Systemprogrammierung

Prozessverwaltung: Einplanungskriterien

Wolfgang Schröder-Preikschat

Lehrstuhl Informatik 4

10. Mai 2011

Gliederung

Prozessorzuteilungseinheit: Faden (engl. *thread*)

Einplanungseinheit (engl. *unit of scheduling*) für die Vergabe der CPU

Ablaufplanung von Fäden erfolgt **betriebsmittelorientiert** und ist ggf. **ereignisgesteuert** oder **zeitgesteuert**

- die Laufbereitschaft eines Fadens hängt von der Verfügbarkeit all jener Betriebsmittel ab, die für seinen Ablauf erforderlich sind
- die Bereitstellung von Betriebsmitteln (ggf. durch andere Fäden) kann die sofortige Einplanung von Fäden bewirken
- oder die Einplanung erfolgt in fest vorgegebenen Zeitintervallen

Vorgänge, die ent- oder gekoppelt (zeitversetzt/zeitgleich) sein können

Einplanung eines Fadens ist nicht gleichzusetzen mit **Einlastung**:

- Einplanung ist der Vorgang der Reihenfolgenbildung von Aufträgen
- Einlastung ist der Moment der Zuteilung von Betriebsmitteln

Stoßbetrieb (engl. *burst mode*)

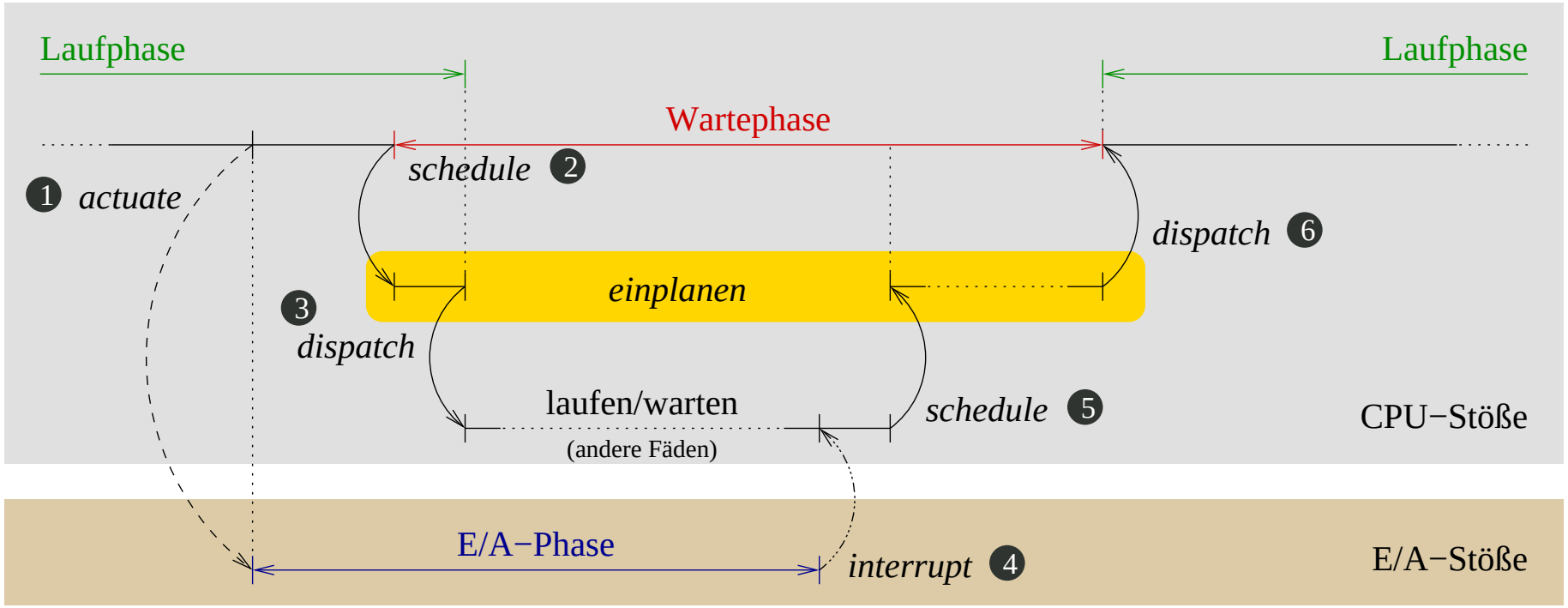
Laufphase: **CPU-Stoß** (engl. *CPU burst*)

