Fachhochschule Vorarlberg

Betriebssysteme

Informatik

WS13/14

Prozesse: Scheduling

Armin Simma

Ablaufplanung/-steuerung (scheduling):

- **Ziel**: möglichst effiziente und scheinbar parallele Bearbeitung mehrerer Aufgaben.
- **Voraussetzung:** effiziente Auftragsverwaltung - Verwaltung der Zustände und Übergänge von Prozessen (aufgrund äußerer oder innerer Ereignisse).
- Dazu muss das Betriebssystem Informationen über den aktuellen Status jedes Prozesses haben.
- Info ist abgelegt im Process Control Block (PCB):
  - Der PCB beschreibt den Status aller Betriebsmittel, die vom Prozess verwendet werden.
  - Die Gesamtheit aller PCBs definiert für das Betriebssystem alle in Arbeit befindlichen Aktivitäten.

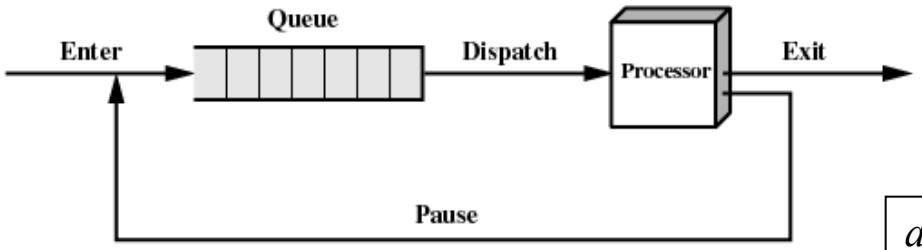
Prozessverwaltung
Dispatcher Programm

Prozesse werden in **Queues** (Warteschlange) verwaltet:

- (Ready) Queue: bereite Prozesse
- Elemente in der Queue
  - PCB von Prozess oder
  - Referenz auf PCB

Anmeldung eines neuen Prozesses:

- Prozess erhält eindeutigen Identifier
- Prozess wird Speicherplatz zugeordnet (enthält Process Image)
- Betriebssystem legt neuen PCB an
- Neuer Prozess wird in Ready-Queue eingereiht



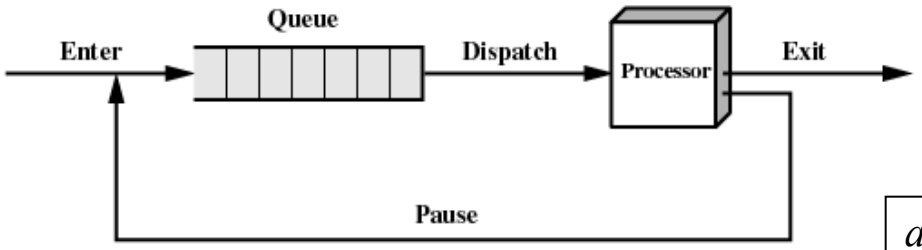
(b) Queuing diagram

aus: Stallings. Betriebssysteme

Prozessverwaltung
Dispatcher Programm

Prozesswechsel (Process switch) durch Dispatcher:

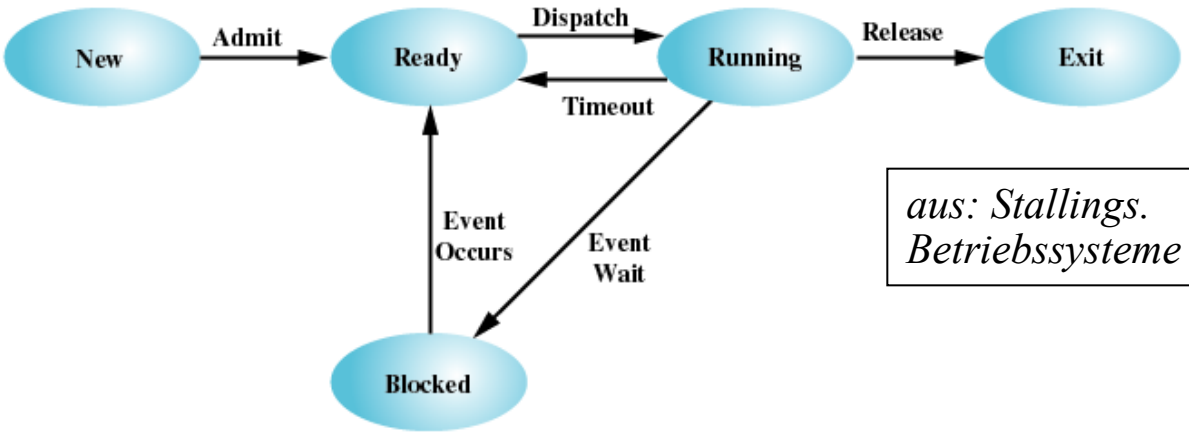
- Laufenden Prozess anhalten
- Dessen Zustand im PCB sichern
- eventuell Daten(Heap)- und Stack-Inhalt sichern
- Scheduler aufrufen
- Zustand des nächsten (bereiten) Prozesses wiederherstellen
- Prozessor an nächsten Prozess übergeben



