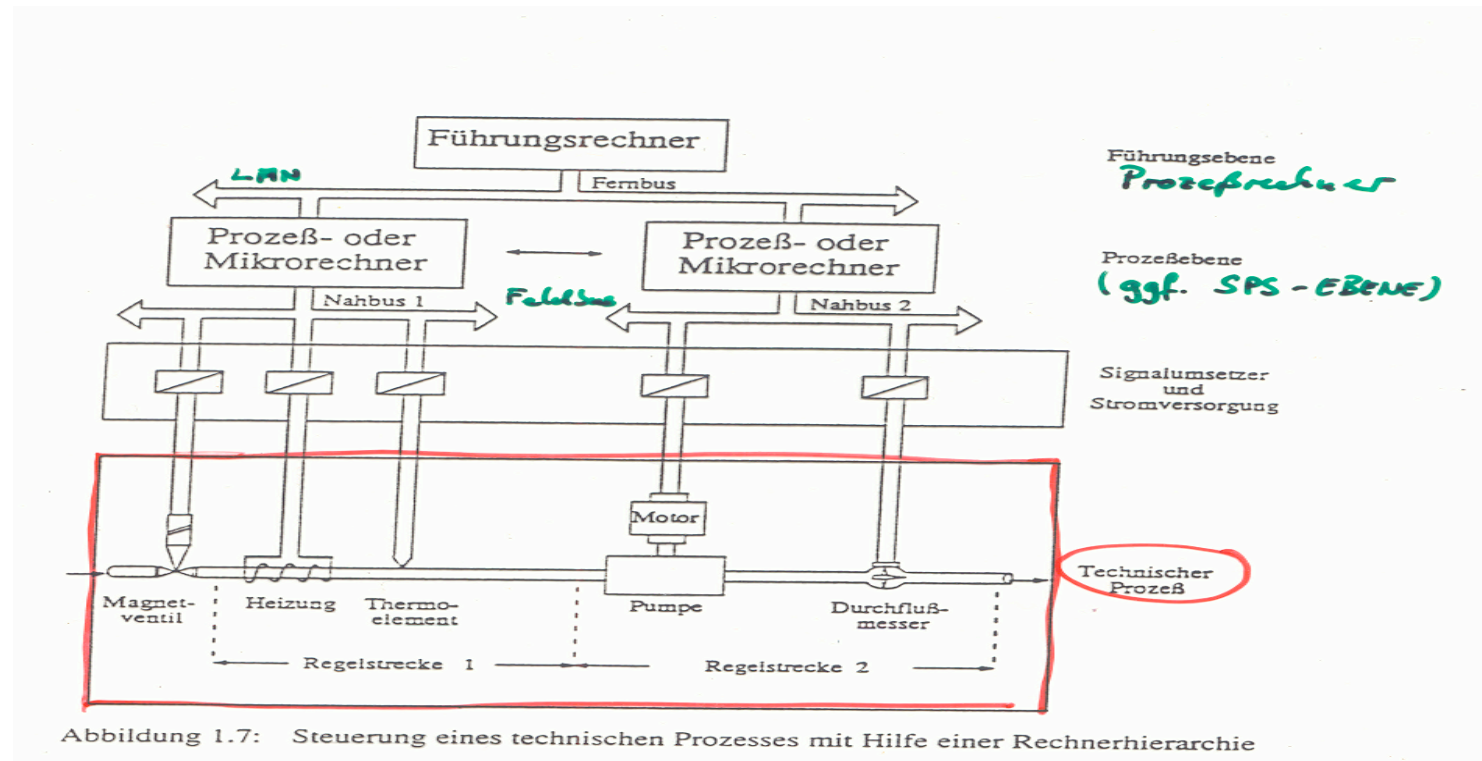


Abbildung 1.7: Steuerung eines technischen Prozesses mit Hilfe einer Rechnerhierarchie

Beispiel: ▶ **Abbildung 1.7** : Steuerung eines technischen Prozesses mit Hilfe einer Rechnerhierarchie



*Verarbeitungsgut: Flüssigkeit*

*Verarbeitungsart: Flüssigkeit wird geheizt ► temperaturgeregelt und die Durchflussmenge gesteuert ► PDT-Hardware beginnt bei Signalumsetzer und erstreckt sich bis zur Führungsebene ► das unmittelbare Erfassen und Verändern der Zustandsgrößen über Sensoren, Mechanik, Pneumatik, Pumpen etc zählt in erster Linie mit zur Anlage.*

## 1.2.3 Klassifikationsmerkmale von technischen Prozessen

► Es gibt viele technische Prozesse, die auf Grund ihrer Eigenheiten durch unterschiedliche Methoden und Mittel automatisiert werden müssen, deshalb zunächst eine weitere Verfeinerung der Klassifizierung neben Prozessgrundtyp und Verarbeitungsarten und -gütern.

### 1.2.3.1 Unterscheidung nach Verarbeitungsstrukturen

• Es gibt folgende unterschiedliche Verarbeitungsstrukturen von Prozessen:

#### 1.kontinuierliche oder auch stetige Verarbeitung

- stetiger Strom von Materie oder Energie wird transportiert oder umgewandelt
- Einzelteile können technisch oder prozessbedingt nicht voneinander getrennt bzw. identifiziert werden

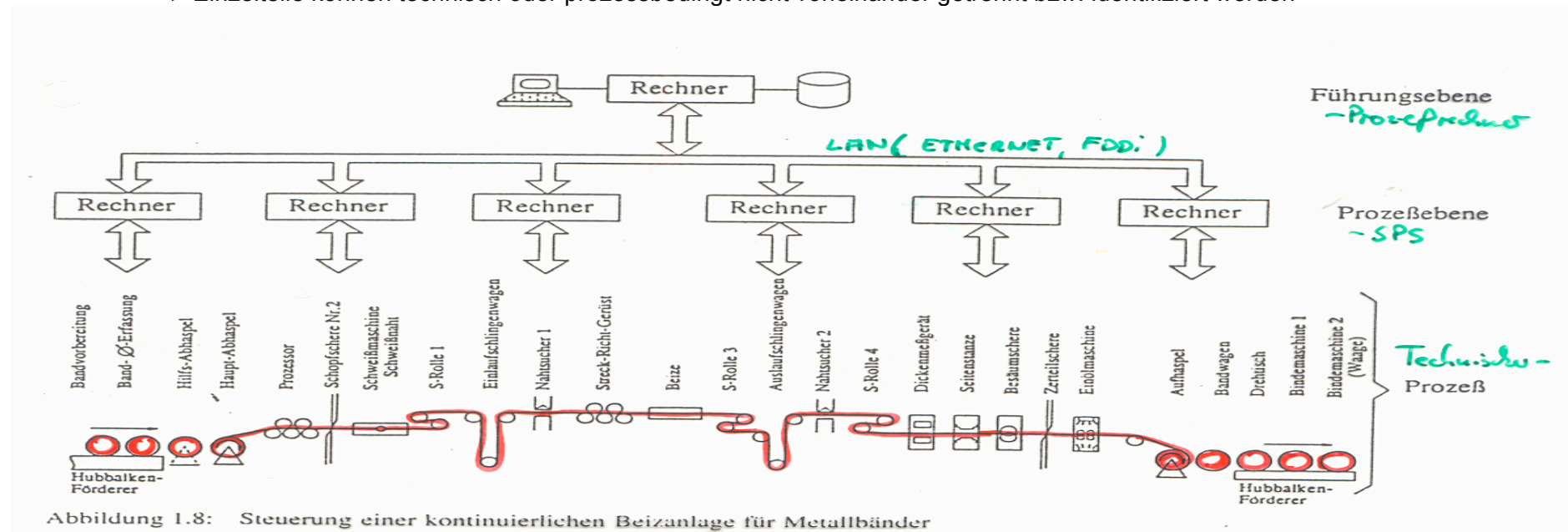


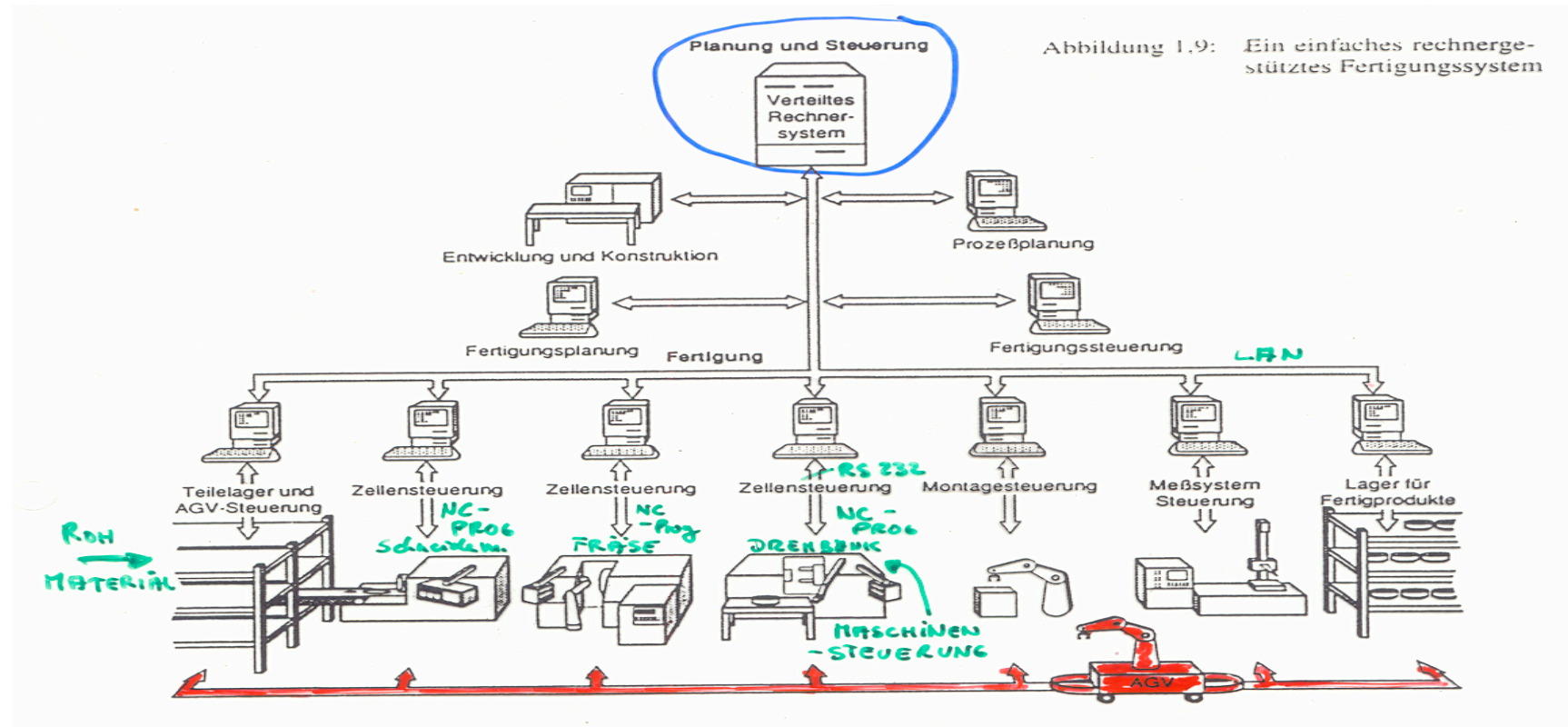
Abbildung 1.8: Steuerung einer kontinuierlichen Beizanlage für Metallbänder

