Analysis of System Performance

IN2072

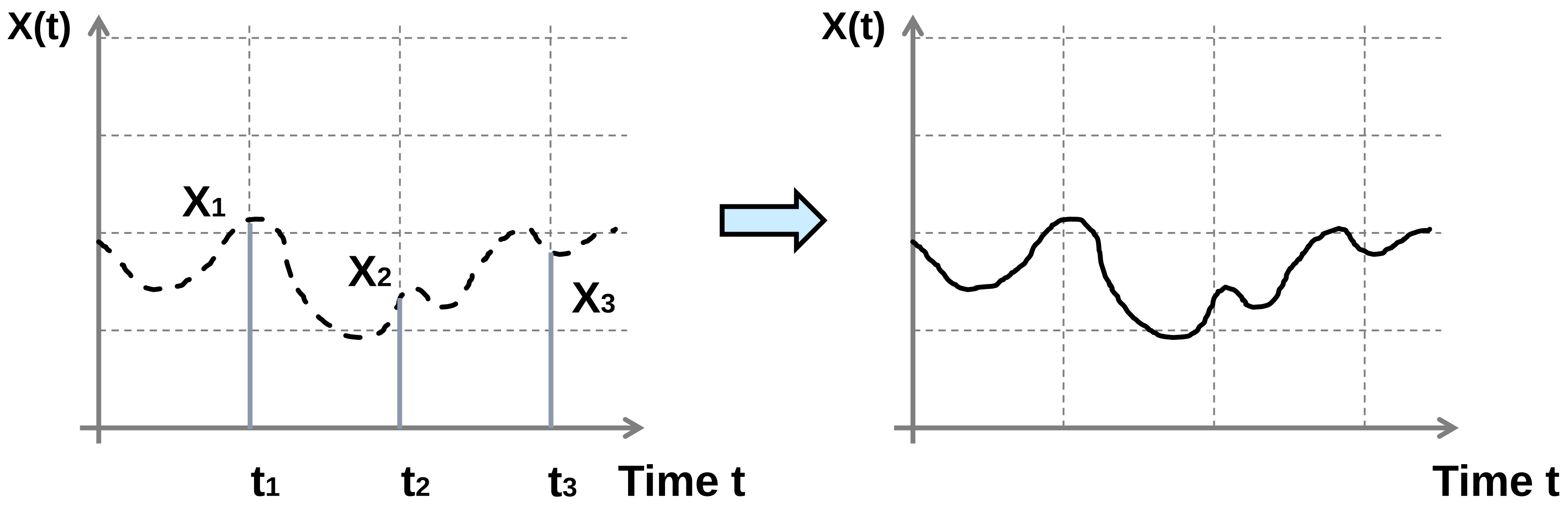
Chapter 2 – Random Process

Part 1

Dr. Alexander Klein
Prof. Dr.-Ing. Georg Carle

Chair for Network Architectures and Services
Department of Computer Science
Technische Universität München
http://www.net.in.tum.de