(b) Queuing diagram

aus: Stallings. Betriebssysteme

Zustände eines Prozesses
Modell 2 (5 states)

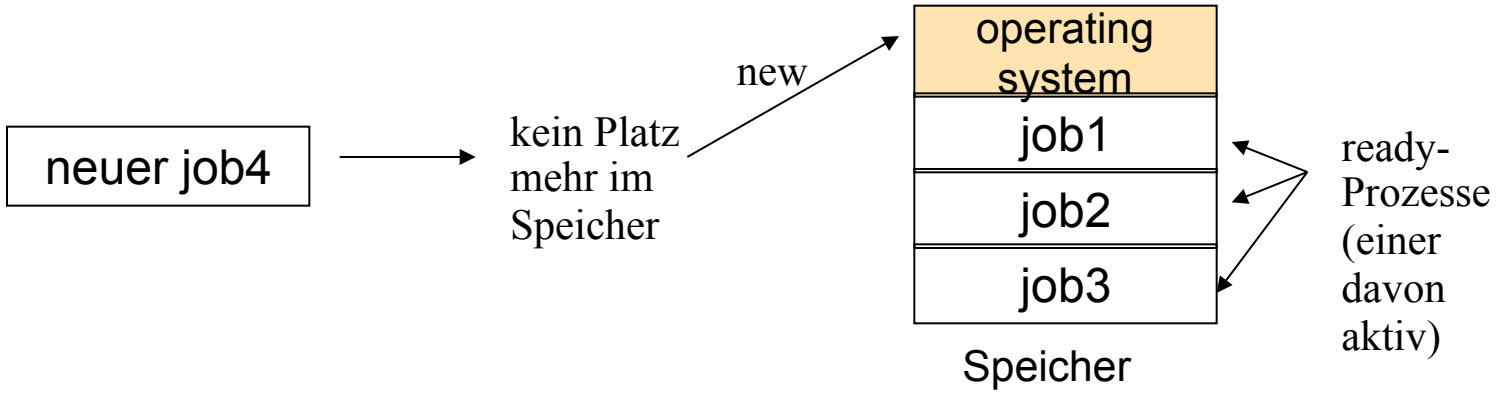


aus: Stallings. Betriebssysteme

- *New* Zustand:
  - Prozess ist noch nicht im System zugelassen
  - Weil zum Beispiel Systemlast zu hoch (z.B. zu wenig (Haupt)Speicher für alle Prozesse)
- *Exit* Zustand:
  - Prozessinformationen werden vielleicht noch benötigt
  - Abrechnung/ Buchhaltung...
  - Zombie in Linux