Beispiele: kontinuierliche Stahlerzeugung, chemische Prozesse, petrochemische Prozesse, Energieerzeugung

Bildbeispiel: => **Bild 1.8:** Beizanlage zur Oberflächenbehandlung von Metallblechen Die Metallrollen werden durch einen Hubbalkenförderer angeliefert und in eine Abhaspelanlage (Abrollanlage) eingelegt. Über Förderrollen gelangt das Blech in eine Schopfschere und wird dort abgerichtet (zurechtgeschnitten), damit das neue Band mit dem alten zusammengeschweißt werden kann. Über ein Streck-Richt-Gerüst kommt das Band in die Beizanlage und danach in eine Abricht (Zuschneide-)anlage zum Seitenschneiden und Besäumen (Saum links und rechts!). Danach wird das Band wieder aufgehaspelt (gerollt) und über ein Fördersystem verpackt (► wenn Rolle voll ist noch vorher abgeschnitten und eingeölt) und abtransportiert. Steuerung über ein hierarchisches Rechnersystem.

## 2. diskontinuierliche oder auch unstetige bzw. diskrete Prozesse

► i.A. sind das Stückprozesse, die bei der Fertigung oder Behandlung von diskreten einzelidentifizierbaren Produkten zu finden sind.



Beispiele: Einzel- und Massenfertigung, Einzel- und Bandmontage, Stückgutförderung, Lagerhaltung, Nachrichtenübertragung

Bildbeispiel ► **Bild 1.9:** Fabrik zur mechanischen Bearbeitung von Werkstücken (bis hin zur Losgröße 1 mit Umrüsten der Maschinen durch den Robbi)

Am Planungs- und Steuerrechner wird die Bearbeitung von Aufträgen geplant und freigegeben. Das Rohmaterial gelangt über das Eingangslager in den Fertigungsablauf. Die individuellen Werkstücke werden an den einzelnen Stationen gemäß den Fertigungsvorgaben bearbeitet. Diese Vorgaben und NC-Programme werden vom Planungs- in den jeweiligen Steuerrechner eingegeben. Diese initialisieren und überwachen die Fertigungsabläufe in den Maschinen (Maschinensteuerung extra). Die fertigen Teile werden in einer Montagestation zusammengebaut und anschließend extra vermessen. Von dort gelangen die fertigen Teile ins Fertiglager. Für den Transport von Bearbeitungsstation zu Bearbeitungsstation ist ein rechnergesteuertes Transportsystem zuständig.

### **3. Hybride Prozesse oder auch Chargenprozesse sind die am häufigsten anzutreffenden Prozesse**

- ▶ Mischform aus 1. und 2.: Die diskrete Verarbeitung von gebündelten zusammen-gehörig-identifizierbaren Teilmengen eines sonst kontinuierlichen Gutes. Beispiel: Brauen eines Bieres Bottichweise in Weihenstephan

### 1.2.3.2 Unterscheidung nach der inneren Ablaufstruktur

Neben der grundsätzlichen Verarbeitungsstruktur innerhalb des Prozesses ergibt sich auch noch eine Ablaufstruktur des Prozesses die es zu berücksichtigen gilt, nach denen sich die PDT zu richten hat.

- **deterministische Ablaufstruktur**

*hier liegt die zeitliche Abfolge fest* Beispiel Schienenverkehr

- **stochastische Ablaufstruktur**

*hier gelten die Gesetze der Statistik* Beispiel Straßenverkehr

- **!►! Wichtig: Wahl der Algorithmen !!!**

### 1.2.3.3 Unterscheidung nach Einsatzgebiet

- **Fertigung**

► Verarbeitung von Gütern insbesondere der Verarbeitung von Material in Stückprozessen (diskontinuierlich) und Erzeugungsprozessen (Umformung, Herstellung)

*Automobilbau  
Elektroindustrie*

- **Verfahrenstechnik**

► kontinuierliche Verarbeitung oder chargenweiser Verarbeitung von Gütern und Energie zum Ziel

*Chemische Industrie  
Energieerzeugung in Kraftwerken*

- **Verteilungsprozesse**

*Lager und Versandhäuser  
Nachrichtennetze (Internet, Verteilung von Information)*

- **Verwaltung**

► Fachgebiet der EDV  
*Bürobereich, Buchhaltung etc.*

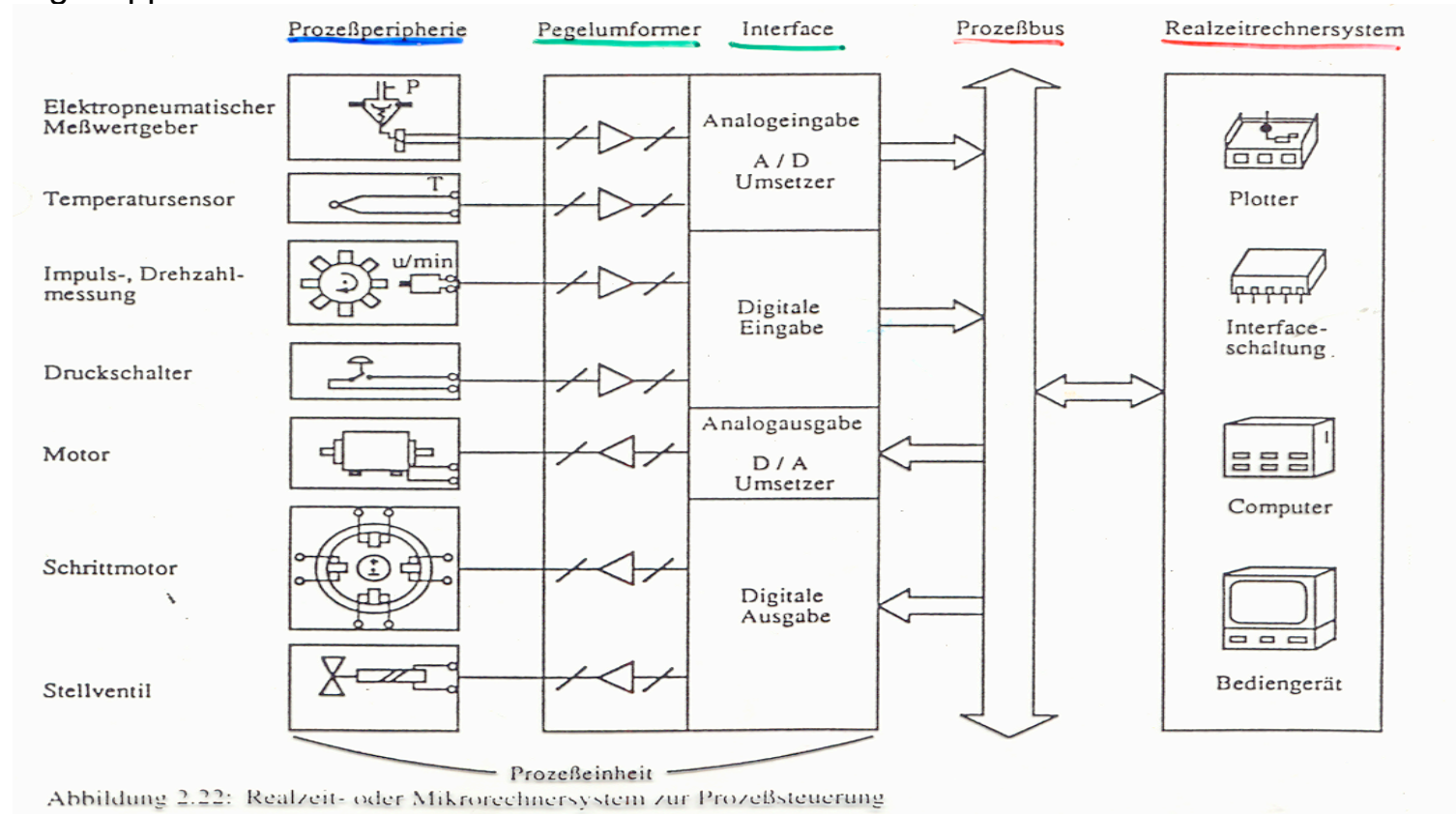
- **Mess- +Prüfprozesse**

- Forschung und Entwicklung
- Dienstleistungsbereich

*Beispiel: Medizinlabors, Umweltschutz*

### 1.2.4 Prozessrechner (PR)

- Als Prozessrechner wird ein Rechner bezeichnet der mittels Prozessperipherie direkt an den technischen Prozess gekoppelt ist.