Stochastic Process

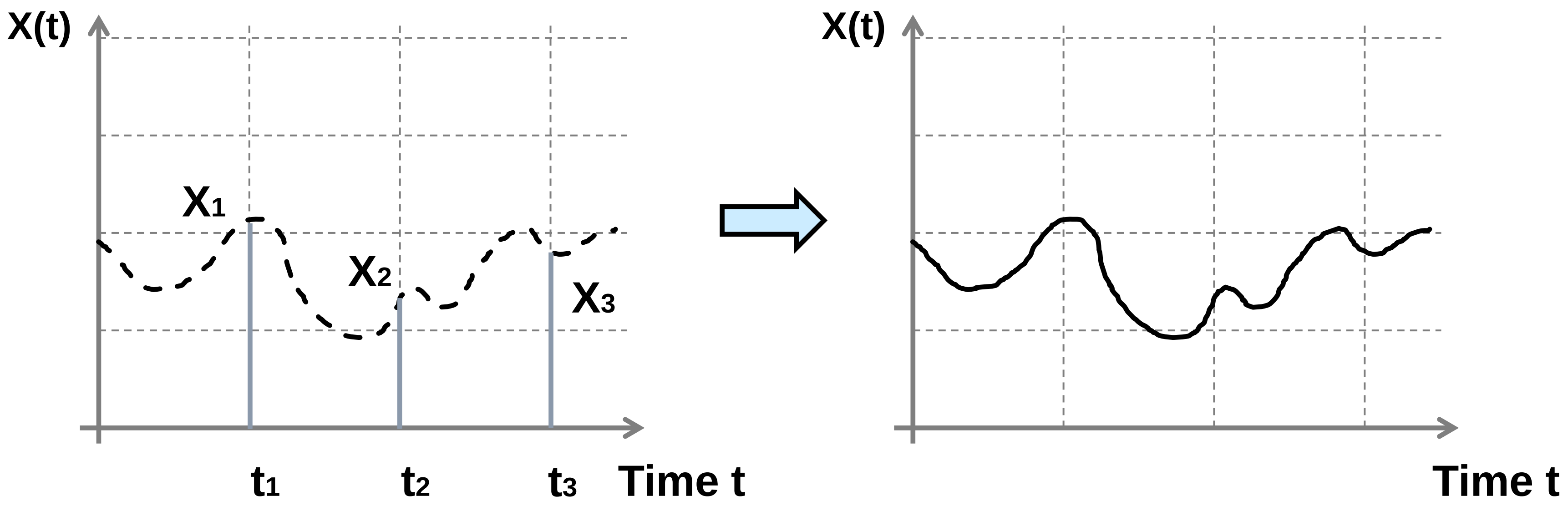


- **Definition (1/2):**

  A stochastic process is defined as a family of random variables $\{X_t : t \in T\}$ where each random variable X_t is indexed by parameter $t \in T$, which is usually called the time parameter if $T \subseteq R_+ = [0,\infty)$. The set of all possible values of X_t (for each $t \in T$) is known as the state space S of the stochastic process.

„Queueing Networks and Markov Chains – Modeling and Performance Evaluation with Computer Science Applications“, G.Bolch, S.Greiner, H. deMeer, K.S. Trivedi