Zustände eines Prozesses
Zustandsübergang New ➔ Ready

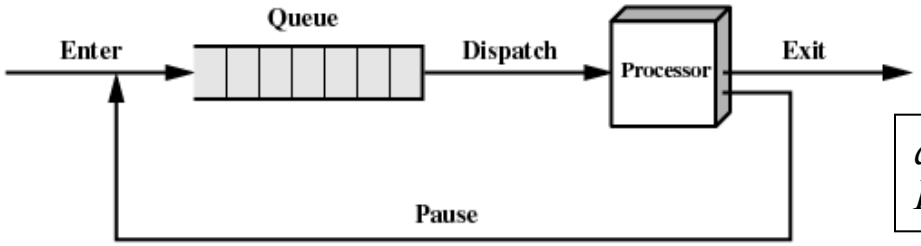
- New ➔ Ready (Admit)
- Prozess wird für Rechnen auf CPU zugelassen:
  - PCB anlegen
  - Eventuell Code und Daten von Platte in Hauptspeicher kopieren
- Entscheidung, wann dies passiert: als *long-term scheduling* bezeichnet
- long-term schedulng macht vor allem für batch Jobs oder Multi-User System Sinn:
- Wenn alle ready-Prozesse komplett in Hauptspeicher untergebracht und dieser HS knapp wird, dürfen keine neuen Prozesse mehr zugelassen werden.



Scheduling-Algorithmus
simples Modell1

- Was wir brauchen:
  - Zeitscheibe
  - Queue (Warteschlange)

Scheduling-Algorithmus
simples Modell1 (1 queue)

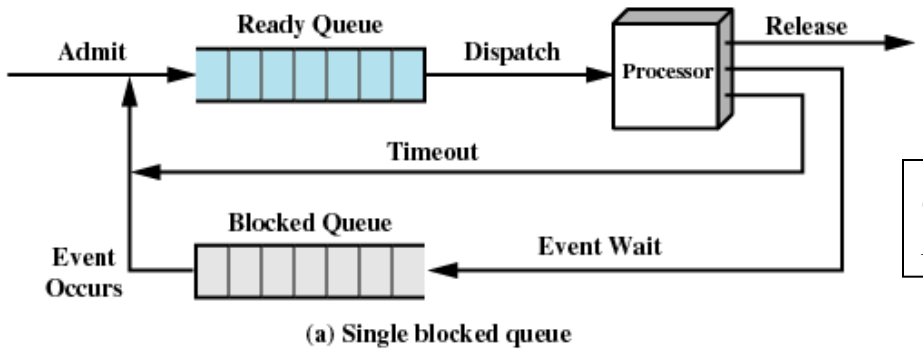


(b) Queuing diagram

aus: Stallings. Betriebssysteme

- Wenn Zeitscheibe zu Ende ist kann System nicht rechtesten Prozess aus Queue aktivieren, denn …
  - dieser ist vielleicht blockiert (wegen E/A)
  - System müsste ganze Queue nach erstem ready Prozess durchsuchen.
    - Zeitaufwändiges Suchen

Scheduling-Algorithmus
Modell2 (2 queues)