- aktive Phase eines Fadens (auch: Rechenphase)
  - alle zur Ausführung erforderlichen Betriebsmittel sind verfügbar
- der Faden ist eingelastet, ihm wurde die CPU zugeteilt

Wartephase: **E/A-Stoß** (engl. *I/O burst*), im weitesten Sinn

- inaktive Phase eines Fadens (auch: E/A-Phase)
  - nicht alle zur Ausführung erforderlichen Betriebsmittel sind verfügbar
- Ein-/Ausgabe abwarten bedeutet, auf Betriebsmittel zu warten
  - konsumierbare Betriebsmittel: Eingabedaten, Nachrichten, Signale
  - wiederverwendbare Betriebsmittel: Puffer, Geräte, . . . , die CPU
- die Betriebsmittel werden letztlich durch andere Fäden bereitgestellt
  - ein E/A-Gerät kann dabei als „externer Faden“ betrachtet werden

Lauf-, E/A- und Wartephasen von Fäden



- ein Faden läuft physisch solange, bis er den Prozessor freiwillig abgibt[a]
- logisch kann er sich zuvor jedoch bereits in der Wartephase befinden

[a] Fäden geben den Prozessor grundsätzlich immer von selbst ab. Sie können dazu gezwungen werden, dies dann auch zu gegebener Zeit zu tun ⇝ Verdrängung (S. 18).

Lauf-, E/A- und Wartephasen von Fäden (Forts.)

Fäden durchlaufen (im BS) einen Kontrollfluss zur **Einplanung** und **Einlastung** anderer Fäden:

1. der laufende Faden stößt einen E/A-Vorgang an (*actuate*)
2. er wartet passiv auf die Beendigung der Ein-/Ausgabe (*schedule*)
   - Anforderung eines wiederverwend-/konsumierbaren Betriebsmittels
3. und lastet einen eingeplanten, laufbereiten Faden ein (*dispatch*)
4. die Beendigung der Ein-/Ausgabe wird signalisiert (*interrupt*)
   - Bereitsstellung des konsumierbaren Betriebsmittels „Signal“
5. der auf dieses Ereignis wartende Faden wird eingeplant (*schedule*)
6. der Faden wird eingelastet, sobald er an der Reihe ist (*dispatch*)

Sonderfall: ein Faden plant und lastet sich selbst ein, wenn sich die CPU mangels anderer laufbereiter Fäden im Leerlauf (engl. *idle state*) befindet

Fäden als Mittel zum Kaschieren von Totzeiten

Arbeitsteilung in nebenläufigen/parallelen Systemen

Überlappung von Lauf- und Wartephasen erhöht die Rechnerauslastung

- die Wartephase eines Fadens als Laufphase anderer Fäden nutzen
- die Stöße anderer Fäden zum „Auffüllen“ von Wartephasen nutzen

Auslastung von CPU und Peripherie (E/A-Geräte) steigert sich

- eine CPU kann zu einem Zeitpunkt nur einen CPU-Stoß verarbeiten
- parallel dazu können jedoch mehrere E/A-Stöße laufen
  - ausgelöst während eines CPU-Stoßes: in der Laufphase eines Fadens wurden mehrere E/A-Vorgänge gestartet
  - ausgelöst von mehreren CPU-Stößen: die Wartephase eines Fadens wurde mit Laufphasen anderer Fäden gefüllt
- Folge: CPU und E/A-Geräte sind andauernd mit Arbeit beschäftigt

- bei weniger Prozessoren als Fäden, sind Fäden zu serialisieren

Zwangsserialisierung von Programmfäden

In Bezug auf eine Instanz des Betriebsmittels „CPU“

Verlängerung der absoluten Ausführungsdauer später „eintreffender“ laufbereiter Fäden ist zu beobachten:

- Ausgangspunkt seien n Fäden mit gleichlanger Bearbeitungsdauer k
- der erste Faden wird um die Zeitdauer 0 verzögert
- der zweite Faden um die Zeitdauer k, der i-te Faden um $(i-1)\cdot k$
- der letzte von n Fäden wird verzögert um $(n-1)\cdot k$

Mittlere Fadenverzögerung

$$\frac{1}{n}\cdot\sum_{i=1}^{n}(i-1)\cdot k = \frac{n-1}{2}\cdot k$$

Vergrößerung der mittleren Verzögerung ist proportional zur Fadenanzahl

Subjektive Empfindung der Fadenverzögerung

Nur bis zu einer bestimmten Last (# eingeplanter Fäden) ...