Stochastic Process



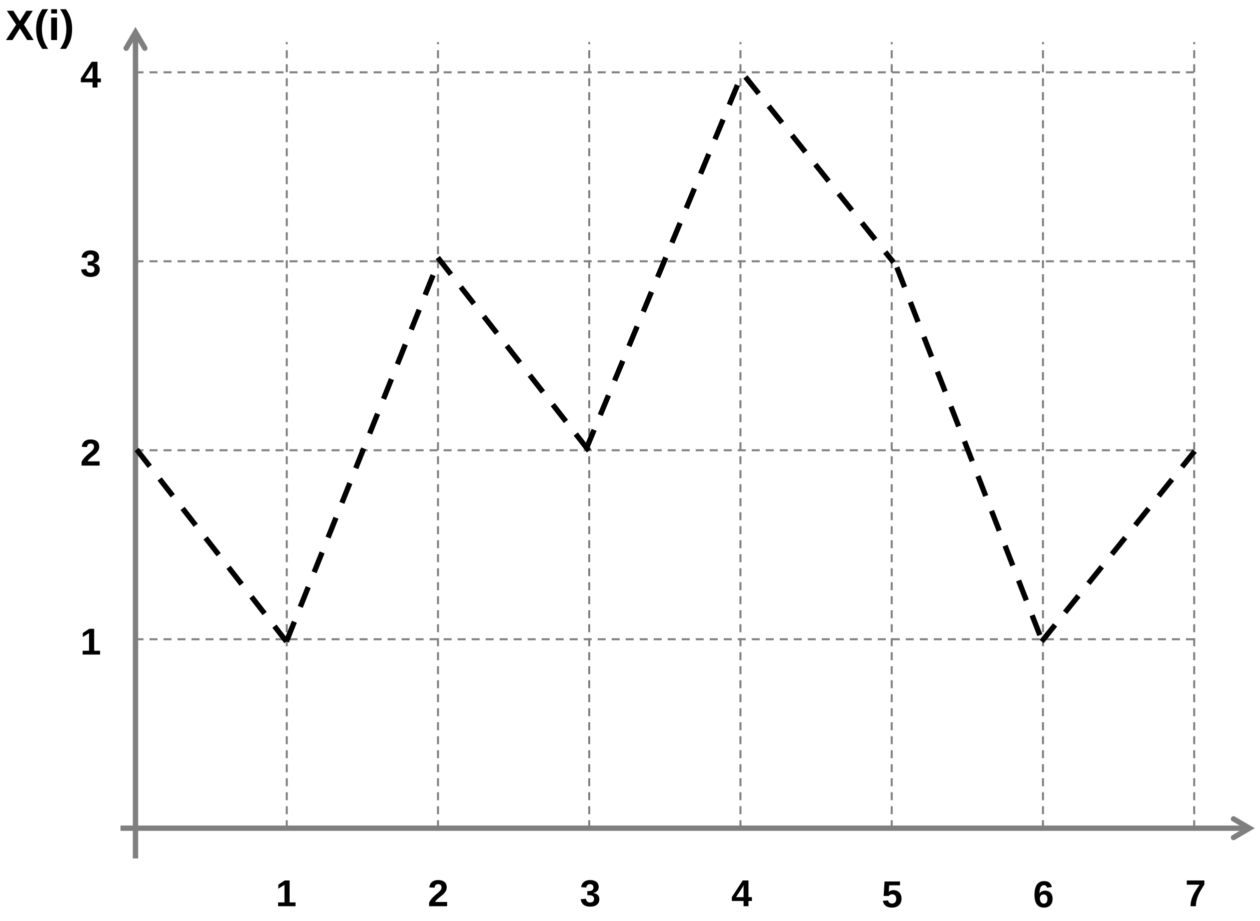
- **Definition (2/2):**

  If countable, discrete-parameter set T is encountered, the stochastic process is called a discrete-parameter process and is commonly represented by (a subset of) $\mathrm{N}_0 = \{0,1,\ldots\}$; otherwise we call it a continuous-parameter process. The state space of the stochastic process may also be continuous or discrete.

„Queueing Networks and Markov Chains – Modeling and Performance Evaluation with Computer Science Applications", G.Bolch, S.Greiner, H. deMeer, K.S. Trivedi