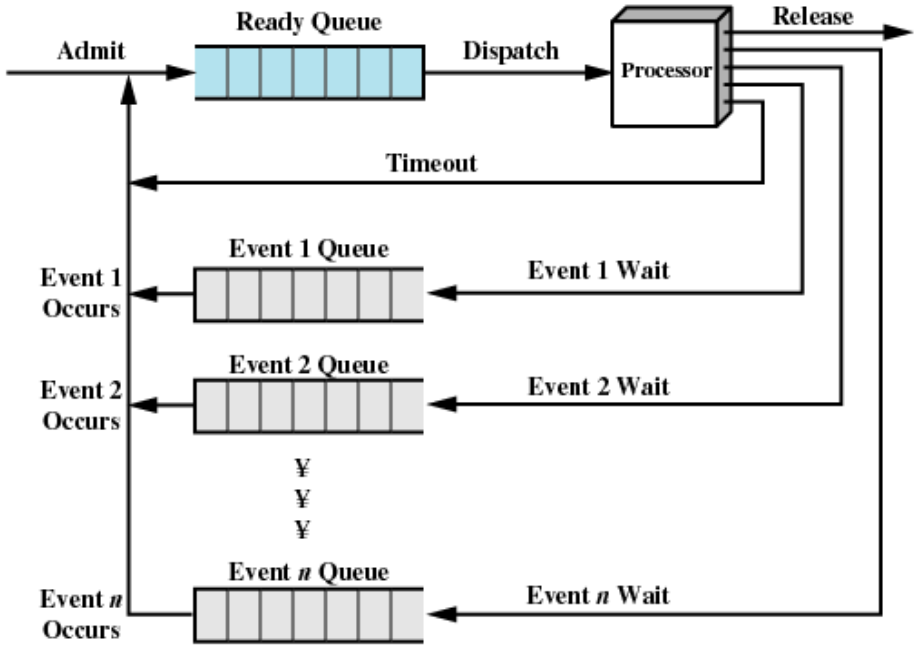


(a) Single blocked queue

aus: Stallings. Betriebssysteme

- 2 queues: **ready** und **blocked** queue
- Wenn Event (Interrupt) von E/A kommt
  - Erster Prozess, der auf **dieses** Event wartet, wird aktiv
    - Zeitaufwändiges Suchen nach Prozess, der auf diesen Interrupt (=Event) wartet

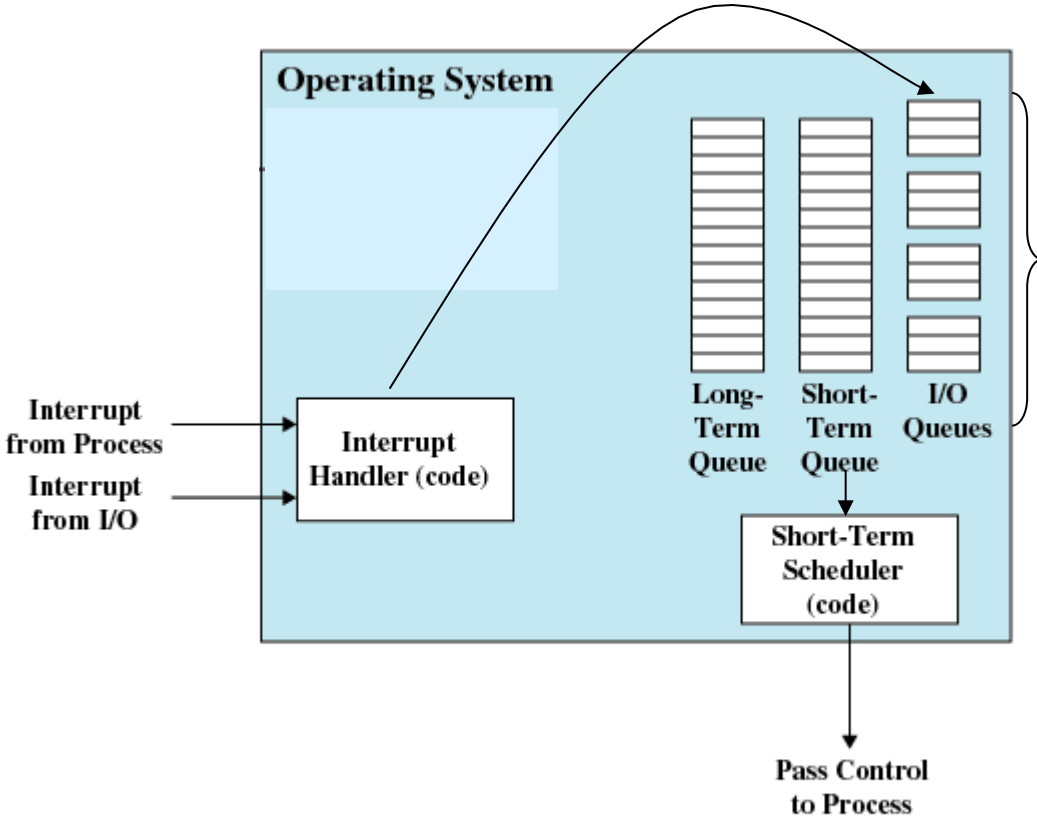
Scheduling-Algorithmus
Modell3 (mehrere queues)



aus: Stallings. Betriebssysteme

- Pro Event(-klasse) eine Queue
  - Harddisk-Controller Interrupt
  - Tastatur-Event…
- Wenn von HW Interrupt (INT) kommt
  - ➔ INT geht ans Betriebssys.
  - ➔ Betriebssys. leitet INT an richtigen Prozess in richtigen Queue weiter

Scheduling-Algorithmus
Modell3 (mehrere queues):Gesamtsicht



- Queues:
- long-term: für new-Prozesse
- Short-term = ready queue
- I/0 = event queue

aus: Stallings. Betriebssysteme

- "komplettes" Betriebssystem (Kernel)
  - Scheduler
    - inklusive Queues
  - +Interrupt-Handler