## **Neu Prozessperipherie:**

- **Elektropneumatischer Messgeber,**
  - ▶ Umwandlung pneumatisches Prozesssignal in einen elektrischen Messwert ▶ verstärkt ▶ A/D -Wandlung ▶ Verarbeitung dann digital
- **Temperatursensor:**
  - ▶ thermoelektrisches Prinzip ▶ Widerstandsänderung ▶ kleine Spannung ▶ Millivolt- Bereich wird verstärkt ▶ A/D Wandlung ▶ Verarbeitung dann digital
- **Impulsdrehzahlmessung:**
  - ▶ Vom Drehzahlgeber werden Impulse mit niedriger Spannung erzeugt ▶ Rechteckformung ▶ Flankenwechsel/Zustandswechsel (vorgeschaltete Zähler) ▶ DigIn Rechner wertet aus: Anzahl der Impulse/Zeit, Impulsdauer ▶ Verarbeitung dann digital
- **Mechanischer Druckschalter:**
  - ▶ Prinzip Schalter ▶ DigIn
- **Motor:**
  - ▶ Zur Regelung eines Stellgliedes Digiwert ▶ D/A ▶ Verstärker ▶ Antriebssignal **Motor**
- **Schrittmotor:**
  - ▶ Motor kann in definierten Winkelpos. eingestellt werden ▶ Rechner schickt eine vorgegebene. Anzahl von Impulsen ▶ DigOut ▶ Verstärker ▶ Schrittmotor
- **Stellventil:**
  - ▶ Ventil schließen == Impuls ausgeben => DigOut => verstärken => Stellventil
- **Pegelumformer und Prozessinterface**
  - ▶ Prozessrechner arbeitet intern digital i.d.R mit 5V (3V) ▶ Signalform und Signalpegel der ankommenden und gesteuerten Signale müssen entsprechend angepasst werden
  - ▶ Pegelumformer und Verstärker (Analogtechnik). Häufige zu lösende Probleme: der galvanischen Massekopplung und Störungsbeeinflussung Rechner <▶ Signallieferant
  - ▶ Umwandlung von analogen in digitale Signale
  - ▶ digitale Signale vom/zum Prozess 24V
  - ▶ Impulsformung
  - ▶ Frequenz zu Spannungsumsetzung
- **Prozessbus**
  - ▶ Verbindung ("Highway") zum eigentlichen Rechner
  - ▶ Aufgabe: Transport Prozessdaten, Adressen Status und Steuersignale
  - Ausprägungen:
    - ▶ es ist der Systembus selbst ▶ wenn Rechner nahe den Prozesssignalen
    - ▶ wenn Rechner weiter weg existiert oftmals gesonderter Peripherie- oder Prozessbus mit serieller oder paralleler Ausprägung
    - ▶ Prozessbusinterface im Rechner und in der Prozessperipherie nötig.

- *Die Bezeichnung "Prozessrechner" und/oder „Automatisierungsrechner“ hat sich eingebürgert, da dieser Rechnertyp vorwiegend zur "Automatisierung technischer Prozesse" eingesetzt wird. Im englischen Sprachraum werden Prozessrechner unter dem Sammelnamen "Minicomputer" geführt.*



## 1.2.5 Prozessdaten und Prozessgrößen

Häufig wird von Prozessdaten und Prozessgrößen gesprochen. ► Deutlicher Unterschied:

- **Prozessdaten** werden zwischen PR und Prozess ausgetauscht
- **Prozessgrößen** beschreiben einen technischen Prozess, unabhängig von seiner Lenkung durch den Rechner.
- **Prozessgrößen:**
  - Prozessgrößen sind technische, physikalische oder chemische Größen, die mit technischen Mitteln erfasst und beeinflusst werden können. Diese werden mit geeigneten Maßeinheiten durch Zahlenwerte (Daten) beschrieben.  
*Beispiel: Geschwindigkeit in m/s, Masse in kg, Ph-Wert als reine Zahl, Drehzahl in u/min*
  - Prozessgrößen sind für jeden einzelnen Prozess und sein Verarbeitungsgut verschieden, aber es gibt für jede Verarbeitungsart charakteristische Größen:
    - Speicherprozess: die Menge ( *kg, J bzw. kWh, Stück* )
    - Transport: der Durchsatz ( *in kg/s, W, Stück/Zeit* )
    - Umformung: Art und Umfang der Änderung; wobei aber nach den Erhaltungssätzen der Physik oder Chemie die Gesamtmengen unverändert bleiben.  
*Beispiel: Lochbohren in Metall: Lochspäne + Werkstück = Gesamtmenge*

- **Prozessgrößen untergliedern sich in:**

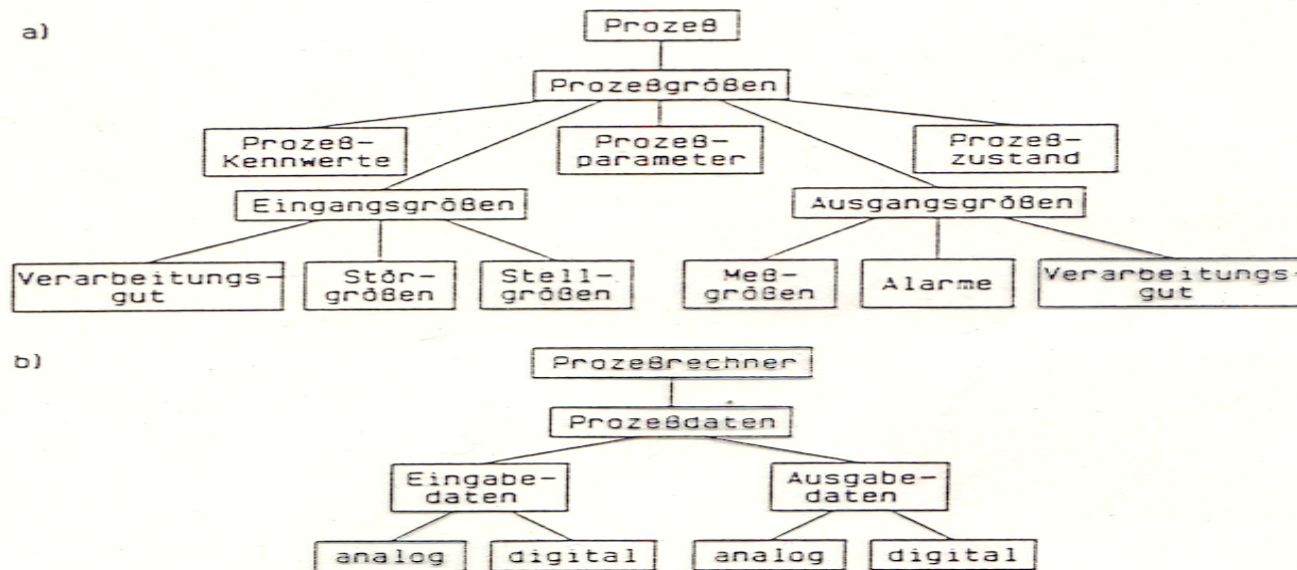


Bild 1.4 Prozessgrößen und Prozessdaten

a) Prozessgrößen charakterisieren einen Prozess. Manche von ihnen sind fest vorgegeben (statisch), z.B. das Verarbeitungsgut; andere sind veränderlich (dynamisch), z.B. der Prozesszustand.

b) Prozessdaten werden immer aus der Sicht des Prozessrechners betrachtet. Die Eingabedaten sind Ausgangsgrößen des Prozesses während die Ausgabedaten zu Eingangsgrößen des Prozesses werden (vgl. Bild 1.5).

- Prozesszustandswerte und -daten

- ▶ sind die im Betrieb auftretenden veränderlichen Größen, die den aktuellen Zustand des Prozesses beschreiben und mit technischen Mitteln erfasst und beeinflusst werden.

- ▶ z.B.: Drehzahl eines Motors, Temperatur einer Waschmaschine

- Prozessparameter

- ▶ sind solche Größen die für einen Prozesslauf konstant sind, aber für einen nächsten Prozesslauf geändert werden können.

- ▶ z.B.: Einstellung der Waschmaschine auf ein Waschprogramm=Kochwäsche austauschbare Werkzeuge in einer NC-Maschine

- Prozesskennwerte
  - ▶ *oder auch Anlagedaten genannt sind technische Daten, die durch den Aufbau des Prozesses, seine Struktur, Konstruktion und Anwendung festgelegt sind. Es sind also Festwerte, zu denen vor allem auch **Grenzwerte** gehören.*
  - ▶ *Beispiel: die Heizleistung bei einer Waschmaschine, der max. Durchsatz in einem Fördersystem*
- Eingangsgrößen
  - ▶ *fließen in den Prozess hinein und wirken auf den Prozess. Sie werden auch als freie Variable bezeichnet, da ihre Werte unabhängig vom Prozesszustand und -ablauf angenommen werden können.*
  - Zufluss von Verarbeitungsgut:
    - ▶ *Der Zufluss von Verarbeitungsgut (Materie, Energie) erfolgt kontinuierlich oder in Chargen und beeinflusst dabei den Zustand des Prozesses.*  
*Beispiel: der Zulauf von kaltem Wasser in die Waschmaschine, je nach Witterung unterschiedlicher Temperatur beim Einlauf, verändert die Waschwassertemperatur, die geregelt sein soll, und damit den Prozesszustand.*
  - Stellgrößen:
    - ▶ *wirken mehr oder minder direkt auf die Prozesszustandsgrößen*  
*Beispiel: auf das Ventil für den Wasserzu- und ablauf*
  - Störgrößen:
    - ▶ *treten meist als unerwünschter und oft schwer kontrollierbarer Zu- oder Abfluss von Verarbeitungsgut auf, oder sogar als Störung von Stell- und Messgrößen.*  
*Beispiel: ein undichtes Ventil, Wärmezufuhr beim Öffnen eines Kühlschranks, Rauschen oder Übersprechen auf Signalleitungen.*
- Ausgangsgrößen
  - ▶ *fließen aus dem Prozess heraus, dabei können sie aber auch auf den Prozess zurückwirken. ▶ stets abhängig vom aktuellen Prozesszustand ▶ Bezeichnung deshalb als abhängige Größen*
  - Abfluss von Verarbeitungsgut:
    - ▶ *hängt vom aktuellen Zustand des Prozesses und Stellgrößen ab. ▶ verändert i.d.R den Prozesszustand, oft wie Zufluss nur mit umgekehrtem Vorzeichen.*  
*Beispiel: der Ablauf (=Abfluß) von Wasser aus der Waschmaschine hängt vom Zustand Temperatur und Wasserstand und der Stellgröße Ablaufventilstellung ab. Der Wasserstand und ggf. die Temperatur verändert sich damit*

- Messgrößen:

- ▶ *sollen die aktuellen Prozesszustandsgrößen unabhängig voneinander angeben, und sollen **keine** Rückwirkung auf den Prozess haben. Auch die Stellgrößen sollten keine direkten unmittelbaren Einwirkungen auf sie haben.*

- Beispiel: die Temperaturanzeige des Kühlwassers im Fahrzeug soll unabhängig von (Prozesszustand) Geschwindigkeit und Beladung erfolgen und darauf auch keinen Einfluss haben. Sie sollte aber auch nicht von der Gaspedalstellung (Stellgröße) abhängen. Wenn der Tankanzeiger von Neigung und Temperatur abhängt so ist das eine Fehlkonstruktion.*

- Alarme:

- ▶ *spezielle Messgrößen signalisieren besondere wichtige Ereignisse, die durch bestimmte Prozesszustandsänderungen definiert sind. Beispiel: Überschreitung der zulässigen Kühlwassertemperatur*

• Prozessdaten untergliedern sich in:

- Ausgabedaten
- Eingabedaten
- Alarmer als Sonderform von Eingabedaten
- ► *Alle Daten können sowohl digital (wertdiskret) als auch analoger (wertkontinuierlich) Natur sein!!*

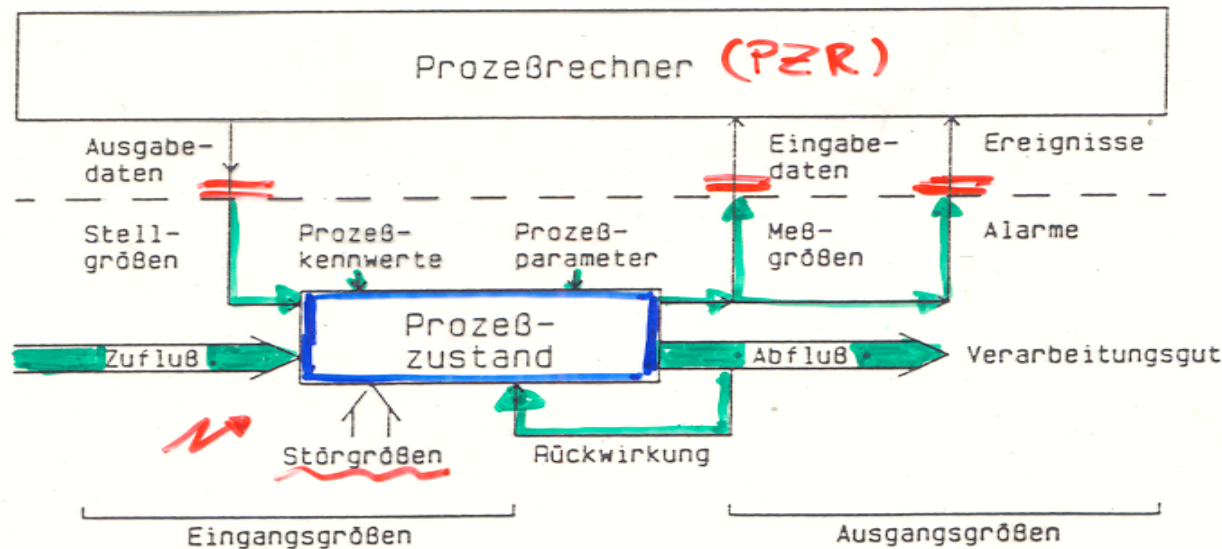


Bild 1.5 Der Prozesszustand wird durch die Eingangsgroßen bestimmt: durch die Stellgrößen, Störgrößen und durch den Zufluß von Verarbeitungsgut; aber auch der Abfluß von Verarbeitungsgut wirkt auf den Prozesszustand zurück.

Der Prozesszustand wiederum bestimmt die Ausgangsgroßen: den Abfluß von Verarbeitungsgut und die Meßgrößen; Alarme werden durch besondere Änderungen des Prozesszustands ausgelöst.

Die Ausgabedaten des Prozessrechners werden zu Eingangsgroßen des Prozesses; Eingabedaten werden aus den Ausgangsdaten des Prozesses gewonnen.

## 1.2.6 Rechenprozess, Task und Multitasking

- Die Software, die auf dem PR "läuft" ist für die Funktionalität bzw. Regelung verantwortlich ► hier spricht man genauer von **Rechenprozessen die die Aufgabe der Verarbeitung der erfassten Messgrößen aus dem technischen Prozess und dessen Beeinflussung haben ("was wie wann getan werden muss")**.
- **Rechenprozesse** haben die Umformung, Verarbeitung und den Transport von **Information** zum Gegenstand.
- Der Ablauf eines Programms in einem Rechner wird als Rechenprozess oder kurz als **Prozess oder Task** bezeichnet.
- ► Das **Programm** selbst ist in diesem Zusammenhang die **statische** Aufschreibung von Befehlen => der Algorithmus => die "Bedienungsanleitung" zum technischen Prozess
- ► Der **Rechenprozess (Task)** ist die **dynamische** Ausführung des Programms => die Ausführung der Bedienungsanleitung in Abhängigkeit der Anforderungen des zu bedienenden technischen Prozesses.
- Auf **einem** Rechner können, sofern die Rechnerleistung ausreicht, quasi gleichzeitig mehrere Rechenprozesse ausgeführt werden. Diese Betriebsart eines Rechners nennt man **Multitasking** (oder auch Multiprozessing).
- Mehrere Rechenprozesse können in **einem** Rechner **dasselbe** Programm benutzen (ihr Ablauf muss hierzu mehrmals angestoßen werden) allerdings benötigt **jeder Prozess seinen eigenen Datenbereich**.

*Beispiel:*

*Angenommen es gibt 10 aktive Terminals an einem Rechner => so existiert sinnvollerweise für jedes Terminal ein eigener Rechenprozess mit privatem Datenbereich in dem alle rein- und rausgehenden Zeichen des jeweiligen Terminals verarbeitet werden. Da jedes Terminal systematisch gleich behandelt wird, existiert das Programm zur Terminalsteuerung nur einmal und wird von jedem der 10 Rechenprozesse mitgenutzt. Die Rechenprozesse werden im Betrieb zu einer Zeit in unterschiedlichen Betriebszuständen sein. Der Eine wartet auf Eingabe, der Andere ist gerade damit beschäftigt Zeichen an den Bildschirm auszugeben. Das heißt jeder Prozess hat einen eigenen 'Programmcounter' der jeweils unterschiedlich in das gemeinsam genutzte Programm zeigt. Damit nichts durcheinander kommt, benutzt das Programm nur Daten und Variable, die es vom jeweils aktiven Prozess zur Verfügung gestellt bekommt. Es hat somit selbst keinen eigenen Variablenbereich.*

- **Zu einer Zeit kann in einem Rechner (Single-Prozessor-System) nur ein Rechenprozess aktiv sein.** Die quasi gleichzeitige Abarbeitung aller Rechenprozesse wird durch eine Stückelung und Verteilung der verfügbaren Rechnerzeit auf die einzelnen Rechenprozesse erreicht.

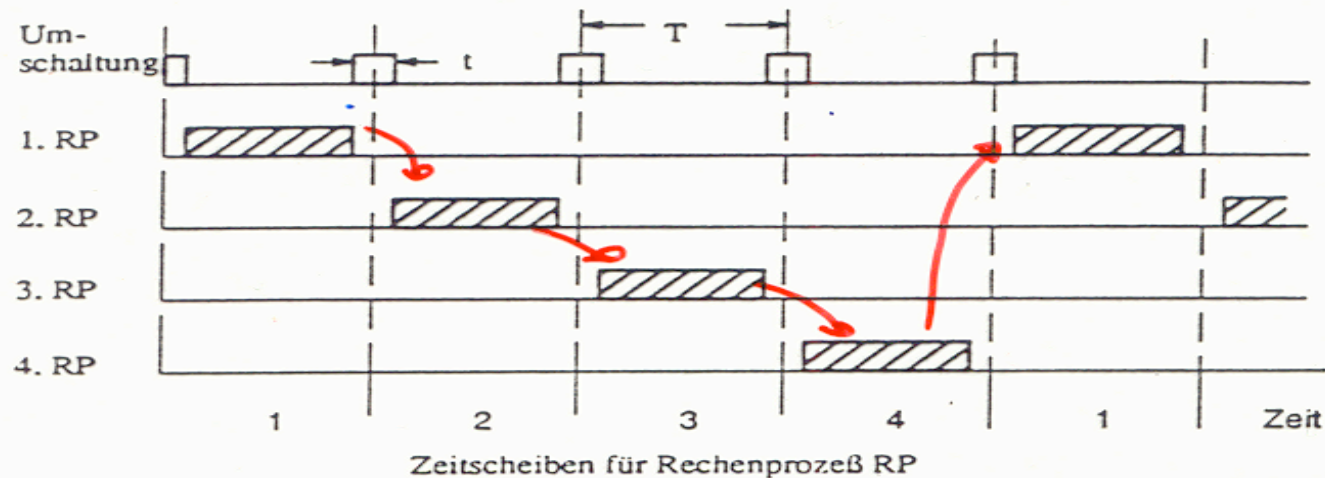


Abb. 1.1: Einfaches Modell für die Aufteilung eines Prozessors auf Rechenprozesse

Beispiel: Abb. 1.1

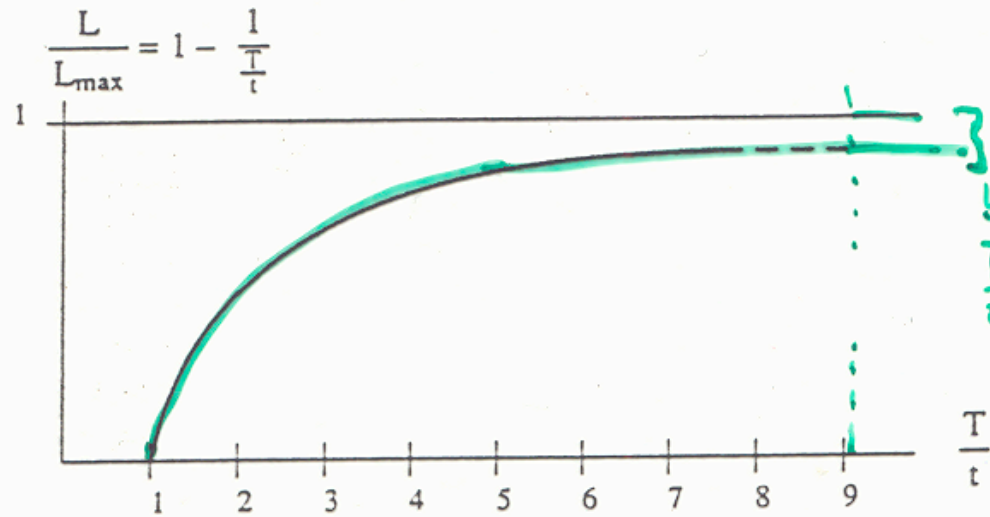
Ein einfaches Modell für eine gleichmäßige Aufteilung der verfügbaren Rechnerzeit eines Rechners auf  $n$ -Rechenprozesse könnte beispielsweise wie folgt aussehen:

Jeder Rechenprozess erhält der Reihe nach jeweils für eine bestimmte Zeitdauer (Zeitscheibe)  $T$  den Rechner zugeteilt. Nachdem der letzte Rechenprozess die Zeit verbraucht hat, erhält der erste wieder eine Zeitscheibe usw.