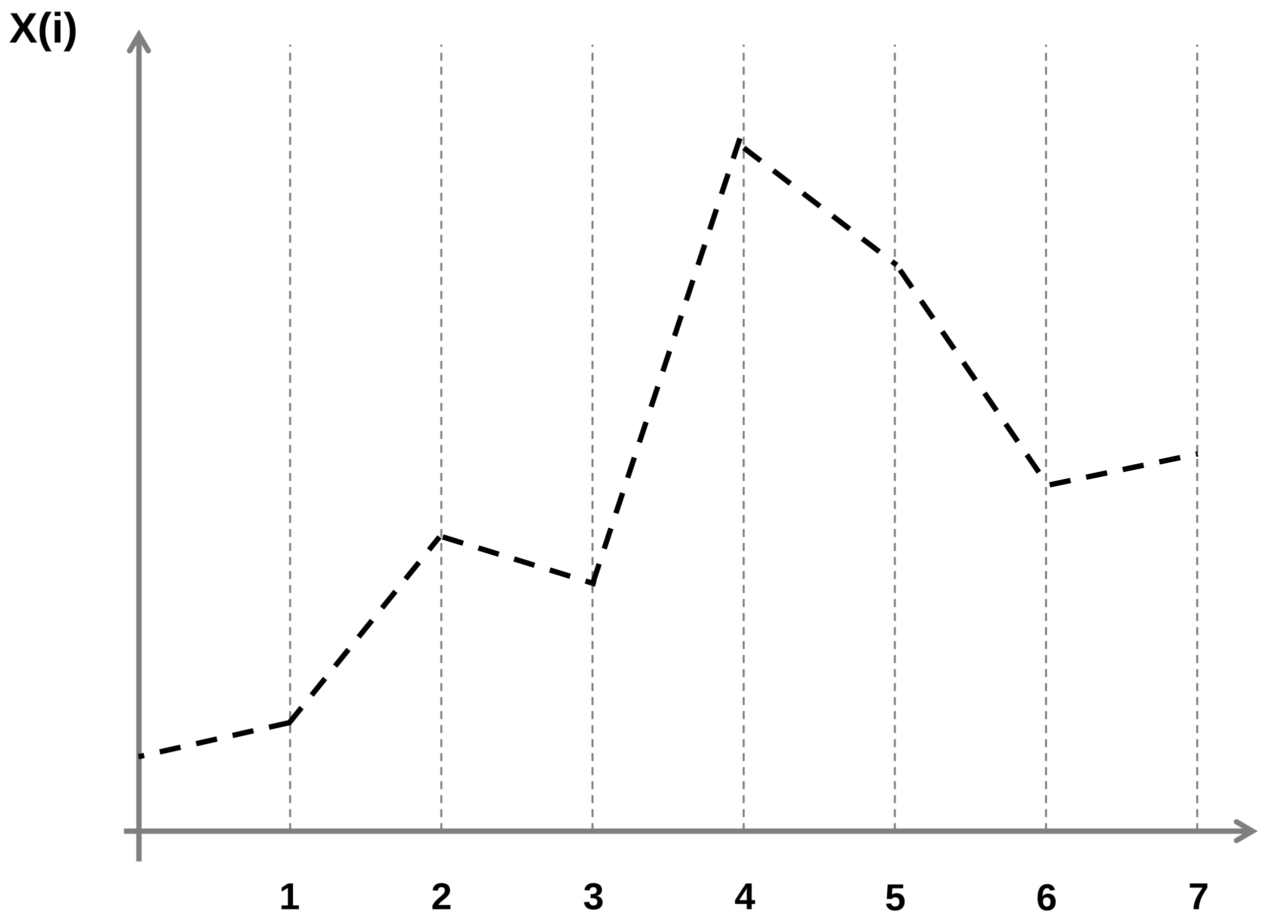
Stochastic Process

- Discrete parameter, time discrete



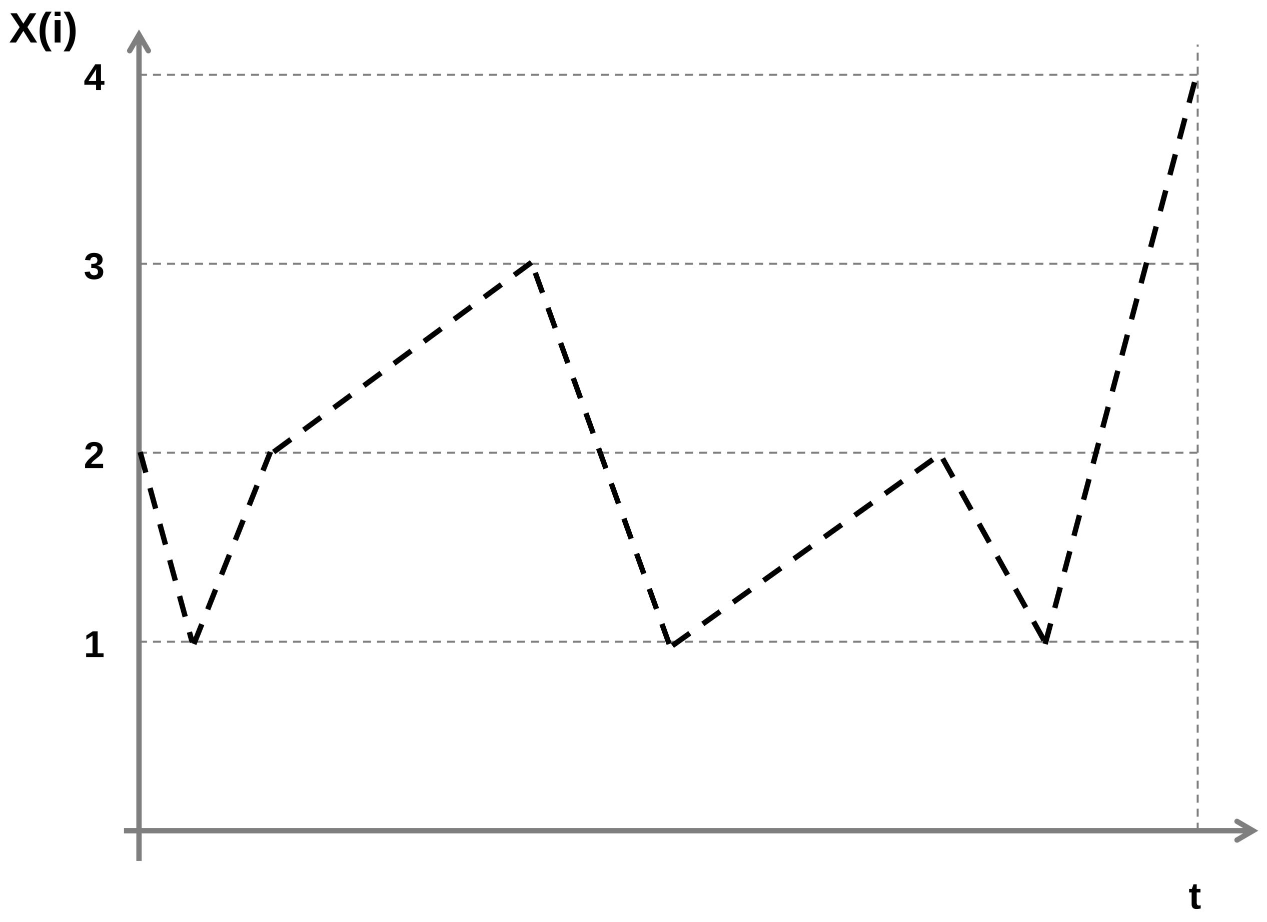
Stochastic Process

- Continuous parameter, time discrete



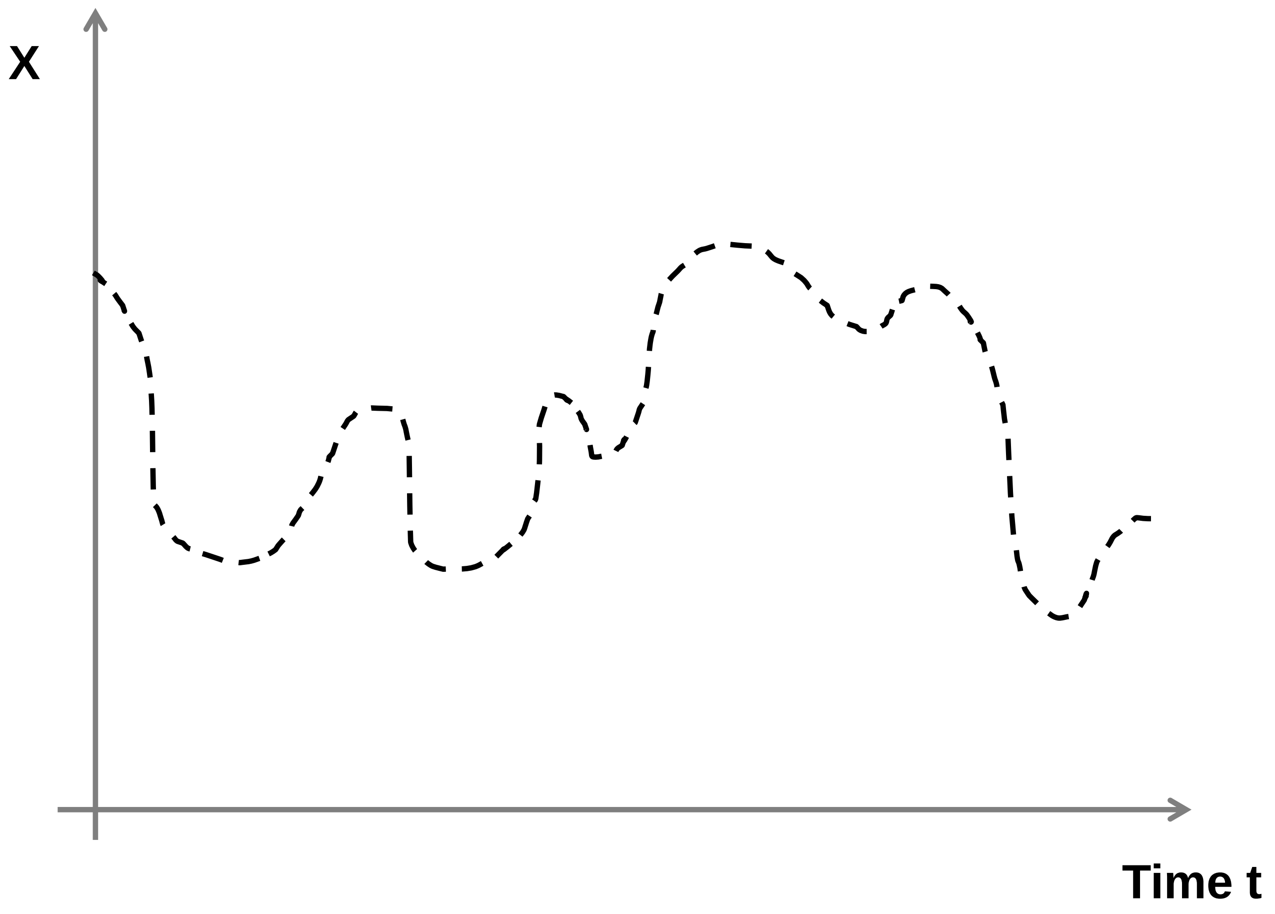
Stochastic Process

- Discrete parameter, time continuous

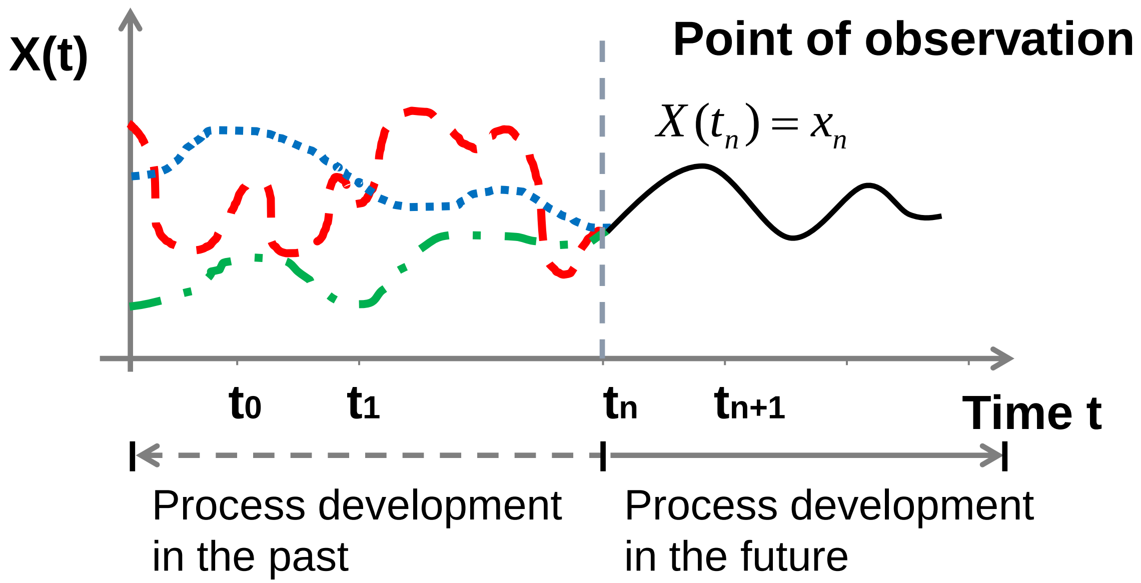


Stochastic Process

- Continuous parameter, time continuous