- Weiter mit bs08_scheduling_Teil2
- Folgeslides nur wenn noch Zeit übrig

Scheduling-Algorithmus
Problem mit Modell 3

- Nicht alle Prozesse passen gleichzeitig in Hauptspeicher
- **Swapping**: Prozesse können ein-/ und ausgelagert werden (vom/ in) Hauptspeicher (auf/von) Platte
- Wenn alle eingelagerten Prozesse auf E/A warten und Hauptspeicher voll
  - Lagere Prozess(e) aus
  - Um neue(n) Prozess(e) zuzulassen (vom New state in Ready durch Admit-Übergang)
- Blockiert-Zustand wird zu **suspend**-state wenn auf Platte ausgelagert
- Neuer Zustand notwendig: **suspend (ausgelagert,** suspendiert)
- **mid-term scheduling**

Modell 4: neuer Zustand Suspend

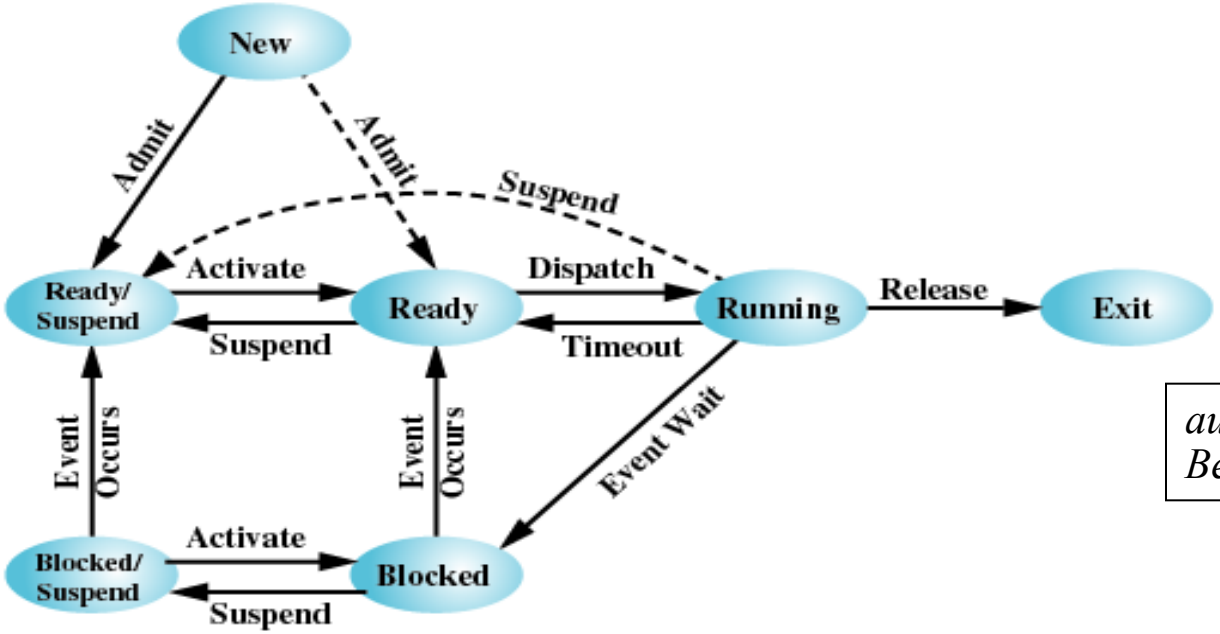
(a) With One Suspend State

aus: Stallings. Betriebssysteme

- Suspend bedeutet also: Ausgelagert (auf Platte) und Blockiert
- Activate bedeutet: E/A Event (INT) für diesen Prozess erhalten

Modell 5: 2 Suspend States

- Auslagerung könnte auch **ohne** Blockierung wegen E/A-Opertion erfolgen, wenn
  - zuwenig Main Memory jedoch
  - Prozess mit höherer **Priorität** kommt im System an



aus: Stallings. Betriebssysteme

- Suspend unterteilen:
  - Ready/ Suspend: Ausgelagert (ohne auf E/A zu warten)
  - Blocked/ Suspend: Ausgelagert und Blockiert

Prozess-zustände unter UNIX

Kommt nicht zur Klausur!