Startzeiten von Fäden verzögern sich im Mittel um: $\frac{n-1}{2} \cdot t_{cpu}$

- mit t_{cpu} gleich der mittleren Dauer eines CPU-Stoßes
- sofern $t_{cpu} \geq t_{ea}$, der mittleren Dauer eines E/A-Stoßes
- die Praxis liefert als Regelfall jedoch ein anderes Bild: $t_{cpu} \ll t_{ea}$

Wartephasen bei E/A-Operationen dominieren die Fadenverzögerung

- zwischen CPU- und E/A-Stößen besteht eine große Zeitdiskrepanz
- der proportionale Verzögerungsfaktor bleibt weitestgehend verborgen
- er greift erst ab einer bestimmten Anzahl von Programmfäden
  - nämlich wenn zu einem Zeitpunkt gilt: $\sum_{i=1}^{m} t_{cpu}^{i} > \sum_{j=1}^{n} t_{ea}^{j}$
- sehr häufig ist die Fadenverzögerung daher nicht wahrnehmbar

Beachte: **Überlast** durch zuviel eingeplante Fäden ist zu **vermeiden**

- akkumulierte Länge der CPU-Stöße der jew. E/A-Auslastung anpassen

Gliederung

Dauerhaftigkeit von Zuteilungsentscheidungen

Logische Ebenen der Prozesseinplanung

langfristige Einplanung (engl. *long-term scheduling*) [s – min]

- **Lastkontrolle**, Grad an Mehrprogrammbetrieb einschränken
- Programme laden und/oder zur Ausführung zulassen
- Prozesse der mittel- bzw. kurzfristigen Einplanung zuführen

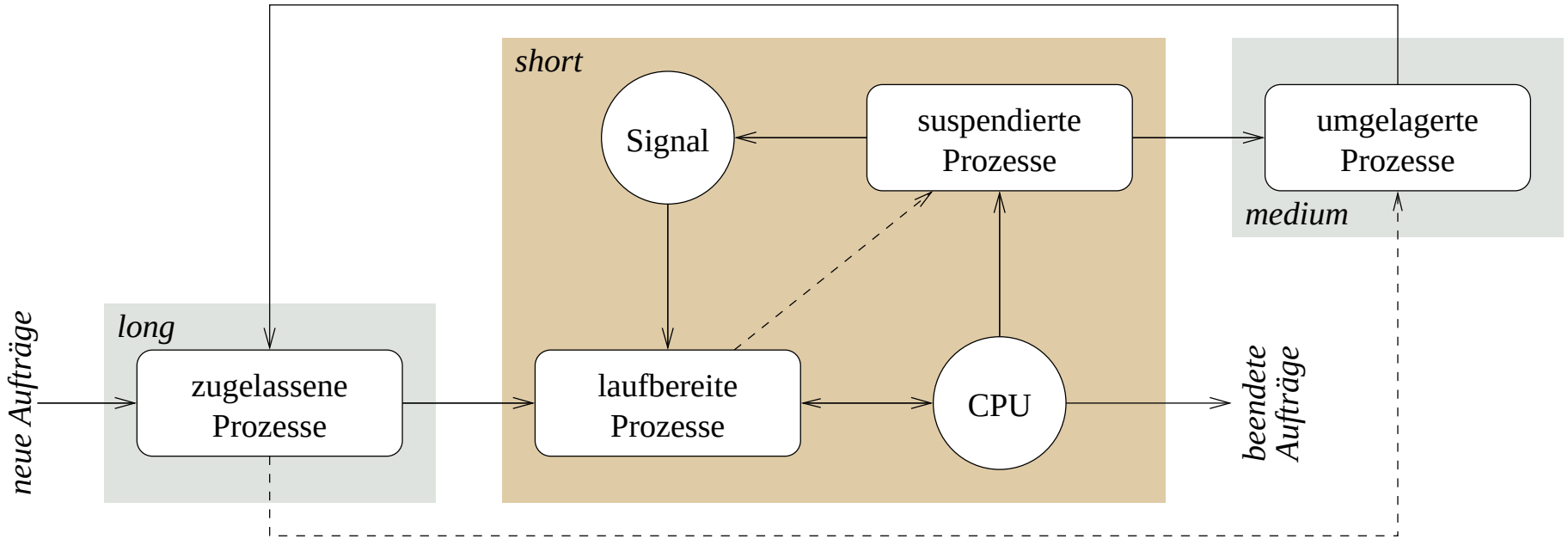
mittelfristige Einplanung (engl. *medium-term scheduling*) [ms – s]