Markov Process


X(t)
Point of observation
$X(t_n) = x_n$
t0
t1
tn
tn+1
Time t
Process development in the past
Process development in the future

- Definition:

  A stochastic process is called Markov Process if its development in the future only depends on the current state of the process. The markovian characteristic can be described by the following expression if its current state is x_n and the observation time is t_n .

$$P\{X(t_{n+1}) = x_{n+1} | X(t_n) = x_n, \ldots, X(t_0) = x_0\}$$
$$= P\{X(t_{n+1}) = x_{n+1} | X(t_n) = x_n\}, t_0 < t_1 < \ldots < t_n < t_{n+1}.$$

Arrival Process

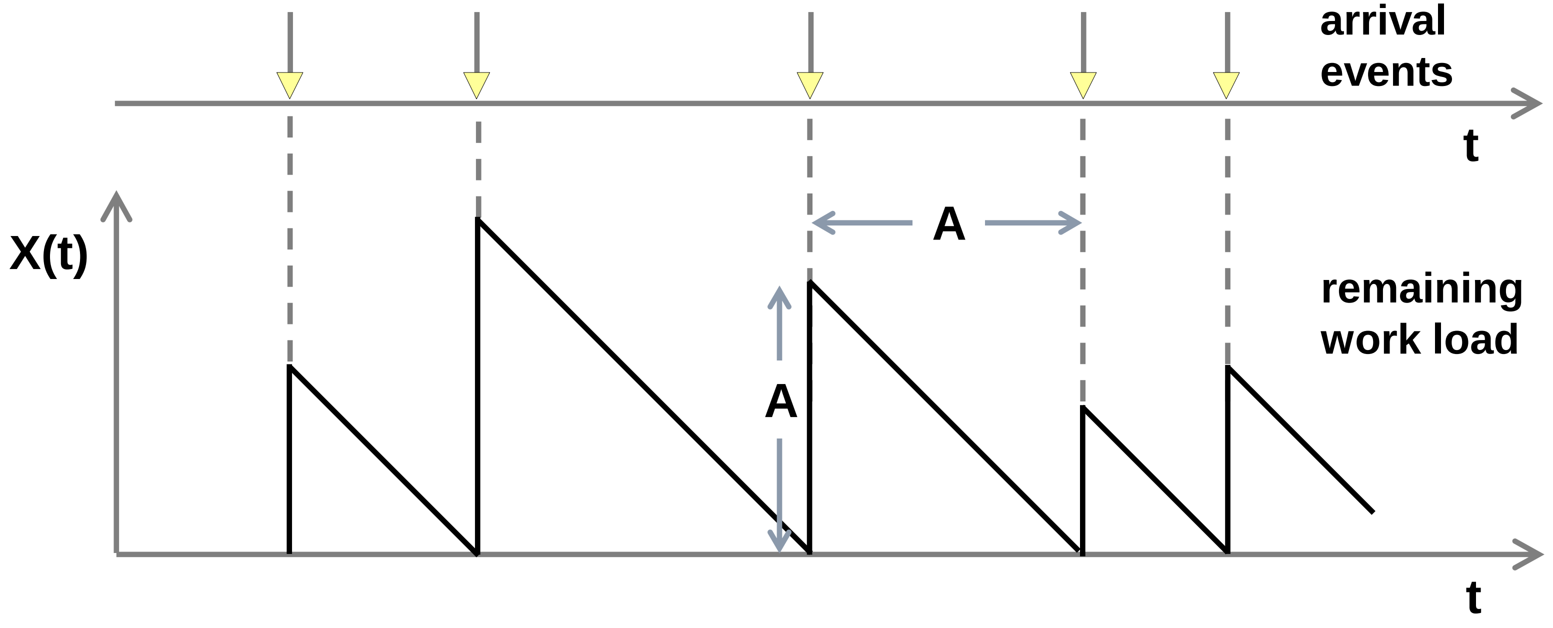
- Definition:

  An arrival process is a stochastic process which describes the chronological order of arrival events.