Für die Umschaltung (Kontextswitch) von einem Rechenprozess auf den nächsten benötigt man eine Zeit  $t$ . Denn zunächst muss der aktuelle Zustand (Kontext) des gerade laufenden Prozesses gerettet bzw. gespeichert werden, damit er beim Erhalten seiner nächsten Zeitscheibe an der unterbrochenen Stelle fortsetzen kann. Im nächsten Schritt muss der gerettete Zustand des nun zum laufen kommenden Rechenprozesses geladen werden, damit an der Stelle weiter gearbeitet werden kann an der letztes Mal dieser Prozess unterbrochen wurde. Nach dem Ladevorgang beginnt dann sofort dieser Rechenprozess für die Dauer seiner Zeitscheibe zu rechnen.

Das Bild zeigt ein Beispiel für  $n=4$  Rechenprozesse (RP) bei  $T=1$  ms und  $t = 0.1$  ms (aus [1]).

## Kontextswitch und Overhead



12% bei  $\frac{T}{t} = 9$   
 $T =$  Zeitschleife  
 $t =$  Kontextwechselzeit

Abb. 1.2: Rechenleistungsverlust bei kleinen Zeitscheiben

$$L = L_{max} - \underbrace{\frac{t}{T} \cdot L_{max}}_{\text{"Verlust"}}$$

↓  
 Max. Rechenleistung

$t \downarrow$  "schnelle HW"  
 $T \uparrow$  ! Realzeitbdg!

$$\frac{L}{L_{max}} = 1 - \frac{t}{T}$$

$$\frac{L}{L_{max}} = 1 - \frac{1}{\frac{T}{t}}$$

z.B.

$\frac{T}{t} = 9$	$t_1 = 100 \mu s$ (80C51)
$T_1 = 300 \mu s$	$t_2 = 20 \mu s$ (68040)
$T_2 = 180 \mu s$	$t_3 = 6 \mu s$ (PowerPC 501)
$T_3 = 54 \mu s$	OS 20ms = $T_1$



Der **Kontextswitch**, vom laufenden Prozess zum nächsten wartenden Prozess stellt einen Verbrauch von Rechnerzeit dar, der keinem Rechenprozess zugute kommt, sondern sogar die für die Anwendungen verfügbare Rechnerzeit reduziert, er ist also unerwünschter aber notwendiger **Overhead**.

*Der Overhead wird in % der Dauer einer Zeitscheibe angegeben.*

*Kontextswitchdauer  $t=0.1\text{ ms}$  , Zeitscheibe  $T=1\text{ms}$  Overhead=  $0.1 / 1.0 * 100\% = 10\%$*

*Der Betriebssystembauer ist nun bestrebt den Overhead so klein wie möglich zu halten.*

*Die Kontextswitchdauer  $t$  ist in der Regel von der Architektur und Leistungsfähigkeit des Rechners abhängig, und somit in einer vorgegebenen Umgebung annähernd konstant. Somit lässt sich über die große Zeitscheibengrößen  $T$  der Overhead rechnerisch minimieren. Gegenläufig dazu muss aber die Zeitscheibe  $T$  so klein gewählt werden, dass jeder der laufenden Prozesse spätestens dann wieder eine Zeitscheibe erhält wenn der zu steuernde technische Prozess es erfordert.*

- Rechenprozesse laufen sinnvoll nur mit Kommunikation zu anderen Prozessen ab.

*Sie erhalten Information bzw- Nachrichten, die entweder von anderen Rechenprozessen herkommen oder von außen (z.B. Zustandsinformation vom technischen Prozess): Sensoren sind gewissermaßen die Wandler, die Information über physikalische Zustände im technischen Prozess aufnehmen und in Nachrichten umsetzen, die von den Rechenprozessen empfangen werden.*

*Sie geben Informationen/Nachrichten ab, entweder zu anderen Rechenprozessen oder nach außen (z.B. Steuerungsinformation an technische Prozesse). Aktoren sind hier die Wandler, die Nachrichten aus Rechenprozessen in physikalische Signale umsetzen*

- Rechenprozesse sind also kommunizierende "kommunizierende sequentielle Prozesse  
▶ (CSP, Communicating Sequential Processes)".

- Die **Rechenprozesse** eines Prozessrechnersystems laufen

- **auf einem zentralen Prozessrechner**

- **auf mehreren Rechnern verteilt** (verteiltes Prozessrechnersystem)

*Die Kommunikation erfolgt entweder innerhalb eines Rechners oder zu verschiedenen Rechnern. Die Verteilung der Rechenprozesse auf die Rechner erfolgt dabei nach Lastsituation oder nach den benötigten Ressourcen entweder statisch oder dynamisch.*

## 1.2.7 Grundprinzipien der Rechner-Prozesskoppelung

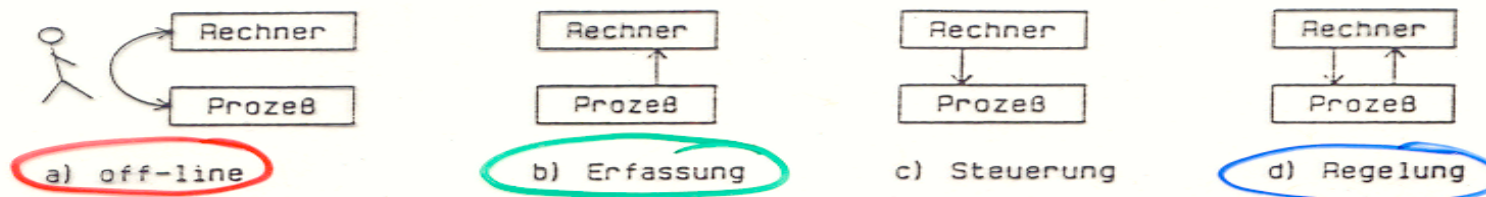


Bild 1.6 Kopplungsarten

- Im off-line-Betrieb transportiert der Mensch die Daten zwischen Rechner und Prozeß.
- Bei der Erfassung von Prozeßdaten dient der Rechner zur Aufbereitung der Daten für den Menschen, der dann regeind in den Prozeß eingreift.
- Bei der Steuerung eines Prozesses durch einen Rechner werden nur Stellwerte für den Prozeß aufbereitet. Voraussetzung dafür ist ein deterministischer Prozeß.
- Bei der Regelung eines Prozesses durch einen Rechner werden die aktuellen Prozeßdaten erfaßt, mit den vorgegebenen Sollwerten verglichen, Stellwerte berechnet und zum Prozeß ausgegeben. Der Mensch kann sich auf die Beobachtung des Prozesses beschränken, und greift nur im Ausnahmefall ein.

- **offline Betrieb** (heute selten) ► indirekt gekoppelter Betrieb. ► Der Mensch transportiert die Daten zwischen Prozess und Rechner

*Beispiel: NC-Maschine alter Machart*

- **online Betrieb (Stand der Technik)** ► direkt gekoppelter Betrieb Rechner und Prozess sind über Prozeßperipherie gekoppelt.

- **offener Betrieb**

- **Datenerfassung** one-way vom Prozess zum PR
  - z.B. Crashrecorder (Flugzeug, Autocrashanlage) Abb1.5
- **Datenausgabe, Steuerung** one-way vom PR zum Prozess
  - CNC Maschinen

- **closed-loop Betrieb**

- **Regelung (Stand der Technik)** geschlossenes digitales Regelsystem, bei dem der PR den Prozess steuert in Abhängigkeit der Messgrößen und Vorgaben.