- Teil der **Umlagerungsfunktion** (engl. *swapping*)
- Programme vom Hinter- in den Vordergrundspeicher bringen
- Prozesse der langfristigen Einplanung zuführen

kurzfristige Einplanung (engl. *short-term scheduling*) [μs – ms]

- **Einlastungsreihenfolge** der Prozesse festlegen — obligatorisch

Phasen der Prozesseinplanung



Voraussetzung für Mehrprozessbetrieb ist die kurzfristige Einplanung

- laufbereite Prozesse erwarten die Zuteilung des wiederverwendbaren Betriebsmittels „CPU“, d.h. den Start ihrer Laufphase
- suspendierte Prozesse erwarten die Zuteilung eines konsumierbaren Betriebsmittels „Signal“, d.h. das Ende ihrer Wartephase

Prozesszustand vs. Einplanungsebene

Prozesse haben in Abhängigkeit von der Einplanungsebene (S. 11) zu einem Zeitpunkt einen **logischen Zustand**:

kurzfristig (engl. *short-term*)
- bereit, laufend, blockiert

mittelfristig (engl. *medium-term*, *mid-term*)
- schwebend bereit, schwebend blockiert

langfristig (engl. *long-term*)
- erzeugt, gestoppt, beendet

Anmerkung

Anwendungsfälle legen fest, welche der Einplanungsebenen von einem Betriebssystem wirklich zur Verfügung zu stellen sind, nicht umgekehrt.

Kurzfristige Einplanung

Festlegung der Prozessorzuteilungsreihenfolge

Betriebssystem bietet **Mehrprozessbetrieb** (engl. *multi-processing*) auf Basis der **Serialisierung von Programmfäden**:

bereit (engl. *ready*) zur Ausführung durch den Prozessor (die CPU)
- der Prozess ist auf der Bereitliste (engl. *ready list*)
- das Einplanungsverfahren bestimmt die Listenposition

laufend (engl. *running*), erfolgte Zuteilung des Betriebsmittels „CPU“
- der Prozess vollzieht seinen CPU-Stoß
- zu einem Zeitpunkt pro CPU nur ein laufender Prozess

blockiert (engl. *blocked*) auf ein bestimmtes Ereignis
- der Prozess erwartet die Zuteilung eines Betriebsmittels
  - mit Ausnahme des Betriebsmittels „CPU“
- ggf. vollzieht der Prozess auch seinen E/A-Stoß

Spezialfall: „blockiert“ $\mapsto$ „laufend“ $\rightsquigarrow$ voll verdrängend (S. 19)

Mittelfristige Einplanung

Festlegung der Umlagerungsreihenfolge

Betriebssystem implementiert die **Umlagerung** (engl. *swapping*) von kompletten Programmen bzw. logischen Adressräumen:

schwebend bereit (engl. *ready suspend*)

- Adressraum des Prozesses ist ausgelagert
  - verschoben in den Hintergrundspeicher
  - „*swap-out*“ ist erfolgt
  - „*swap-in*“ wird erwartet
- die Einlastung des Prozesses ist außer Kraft
  - genauer: aller Fäden des Adressraums

schwebend blockiert (engl. *blocked suspend*)

- ausgelagerter ereigniserwartender Prozess
- Ereigniseintritt $\mapsto$ „schebend bereit“

Variante: „schwebend bereit“ $\mapsto$ „bereit“ $\rightsquigarrow$ langfristige Einplanung