  - Batcharrival:

    A batch arrival represents the multiple arrivals at the same point in time.

  Arrival process as state process

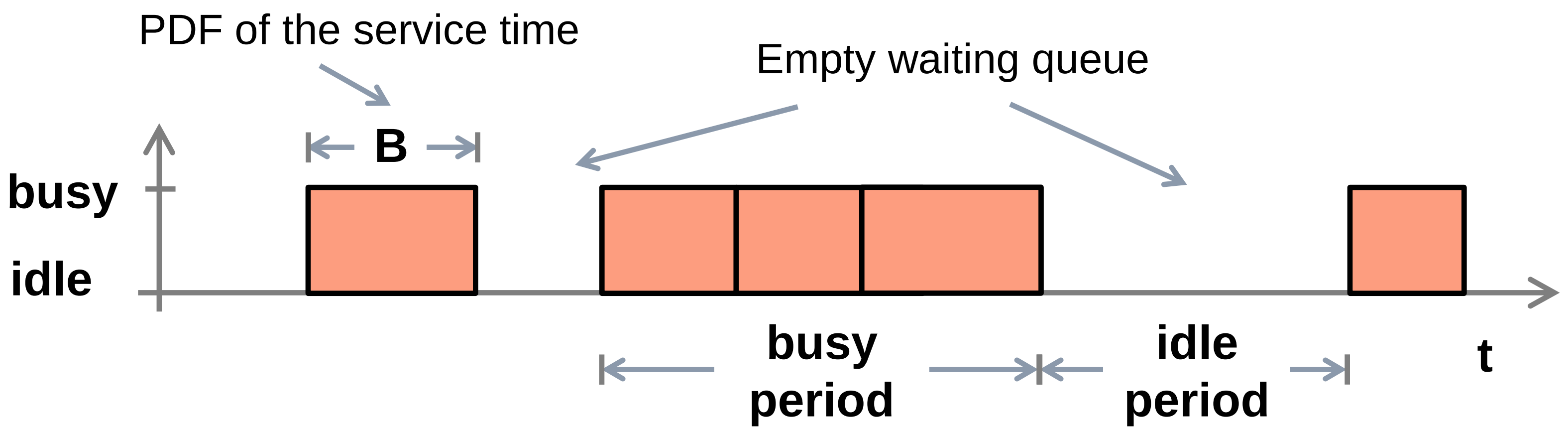


Service Process

- Definition:

  A service process describes the behavior of one or multiple service units and specifies the chronological order of the service routine.