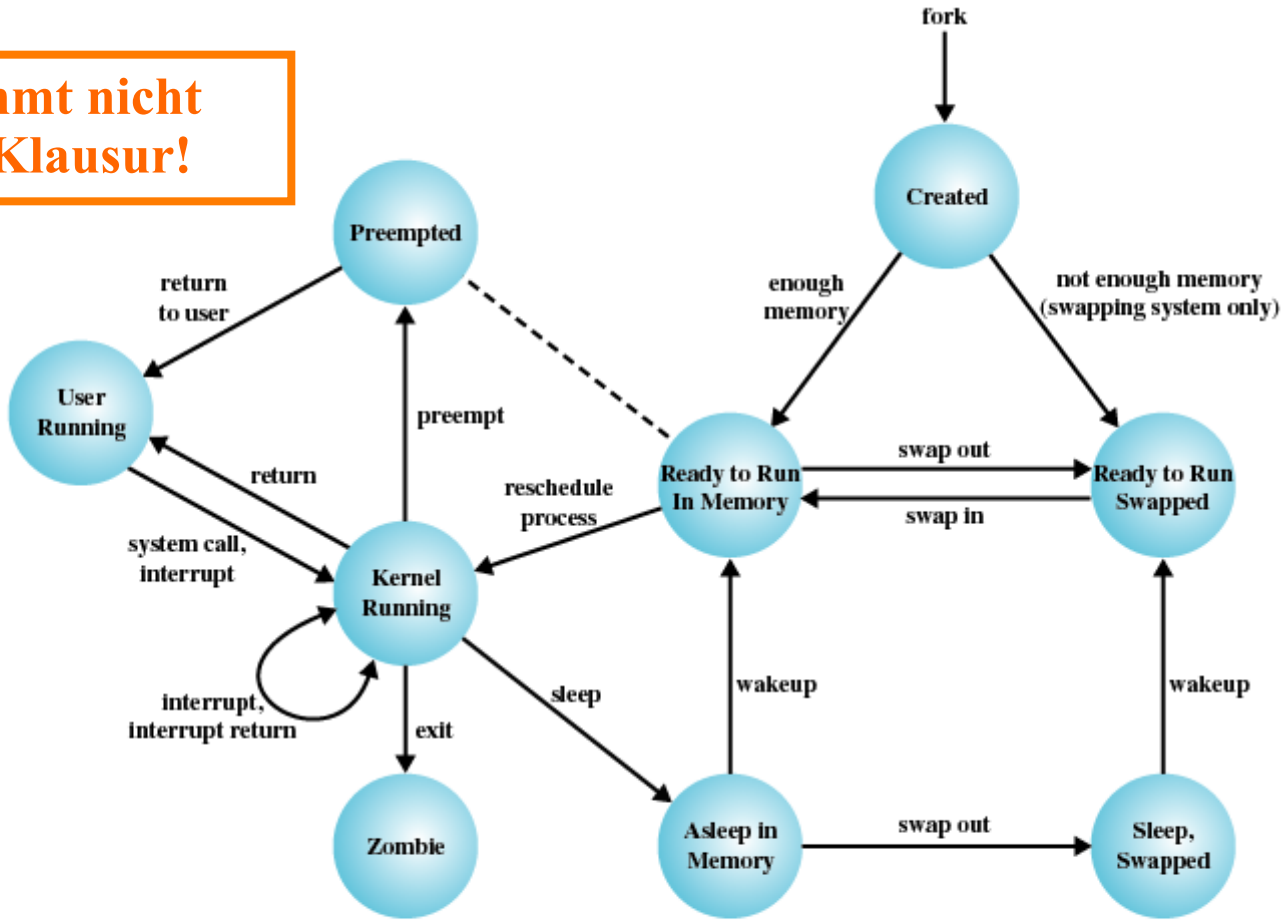
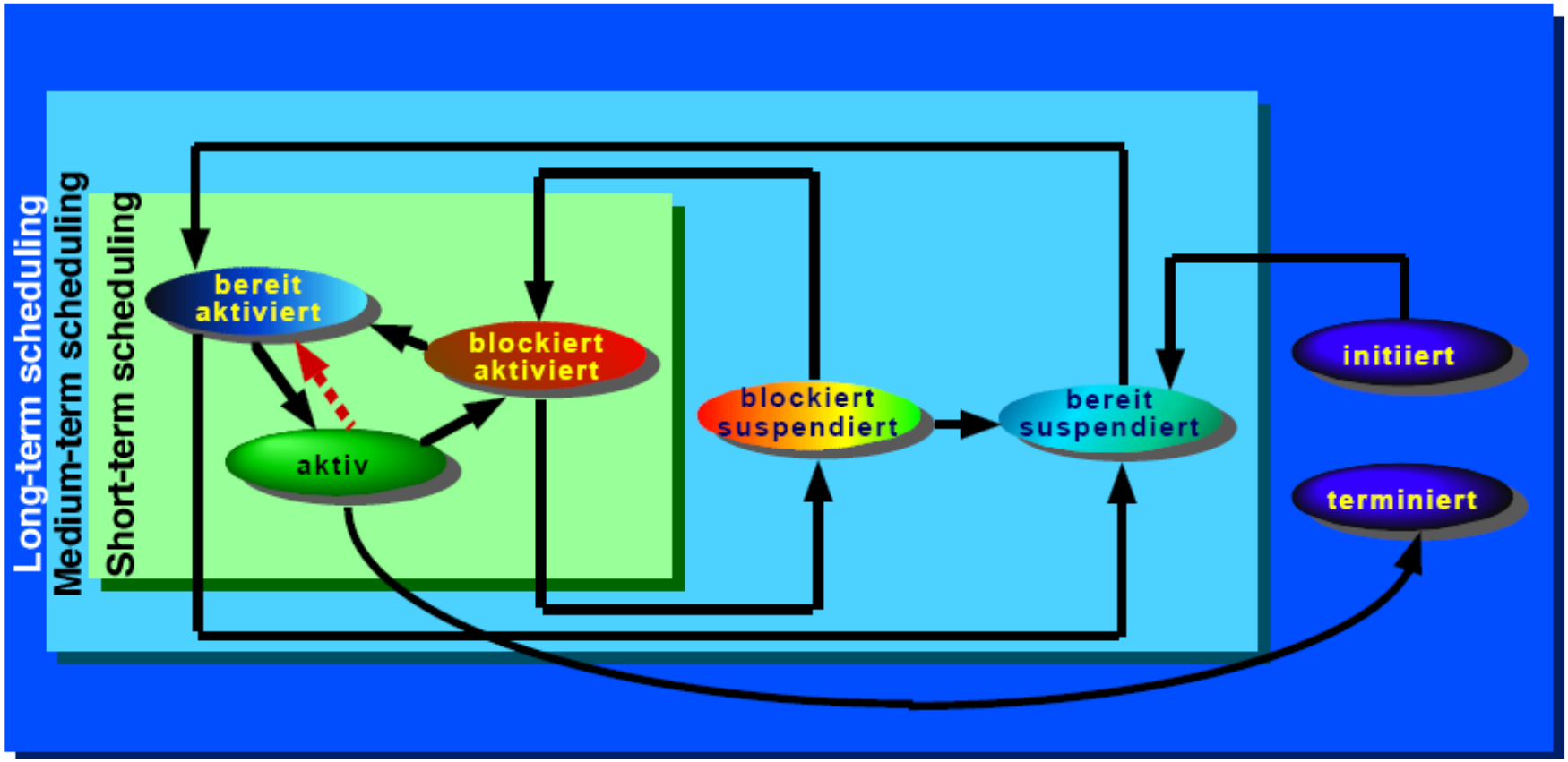


Figure 3.17 UNIX Process State Transition Diagram

aus: Stallings. Betriebssysteme

Vergleich Long-/Medium-/Short-term Scheduling



Vgl. “vorvorige” Folie

- initiiert = new
- terminiert = exit
- bereit aktiviert = ready
- blockiert aktiviert = blocked

Vergleich Long-/Medium-/Short-term Scheduling

- Wir betrachten in Folge nur **short-term** scheduler
- auch **dispatcher** genannt