Langfristige Einplanung

Festlegung der Zulassungsreihenfolge

Betriebssystem verfügt über Funktionen zur **Lastkontrolle** und steuert den Grad an Mehrprogrammbetrieb:

erzeugt (engl. *created*) und fertig zur Programmverarbeitung
- Prozess ist instanziiert, Programm wurde zugeordnet
- ggf. steht die Speicherzuteilung jedoch noch aus

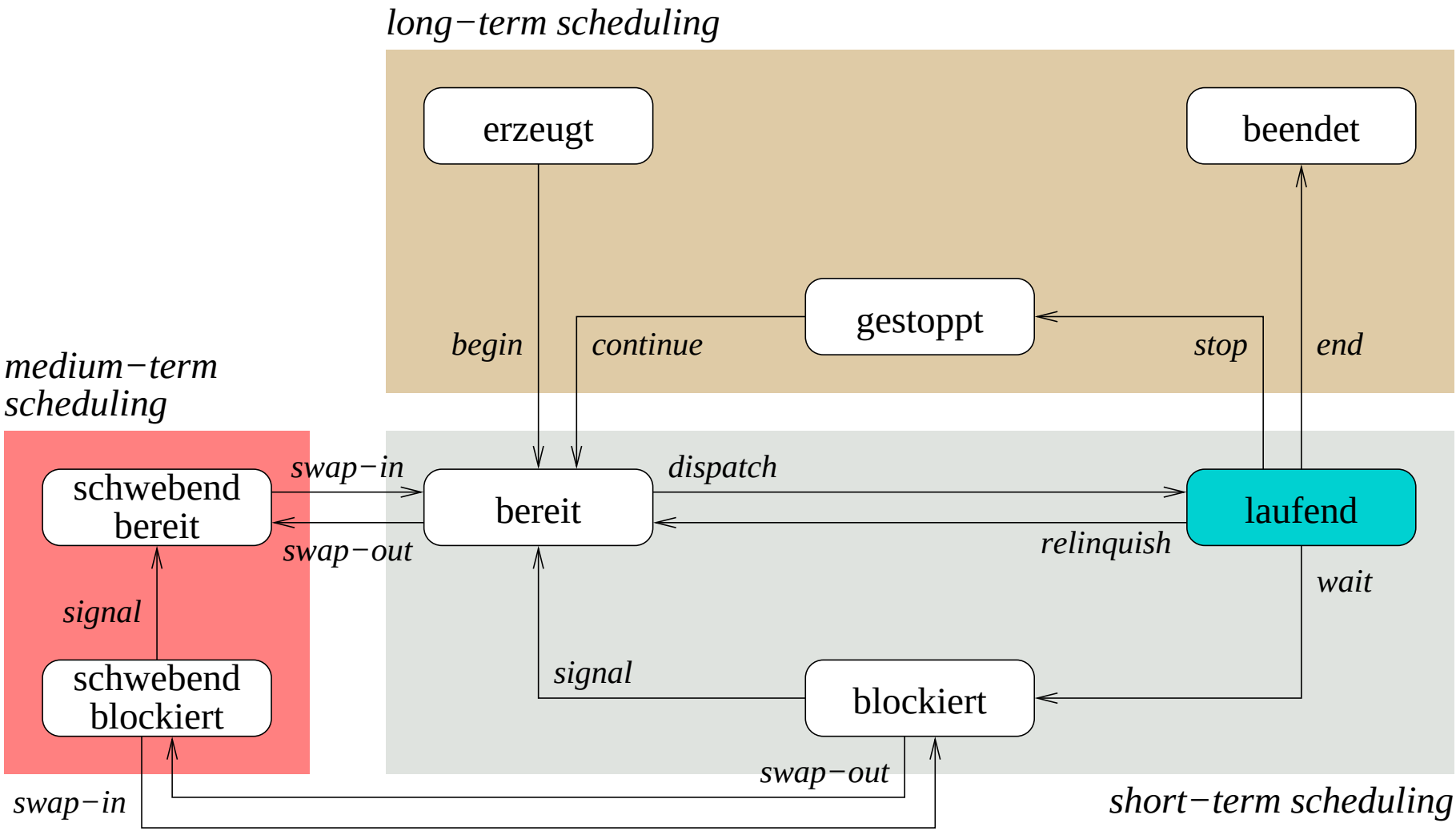
gestoppt (engl. *stopped*) und erwartet seine Fortsetzung
- Prozess wurde angehalten (z.B. ^Z bzw. `kill(2)`)
- Gründe: Überlast, Verklemmungsvermeidung, ...