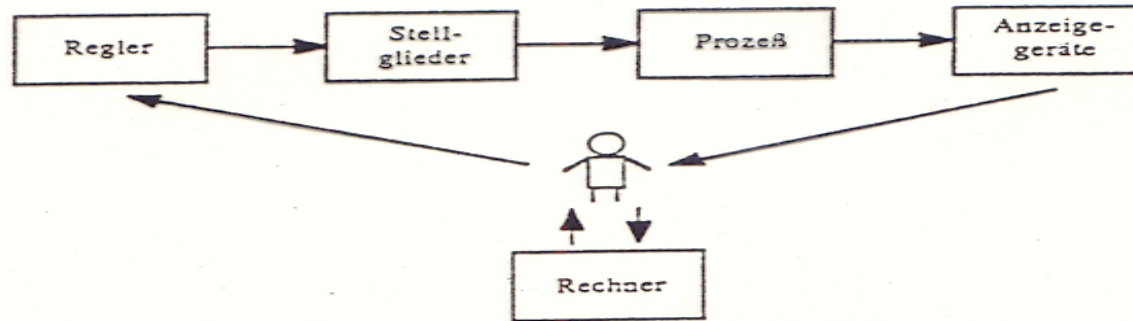
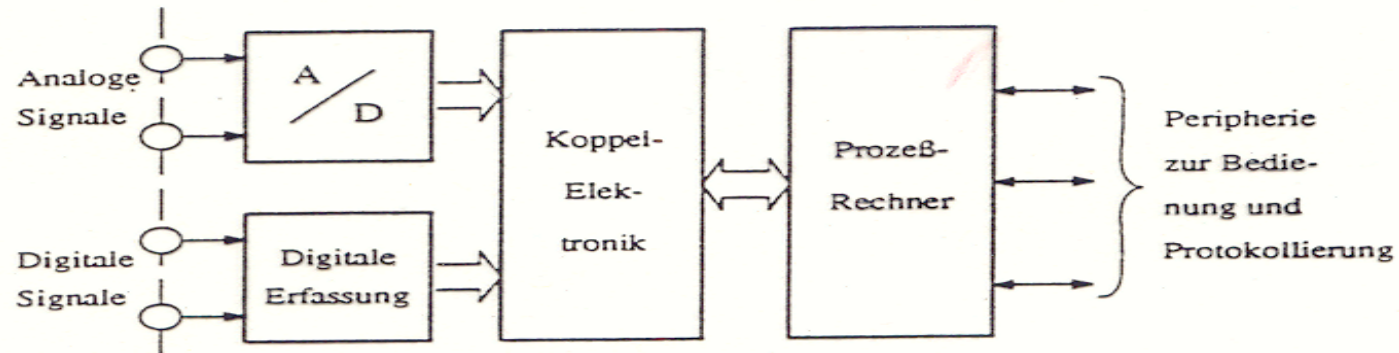
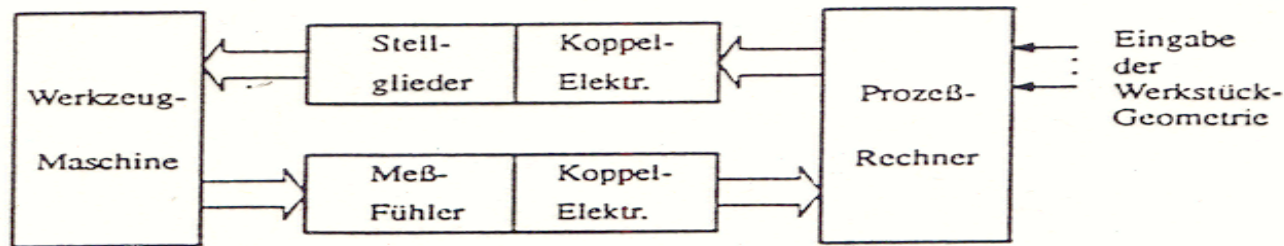


Abbildung 1.15: Indirekte **Off-Line-Rechner-Prozeß-Kopplung** über Datenträger



**Abb. 1.5: Online-Datenerfassung**



**Abb. 1.6: Closed-Loop-Betrieb**

## 1.2.8 Zuverlässigkeit und Sicherheit

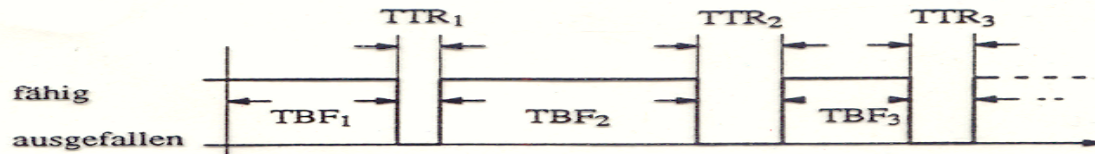
- **Zuverlässigkeit:** Unter Zuverlässigkeit einer technischen Anlage versteht man die Fähigkeit des Systems unter vorgegebener Zeitdauer und zulässigen Betriebsbedingungen die spezifizierte Funktion zu erbringen.
- **Sicherheit:** Die Sicherheit ist das Nichtvorhandensein einer Gefahr für den Menschen, die Umwelt und den Sachwert. Oder „weicher“: “Sicherheit ist eine Sachlage, bei der das Risiko nicht größer als das Grenzkrisiko ist.” [DIN/VDE 31000 Teil2] Als Grenzkrisiko ist dabei das größte, noch vertretbare Risiko zu verstehen [Hal99].

*Sicherheit <> Zuverlässigkeit ► Beispiel: Der Ruhezustand einer Anlage sei sicher, aber in diesem Zustand erfüllt die Anlage ihre Betriebsaufgabe nicht zuverlässig.*

- Die Zuverlässigkeit wird i.d.R. beeinträchtigt durch **Fehler** und gegebenenfalls **Funktionsausfall**.
- An die Zuverlässigkeit von Prozessrechnern werden besondere Anforderungen gestellt: Prozessrechner müssen oft im 7-Tage-24-Stunden-Betrieb funktionsfähig sein.
- ► Funktionsausfall kann dazu führen:
  - damit die gesamte Produktionsanlage ausfällt, womit sehr hohe Kosten verbunden sein können,
  - sich die Produkt-Qualität verschlechtert (oder der Energie- bzw. Rohstoff-Bedarf),
  - hohe Kosten bei der Wiederinbetriebnahme entstehen.
- Es wird immer wirtschaftlicher, hochzuverlässige Prozessrechner einzusetzen, ggf. fehlertolerante Systeme, welche auch beim Ausfall einzelner Komponenten ihre Funktionen weiter erfüllen.
- Es gibt auch sicherheitskritische Anwendungen von Prozessrechnern, deren unkontrollierter Ausfall die Sicherheit von Menschen gefährdet.

*Beispiele hierfür sind Anlagen in der Intensivmedizin (Patientenüberwachung), in der Kernkraftwerks-Überwachung sowie bei der Steuerung von Flugzeugen. Bei sicherheitskritischen Anwendungen spielt der "fail-safe" -Begriff eine besondere Rolle: Bei Erkennung des Ausfalls muss der Prozessrechner das System in einen sicheren Zustand bringen (z.B. das Eisenbahnsignal auf "Halt" stellen). Sicherheitskritische Anwendungen machen zum Teil sehr aufwendige technische Maßnahmen notwendig.*

## Verfügbarkeit (1)



TBF: Time Between Failure  
 TTR: Time To Repair

MTBF Mean TBF  
 MTTR Mean TTR  
 (inkl. Fehlererkennung!)

Abb. 2.69: Zeitverlauf des Systemzustands

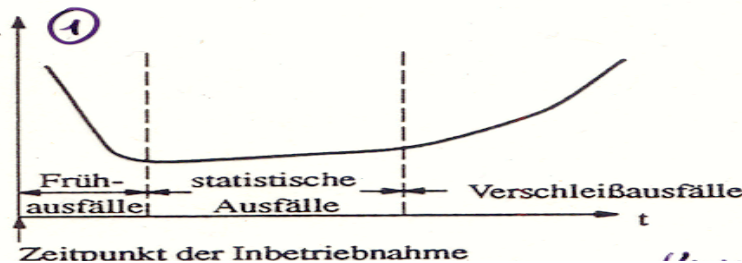
Verfügbarkeit  $p = \frac{MTBF}{MTTR + MTBF}$  und  
 Unverfügbarkeit  $q = \frac{MTTR}{MTTR + MTBF}$   
 mit  $p + q = 1$

} stationärer Fall  
 $p, q \neq f(t)$

- $\lambda = \frac{1}{MTBF}$  Ausfallrate
- $\nu = \frac{1}{MTTR}$  Reparaturrate

### Nicht stationärer Fall:

$q(t)$   
 Unverfügbarkeit



① MTTR ↑ "lange Reprzeit"  
 MTBF ↓ "Früh-ausfälle"  
 "Neue Anlage"

Abb 2.71 Zeitverlauf der Unverfügbarkeit  $q(t)$  „Badewannenkurve“

## Verfügbarkeit (2)

- Annahme: Einzelkomponenten voneinander statistisch unabhängig
- ? Gesamtverfügbarkeit eines aus Einzelkomponenten zusammengesetztes System

### a) Serienschaltung



$K_i$ : Einzelkomponente

$P_i$ : Verfügbarkeit der Einzelkomponente  $K_i$

$q_i$ : Unverfügbarkeit  $K_i$

$$P_{\text{gesamt}} = \prod_{i=1}^n P_i$$

$$\lambda_{\text{gesamt}} = \sum_{i=1}^n \lambda_i$$



$$MTBF_{\text{gesamt}} = \frac{1}{\sum_{i=1}^n \frac{1}{MTBF_i}}$$

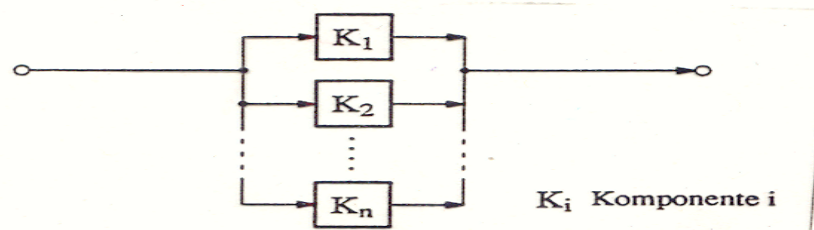
$P_i \approx 1$  !