beendet (engl. *ended*) und erwartet seine Entsorgung
- Prozess ist terminiert, Betriebsmittelfreigabe erfolgt
- ggf. muss ein anderer Prozess den „Kehraus“ vollenden

Achtung: „gestoppt“ werden können auch bereite/blockierte Prozesse

Abfertigungszustände im Zusammenhang

Je nach Betriebssystem/-art sind weitere „Zwischenzustände“ anzufinden



Einplanungs-/Auswahlzeitpunkt

Übergänge in den Zustand „bereit“ aktualisieren die Bereitliste:

- eine Entscheidung über die Einlastungsreihenfolge wird getroffen
- eine Funktion der Einplanungsstrategie wird ausgeführt

Einplanung (engl. *scheduling*) bzw. Umplanung (engl. *rescheduling*):

- nachdem ein Prozess erzeugt worden ist: *begin*
- wenn ein Prozess freiwillig die CPU abgibt: *relinquish*
- falls das von einem Prozess erwartete Ereignis eingetreten ist: *signal*
- sobald ein Prozess wieder aufgenommen werden kann: *continue*

Prozesse können dazu gedrängt werden, die CPU freiwillig abzugeben

- sofern **verdrängende** (engl. *preemptive*) **Prozesseinplanung** erfolgt

Verdrängende Prozesseinplanung

Ereigniseintritt → Einplanung → Einlastung

Verdrängung (engl. *preemption*) des laufenden Prozesses von der CPU bedeutet folgendes:

1. ein Ereignis tritt ein, dessen Behandlungsverlauf zum Planer führt
   - der das Ereignis ggf. erwartende Prozess wird eingeplant
2. der (vom Ereignis unterbrochene) laufende Prozess wird eingeplant
3. der einzulastende Prozess wird ausgewählt & (wieder) aufgenommen
   - ggf. handelt es sich dabei um den unter 1. eingeplanten Prozess

Einplanung und Einlastung von Prozessen erfolgt nicht immer zeitnah zum Ereigniseintritt (d.h., dem Moment der Verdrängungsaufforderung):

- die Verdrängung eines Prozesses verzögert sich ggf. unbestimmt lang
- Ursache dafür ist u.a. die Architektur von Betriebssystem(kern)en

- ggf. entstehende Latenzzeiten können Anwendungen beeinträchtigen

Latenzzeiten und Determinismus

Verdrängung als Querschnittsbelang von Betriebssystemen

Einplanungslatenz (engl. *scheduling latency*) ist unvermeidbar, nicht jedoch ihre Unbestimmtheit — **deterministische Einplanung**:

- zu jedem Zeitpunkt ist der nachfolgende Schritt eindeutig festgelegt
  - unabhängig von Systemlast/-aktivitäten in dem Moment
- die Latenzzeit ist konstant oder mit fester oberer Schranke variabel

Einlastungslatenz (engl. *dispatching latency*), die Zeitspanne zwischen Einplanung und Verdrängung — führt zur weiteren Differenzierung:

verdrängend (engl. *preemptive*) $\sim$ „programmierte Verdrängung“
- Einlastung nur an bestimmten Stellen freigegeben
- **Verdrängungspunkte** (engl. *preemption points*)

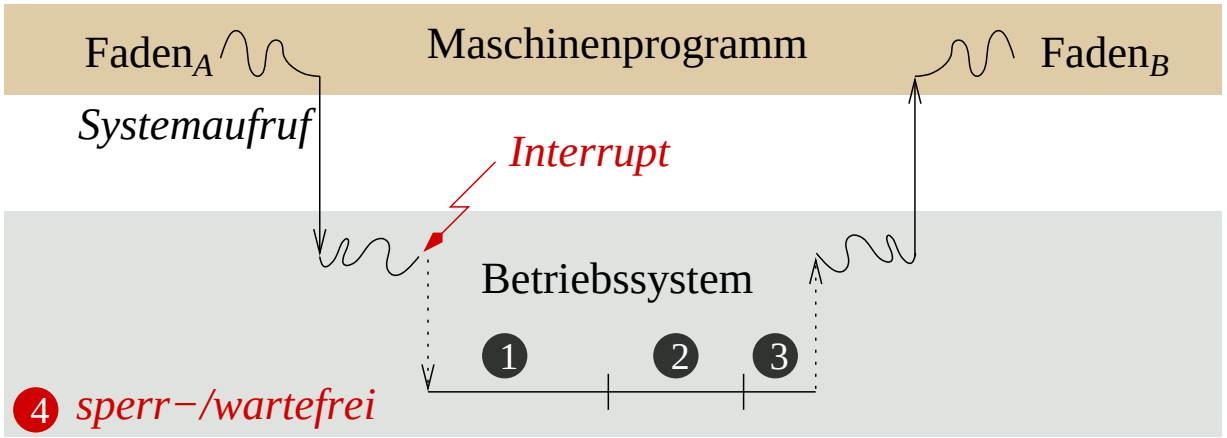
voll verdrängend (engl. *full preemptive*) $\equiv$ Einlastung jederzeit erlaubt

Unterbrechungslatenz (engl. *interrupt latency*), die Zeitspanne zwischen Signalisierung und Behandlungsbeginn einer Programmunterbrechung

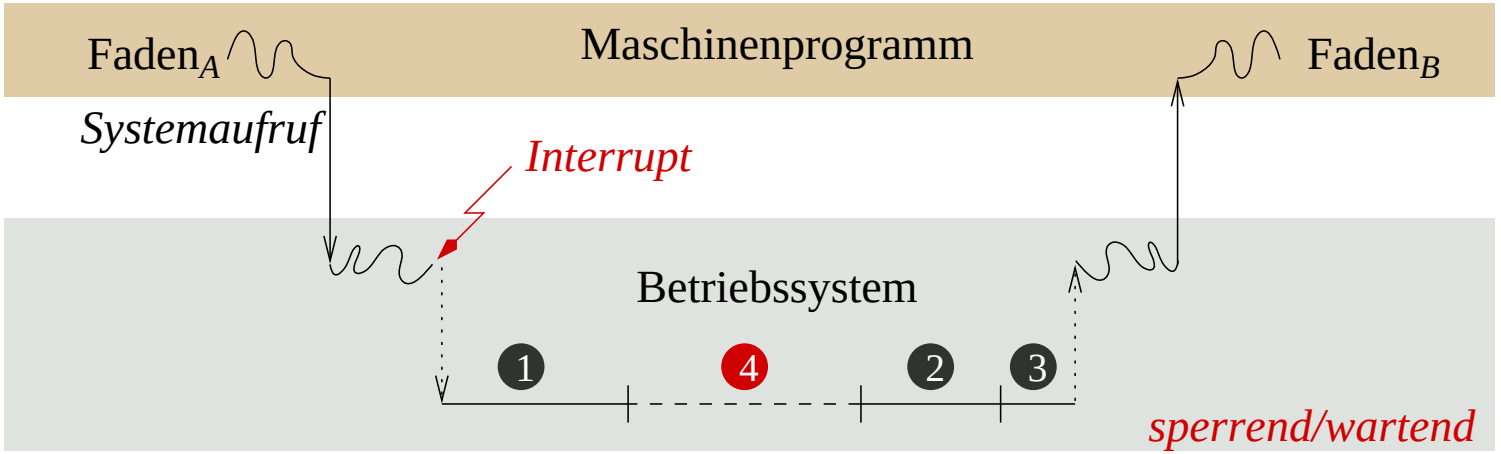
Latenzzeiten in Bezug zum Betriebsmodus

Asynchrone Programmunterbrechungen bleiben eine Quelle der Ungewissheit

voll verdrängend



1. Behandlung
   - Interrupt
2. Einplanung
3. Einlastung
4. Synchronisation



verdrängend

Gliederung

Dimensionen der Prozesseinplanung

Kriterien zur Aufstellung einer Einlastungsreihenfolge von Prozessen

benutzerorientierte Kriterien fokussieren auf **Benutzerdienlichkeit**

- d.h. das vom jeweiligen Benutzer wahrgenommene Systemverhalten
- bestimmen im großen Maße die Akzeptanz des Systems
  - bedeutsam für die Anwendungsdomäne in technischer Hinsicht
  - z.B. Einhaltung und Durchsetzung von Gütemerkmalen

systemorientierte Kriterien haben **Systemperformanz** im Vordergrund

- d.h. die effektive und effiziente Auslastung der Betriebsmittel
- bestimmen im großen Maße die „Rentabilität“ des Systems
  - bedeutsam für die Anwendungsdomäne in kommerzieller Hinsicht
  - z.B. Amortisierung hoher Anschaffungskosten von Großrechnern