$q_i \ll 10^{-2}$

$$P_{\text{gesamt}} = 1 - q_{\text{gesamt}} = \prod_{i=1}^n (1 - q_i) \approx 1 - \sum_{i=1}^n q_i$$

oder :  $q_{\text{gesamt}} \approx \sum_{i=1}^n q_i = n \cdot q_i$  ! "Serie"

### b) Parallelschaltung



$q_{\text{gesamt}} = \prod_{i=1}^n q_i$  ! "parallel"

$P_{\text{gesamt}} = 1 - q_{\text{gesamt}} = 1 - \prod_{i=1}^n q_i$

## Fehlertolerante Prozeßrechnersysteme (PZR) zur Erhöhung der Verfügbarkeit (1)

- Ein Fehler soll hier verstanden werden als: die Abweichung des Systems oder Teile des Systems vom gewünschten -geplanten- Verhalten
  - ==> Konsequenz: Zuverlässigkeit der Anlage wird negativ beeinträchtigt
  - ==> 1. Wunsch: Fehlervermeidung ('Perfektion') wenn möglich
  - ==> 2. Wunsch: **Fehlertoleranz** ('Redundanz') wenn 1.Wunsch nicht erfüllbar
- **Fehlertolerant** bedeutet, dass das PZR seine Funktion, den zu steuerenden technischen Prozeß zu bedienen, weiterhin erfüllt, obwohl ein Fehler aufgetreten ist. Voraussetzungen:
  1. **Ein Fehler wird sicher als solcher erkannt (Fehlerdiagnose,Fehlererkennung)**
  2. **Es ist Redundanz (==>Ersatz) vorhanden**, um den Fehler zu kompensieren ==> Vergleiche 'Parallelschaltung' im Zuverlässigkeitsersatzschaltbild
  3. In Folge einer Fehlererkennung kompensiert der Ersatz den erkannten Fehler, durch Erbringung der gewünschten (Teil-)Funktion

- **Fehlertypen:**
  - **permanente Fehler** ==> Fehler tritt auf und bleibt dauernd bestehen (gut zu finden!)
  - **transiente Fehler** ==> Fehler tritt auf und verschwindet wieder (10 Mal häufiger als permanente Fehler, oft schwierig zu finden!)
- Kategorien von **Fehlern:**
  - Entwurfsfehler
    - Spezifikationsfehler
    - Implementierungsfehler
    - Dokumentationsfehler
    - ==> Minimierung durch Entwicklung sog. diversitären Lösungen (2 EW-Teams)
  - Herstellung- oder physikalische Fehler (Altern, Verschleiß, Überlastung....)
    - ==> Minimierung von Herstellungsfehlern durch Qualitätssicherung
  - Betriebs- und Bedienfehler
    - (unvorhergesehene) Störungen der Prozeßumwelt (EMV, mechanische thermische Einflüsse...)
    - Bedienfehler (menschliches Versagen)
    - Wartungsfehler
    - ==> Minimierung der Bedienfehler durch intensives Training des Personals mittels 'Procedures'



## Fehlertolerante Prozeßrechnersysteme (PZR) zur Erhöhung der Verfügbarkeit (2)

- **Die zu erkennenden Fehler und die daraus resultierenden Verfahren zur Erkennung** derselben werden zur Entwurfszeit des PZR definiert ==> Erstellung eines **Diagnosemodells (FMEA Failure Mode + Effect Analysis)**
- Die **Fehlertoleranz** wird ebenfalls zur Entwurfszeit des PZR eingebaut.
- **Fehlererkennung durch:**
  - zyklische Hard- und/oder Softwaretests
  - Hardware-Voter ( 2 aus 3 , 3 aus 5) ==> Probleme: Verbindungsleitungen, Hardware des Voters muss auch redundant sein
  - Software-Voter: Ergebnisvergleich nach redundanter Berechnung derselben Aufgabe in unterschiedlichen Softwareprozessen auf dem gleichen Rechner oder unterschiedlichen Rechnern
  - Assertions (Zusicherungen): Bei vorab bekannten Zwischenergebnissen ein Vergleich mit den aktuell berechneten Ergebnissen

- Arten der **Redundanz**:
  - dynamische Redundanz: Der Ersatz übernimmt erst dann die Funktion wenn ein Fehler aufgetreten ist, ansonsten steht er für andere Aufgaben zu Verfügung (==> z.B. Entwicklungsrechner...)
    - 1. Problem: sichere Fehlererkennung (==> Assertions, Voter nicht einsetzbar)
    - 2. Problem: Einphasen des Ersatzsystems
      - ==>Backward-Recovery-Systeme: Es gibt definierte Checkpoints in definierten zeitlichen Abständen an denen gewiss ist, dass das System i.O. ist. An diesen Checkpoints wird der gesamte interne Zustand so gesichert, dass der zuletzt erreichte Checkpoint im Fehlerfall als Anlaufzustand des Ersatzsystems dient.
  - statische Redundanz: Es gibt quasi keinen Unterschied zwischen Ersatz und Hauptsystem. Beide bearbeiten parallel dasselbe:
    - räumlich parallel ==> mehrfach vorhandenen Hardware
      - ==> SRU-Konzept (smallest replaceable unit) Wenn ein Fehler erkannt wird, dann wird die betroffene defekte Hardware (manuell) durch Wechseln der betreffenden SRU ersetzt. (SRU kann sein: Bauteil,Subsystem,System...)
    - zeitlich sequentiell ==> derselbe Bearbeitungsablauf wird zeitlich hintereinander -in Hardware oder/und Software- wiederholt

## Sicherheitsaspekte

- **Vor der Verfügbarkeit geht die Sicherheit der Anlage**
- Wird ein sicherheitskritischer Fehler erkannt ==> das System muss -falls vorhanden- in den "fail-safe" Zustand gebracht werden (=Sicherheitsbetrieb) ==> Reduktion der Verfügbarkeit des Gesamtsystems denn:
- Fehlererkennung ist nur bei vorhandener Redundanz möglich ==> mehr Hardware, mehr Fehlerquellen, höhere Unverfügbarkeit insgesamt
- Wie beeinflussen Sicherheitsüberlegungen die Verfügbarkeit der Anlage?
- Gegeben: 3-fach redundante Hardware  $q_{hw} \ll 10^{-4}$ , idealer Voter ( $q=0$ ).
- 1. **Sicherheitsbetrieb:** Tritt ein Fehler auf -d.h. eine Hardware liefert ein anderes Ergebnis- wird die Anlage in den 'fail-safe' Zustand gebracht

Unverfügbarkeit:  $q_{ges} = 1 - p_{ges} = 1 - (1 - q_{hw})^3$  ; mit  $(q_{hw} \ll 10^{-4}) \approx 3 * q_{hw}$  (Sobald einer ausfällt, fällt das Gesamtsystem aus)

Wahrscheinlichkeit eines unsicheren Zustands (unerkannter Fehler)  $W_a$ : Alle 3 Hardwaren fallen zeitgleich aus **und** produzieren den gleichen Fehler ==> Voter findet keine Abweichung trotz Fehler ==> Betrieb läuft fälschlicherweise weiter.

$$W_{as} \leq q^3 \quad q_{hw} \approx 10^{-4} \quad W_{as} \approx 10^{-12}$$

## 2. TMR-Betrieb (Triple Modular Redundancy) : Mehrheitsentscheidung 2-aus-3

Unverfügbarkeit:  $q_{\text{ges}} = q^3 + 3 * (q^2 * (1-q))$   
 $q^3$       Wahrscheinlichkeit alle ausgefallen  
 $q^2 (1-q)$  Wahrscheinlichkeit zwei ausgefallen, einer funktioniert

$W_a$ : 3 oder 2 Hardwaren ausgefallen und liefern gleiches falsches Ergebnis ==> dann bleibt der Fehler unerkannt und ggf. die einzig fehlerfrei arbeitende Hardware wird weggeschaltet.

$$W_{\text{atmr}} \leq q^3 + 3 * (q^2 * (1-q)) ; \quad W_{\text{atmr}} = 10^{-12} + 3 * 10^{-8} * 0.9999 \approx 10^{-7}$$

## 3. Höchste Verfügbarkeit: Solange noch eine Hardware funktioniert wird weitergemacht.

Unverfügbarkeit:  $q_{\text{ges}} = q_{\text{hw}}^3$  (Erst wenn alle ausgefallen sind, fällt das Gesamtsystem aus)

$W_{\text{av}}$ : Der Voter entscheidet sich für den fehlerhaften Rechner als richtigen Rechner wenn ein Fehler auftritt.

$$W_{\text{av}} \leq 3*q$$

## 1.3 Prozessbeschreibung (-modelle) und Systembetrachtung

- Um einen Prozess steuern bzw. regeln zu können wird ► die **Prozessbeschreibung** und das **Prozessmodell** benötigt
- Ein Prozessmodell wird erstellt, um folgendes aufzuzeigen und zu überprüfen:
  - Prozessstruktur
  - Zusammenhänge: Eingangsdaten, Ausgangsdaten und Zustandsdaten
- Mögliche Modellformen:
  - Gegenständliches Modell (im verkleinerten Maßstab oder in Zeitlupe)  
*Beispiele: Abläufe und Zustände durch besondere Vorrichtungen erfasst und verdeutlicht werden. z.B. Rauchfahnen im Windkanal, polarisiertes Licht bei der Materialuntersuchung.*
  - abstrakten oder mathematischen Modell  
*Beispiel: mathematische Gleichungen, numerische Tabellen, Petrinetze, Zustandsereignisdiagramme, SDL, Flusspläne oder andere Diagramme*
  - Rechnermodell: Simulation mit einem Computer  
*Beispiel: Simulation mittels eines Programms, das wiederum auf einem mathematischen Modell beruht, den Prozess simuliert und das sich dabei wie ein gegenständliches Modell verhält.  
Beispiel: Graphische Animation eines Fertigungsablaufs: Montage, Lagervorgänge etc. mit "echten" Modellparametern ► quasi Nachstellen der Realität ► Möglichkeit der Ermittlung der Gesamtleistung einer Anlage. ► (kostengünstige) Simulation von Fehlerfällen*

## 1.3.1 Prozessbeschreibung

- **Vorgang der Erstellung: Strukturierung** des Prozesses ► **Zerlegung** in Teilprozesse ► Verfeinerung bis zu **Elementarprozessen** (► entspricht der strukturierten Programmierung)
- ► **Ergebnis: Verteilung der Beschreibung** des Gesamtprozesses **auf die Elementarprozesse selbst und deren Zusammenhänge** untereinander.
- **Elementarprozesse** werden meist **mittels mathematischer** Modelle beschrieben: Gleichungen über technische, chemische oder physikalische Vorgänge.
- **Prozessstrukturen** werden meist **durch graphische Methoden** beschrieben:
  - *Sie müssen meist den statischen, stationären und dynamischen Strukturen des Prozesses angepasst werden.*
- **statische Strukturen** des Prozesses durch **Blockschaltbilder**
  - *beschreibt Grobstruktur: ► Konstruktion, Darstellungselemente sind dem Einsatzgebiet angepasst (Verfahrenstechnik, Energietechnik, Elektrotechnik...) oft kein Rückschluss auf die Funktion entnehmbar.*
  - Zu beschreibende Zusammenhänge: Prozessgrößen, Eingangs-, Ausgangs- und Zustandsgrößen*
- **stationäre Abläufe des Prozesses**
  - *Zu beschreibende Zusammenhänge: Für den eingeschwungenen Fall (Arbeitspunkt) die Zusammenhänge der Prozessgrößen.*
  - meist im Blockschaltbilder als Wirkungs- oder Transportwege eingezeichnet.
  - Flussplan, Struktogramm !! Unterschied zu Programmierung: Parallelität der Aktionen !!
  - Netzpläne (DIN69 900), Netze aus Instanzen und Kanälen (später)
- **dynamische Abläufe**
  - Zustandsdiagramme ► Automobilbau "Statemate, Rhapsody, Matlab/Simulink"
  - PETRI-Netze, ► UML 2.x (Unified Modelling Language) ► SDL Structure and Description Language ► Telekommunikation
  - Zu beschreibende Zusammenhänge: Ein- und Ausschaltvorgänge, zeitlichen Veränderungen und Zusammenhänge beim Arbeitspunktwechsel (kann auch durch plötzliche innere oder äußere Störgrößenaufschaltung geschehen!!)*
  - Beispiel: Im Triebwerksbau ist bekannt, dass das Hochfahren eines Jet-Triebwerks schnell und definiert durchgeführt werden muss, da kritische Schwingungs-Systemzustände durchfahren werden müssen, die sicher zur Zerstörung führen würden. Ausschließlich der*