Ausschlusskriterien sind dies nicht, vielmehr **Schwerpunktsetzung**:

- gute Systemperformanz ist auch der Benutzerdienlichkeit förderlich

Benutzerorientierte Kriterien

Charakteristische Anforderungserkmale bestimmter Anwendungsdomänen

Antwortzeit Minimierung der Zeitdauer von der Auslösung eines Systemaufrufs bis zur Entgegennahme der Rückantwort, bei gleichzeitiger Maximierung der Anzahl interaktiver Prozesse.

Durchlaufzeit Minimierung der Zeitdauer vom Starten eines Prozesses bis zu seiner Beendigung, d.h., der effektiven Prozesslaufzeit und aller anfallenden Prozesswartezeiten.

Termineinhaltung Starten und/oder Beendigung eines Prozesses (bis) zu einem fest vorgegebenen Zeitpunkt.

Vorhersagbarkeit Deterministische Ausführung des Prozesses unabhängig von der jeweils vorliegenden Systemlast.

Systemorientierte Kriterien

Wünschenswerte Anforderungserkmale vieler Anwendungsdomänen

Durchsatz	Maximierung der Anzahl vollendeter Prozesse pro vorgegebener Zeiteinheit, d.h., der (im System) geleisteten Arbeit.
Prozessorauslastung	Maximierung des Prozentanteils der Zeit, während der die CPU Prozesse ausführt, d.h., „sinnvolle“ Arbeit leistet.
Gerechtigkeit	Gleichbehandlung der auszuführenden Prozesse und Zusicherung, den Prozessen innerhalb gewisser Zeiträume die CPU zuzuteilen.
Dringlichkeiten	Vorzugbehandlung des Prozesses mit der höchsten (statischen/dynamischen) Priorität.
Lastausgleich	Gleichmäßige Betriebsmittelauslastung; ggf. auch Vorzugbehandlung der Prozesse, die stark belastete Betriebsmittel eher selten belegen.

Betriebsart vs. Einplanungskriterien

Prozesseinplanung impliziert eine bestimmte Betriebsart und umgekehrt

allgemein	Gerechtigkeit, Lastausgleich
	• **Durchsetzung der jeweiligen Strategie**
Stapelbetrieb	Durchsatz, Durchlaufzeit, Prozessorauslastung
Dialogbetrieb	Antwortzeit

Echtzeitbetrieb	Dringlichkeit, Termineinhaltung, Vorhersagbarkeit
	• oft im Konflikt mit Gerechtigkeit/Lastausgleich

Proportionalität

Für bestimmte Prozesse ein Laufzeitverhalten „simulieren“, das nicht unbedingt dem technischen Leistungsvermögen des Rechensystems entspricht:

- Nutzer haben oft eine inhärente Vorstellung über die Dauer bestimmter Aktionen
- dieser sollte das System aus Gründen der Benutzerakzeptanz entsprechen

Gliederung

Resümee

- **Einplanungseinheit** für die Prozessorvergabe ist der Faden
  - seine Lauf- und Wartephasen betreiben einen Rechner stoßartig
  - Fäden sind Mittel zum Kaschieren von Totzeiten anderer Fäden
- Betriebssysteme treffen **Zuteilungsentscheidungen** auf drei Ebenen:

long-term scheduling	Lastkontrolle des Systems
medium-term scheduling	Umlagerung von Programmen
short-term scheduling	Einlastungsreihenfolge von Prozessen/Fäden

- die **Entscheidungskriterien** haben verschiedene Dimensionen:

Benutzer	Antwort-/Durchlaufzeit, Termine, Vorhersagbarkeit
System	Durchsatz, Auslastung, Gerechtigkeit, Dringlichkeit, Lastausgleich

- Durchsetzung der Kriterien impliziert eine bestimmte **Betriebsart**
  - umgekehrt: Betriebsarten erwarten die Durchsetzung gewisser Kriterien