*definierte Arbeitspunkt ist quasi stationär zu beschreiben. Der Ein- und Ausschaltvorgang kann nur empirisch und dynamisch beschrieben werden.*

### 1.3.2 Prozesserkennung (Modell-Bildung)

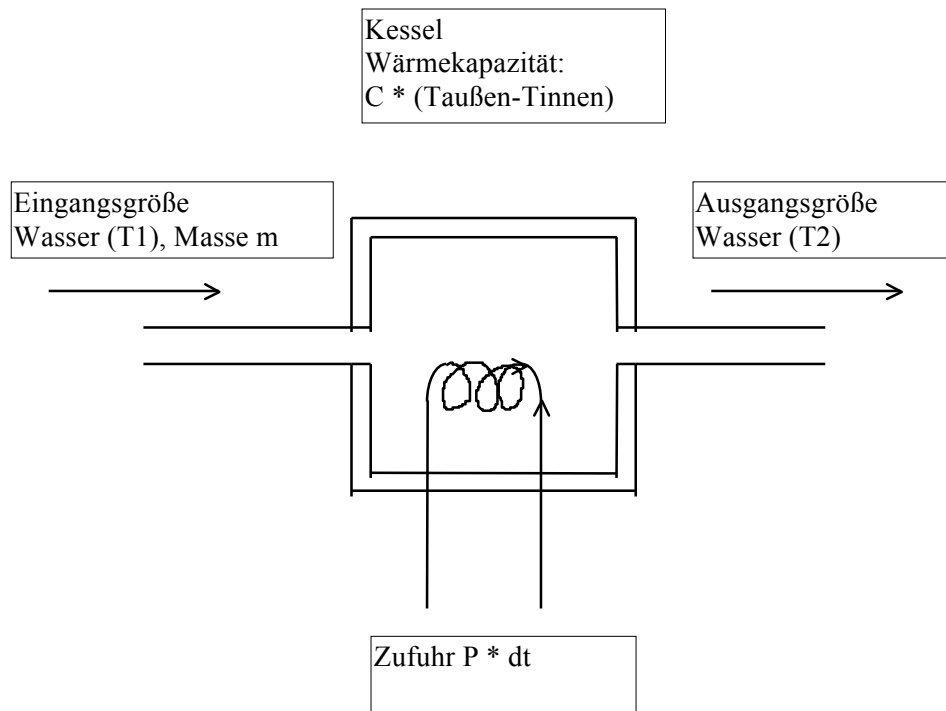
- Die Erstellung eines (abstrakten) Modells bezeichnet man als **Prozesserkennung**
- Zwei unterschiedliche Methoden:
  - **empirische Methode**  
Gegeben i.d.R. der fertig realisierte technische Prozess, der **nachträglich** automatisiert werden soll  
Die Prozessbeschreibung erfolgt **empirisch** an Hand der Erfahrung.  
(=> bei Verwendung der Prozessdaten zur Ermittlung der Anlagedaten spricht man von **Prozessidentifikation**)
  - **analytische Methode**  
Ablauf wird durch Gleichungen beschrieben =>Automatentheorie  
Ergebnis: analytisches Prozessmodell. Alle Prozessgrößen analytisch zu jedem Zeitpunkt berechenbar.
- **Adaptive Prozessmodelle:**
  - Im Modell werden die unbekannt Prozessgrößen (insbesondere Störgrößen) im laufenden Betrieb erfasst und als anpassbare Parameter in die Berechnungen einbezogen werden.
  - Problem: =>unbekannte Prozessgrößen schlecht vorhersagbar aber die Kompensation leicht im Modell zu beschreiben.

*Beispiel: automatische Spracherkennung. Aufgabe sprecherunabhängig Sprache in Schriftzeichen zu übersetzen.  
Problem: Das Erkennen basiert auf Mustervergleichsmethoden. Die vorgegebenen Muster sind die eines Sprechers.*

- **lernende Prozessmodelle** bezeichnet man als solche:
  - => die in lernende Programme eingebaut sind, die im Betrieb nicht nur passiv die Parameter auf Grund von Prozessdaten bestimmen, sondern die optimalen Werte aktiv selber suchen.
  - => Diese Art von Modellbildung gehört in das Gebiet der künstlichen Intelligenz und ist im praktischen Betrieb sehr selten zu finden.
- Die Auswahl der Methode zur Erstellung eines Modells richtet sich nach dem Prozess und der Eignung für die jeweilige Prozessbeschreibungsebene.



• **Beispiel:**



**Prozesserkennung:**

1. Elementarprozeß, Aufheizen von Wasser durch elektrische Energie;

1a. Ablauf:

- ▶ Kessel wird gefüllt, kein Abfluss
- ▶ heizen, Eingang- und Ausgang geschlossen
- ▶ Kessel wird geleert kein Zufluss.

2. Merkmale: Umformung von Energie, als Chargenprozess, deterministisch;

3. Prozesserkennung:

empirisch: "Zufuhr von elektrischer Energie erhöht die Temperatur." (vereinfacht!!)

analytisch:  $W = c * m * (T_{\text{ende}} - T_{\text{anfang}}) = P * t [J]$

Prozessmodell, hier: statisches Modell mit Prozessgrößen:

Eingangsgrößen: Wassermenge ( $m$  [kg]), Wassertemperatur ( $T_1$ ), Elektrische Leistung ( $P$ ) Heizzeit ( $dt$ );

Parameter: spezifische Wärme von Wasser ( $c$ ) 4,18 [kJ/(kg\*K)]

Störgrößen: Wärmekapazität des Behälters ( $C$ ) und dessen Temperatur ( $T_k$ ) ▶ Wärmeverluste durch Wärmeleitung und -strahlung des Behälters

Prozess-Zustandsgrößen: Temperatur  $T$  des Wassers im Behälter

Prozess-Ausgangsgrößen: Wassertemperatur ( $T_2$ )

- stationäres Modell (ohne Wärmeverluste):

$$\text{aus } P * dt = c * m * (T_2 - T_1) + C * (T_2 - T_k)$$

$$\text{folgt } T_2 = (P * dt + c * m * T_1 + C * T_k) / (c * m + C)$$

- dynamisches Modell:

*Es beschreibt die Ausbreitung der Wärmeenergie und der Temperaturverteilung im Verlaufe der Zeit innerhalb des Behälters. Dazu müssen Wärmeleitung und -konvektion durch Differentialgleichungen höherer Ordnung berechnet werden. => hier nicht!*

- *Fazit Beispiel: Großer Aufwand, der für die Erstellung eines zuverlässigen Prozessmodells*
- In der Praxis ist der Normalfall dass das neu erstellte Modell noch unzulänglich ist und das Prozessverhalten nur unzureichend beschrieben wird.
  - ► häufige Ursache: Störgrößen nicht genügend genau bekannt oder unvollständig beschrieben.
  - ► Folge: iterative Verbesserung des Modells zu jeder Zeit:
- Standard-Vorgehen : analytische, numerische Untersuchungen oder Simulation am Computermodell der Auswirkung von kleinen Änderungen der Eingangsgrößen auf den Prozess.
  - => Problem: Die unvermeidlichen Vereinfachungen durch das eingesetzte Modell verdecken u.U. Prozesseffekte die dann im Betrieb auftreten und gravierende Auswirkungen auf den Prozess haben können
  - => wie minimiert man das Problem in den Fällen in denen keine Iteration am realen Prozess erfolgen kann?: Verbesserung auf Grund eines Vergleichs eines gegenständlichen Modells und einer Computermodell-Simulation -> dadurch sind 2 verschiedene Modelle vorhanden
  - => Annahme dabei: verschiedene Verfahren werden auch verschiedene Abweichungen ergeben

## Inhaltsverzeichnis Kapitel PDV K

K 1 Einführung in die Prozessdatenverarbeitung(PDV) und Begriffe .....	1
1.1 Einführung und Geschichte der PDV .....	1
1.1.1 Ziel der Prozessdatentechnik (PDV)-Vorlesung, oder was ist Prozessdatentechnik? .....	7
1.1.2 Allgemein: Automatisierung von Prozessen .....	9
1.1.3 Ziele der Automatisierung in der PDV .....	11
1.2 Begriffe und deren Bedeutung aus der PDV .....	13
1.2.1 Prozess allgemein, Unterscheidung nach Verarbeitungsart und Verarbeitungsgut.....	13
1.2.2 Der technische Prozess .....	16
1.2.3 Klassifikationsmerkmale von technischen Prozessen .....	18
1.2.3.1 Unterscheidung nach Verarbeitungsstrukturen.....	18
1.2.3.2 Unterscheidung nach der inneren Ablaufstruktur .....	21
1.2.3.3 Unterscheidung nach Einsatzgebiet .....	21
1.2.4 Prozessrechner (PR).....	22
1.2.5 Prozessdaten und Prozessgrößen.....	25
1.2.6 Rechenprozess, Task und Multitasking.....	30
1.2.7 Grundprinzipien der Rechner-Prozesskoppelung.....	34
1.2.8 Zuverlässigkeit und Sicherheit.....	36
1.3 Prozessbeschreibung (-modelle) und Systembetrachtung .....	45
1.3.1 Prozessbeschreibung.....	46
1.3.2 Prozesserkennung (Modell-Bildung).....	48