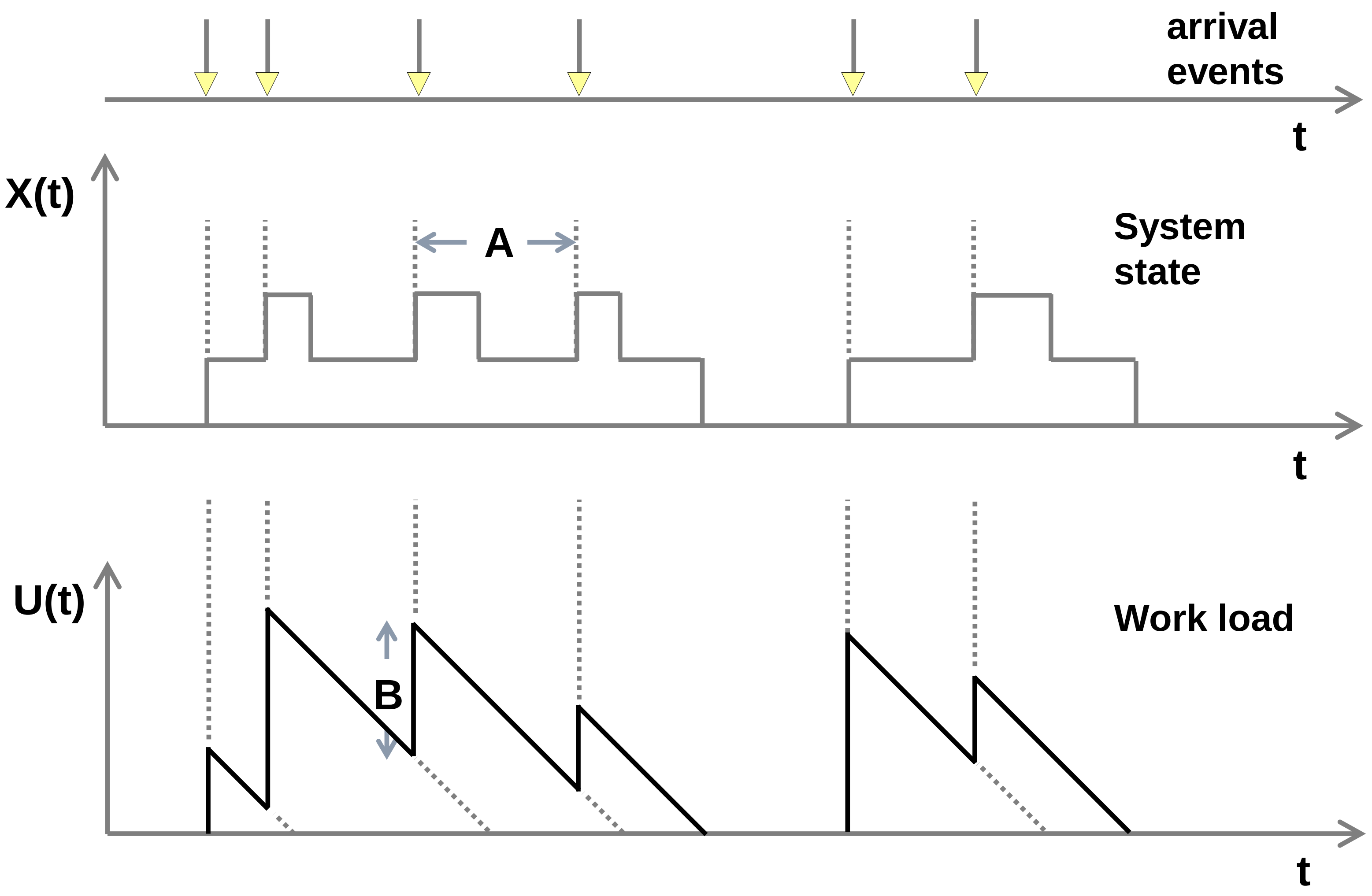
  Service process of a single service unit:



State Process

- Visualization:

Arrival events

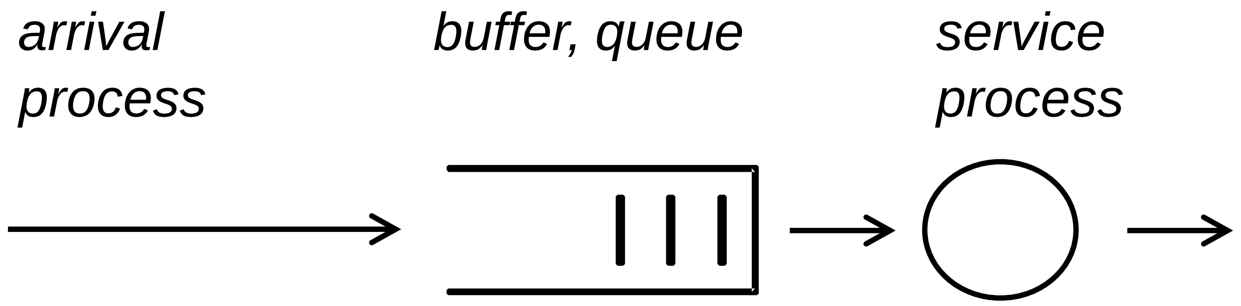


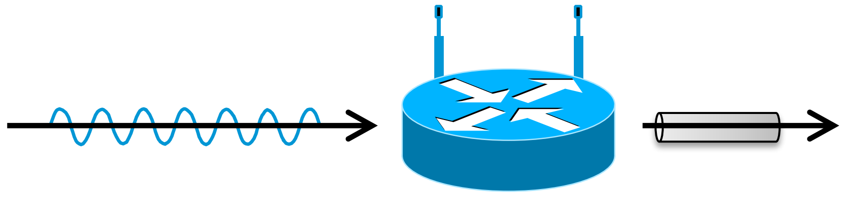
Motivation

- Evaluate Properties of Computer Systems:
  - Correctness (verification)
  - Performance
- Modern Computer Systems:
  - Guarantee minimum performance (e.g. delay for real time systems)
  - Comparison of different systems
- Evaluation:
  - Measurement
  - Modeling
  - Performance Evaluation
- Methods to improve the performance:
  - Design
  - Development
  - Tuning
  - Comparison of different systems

Queuing Network Model

Waiting Queue Theory



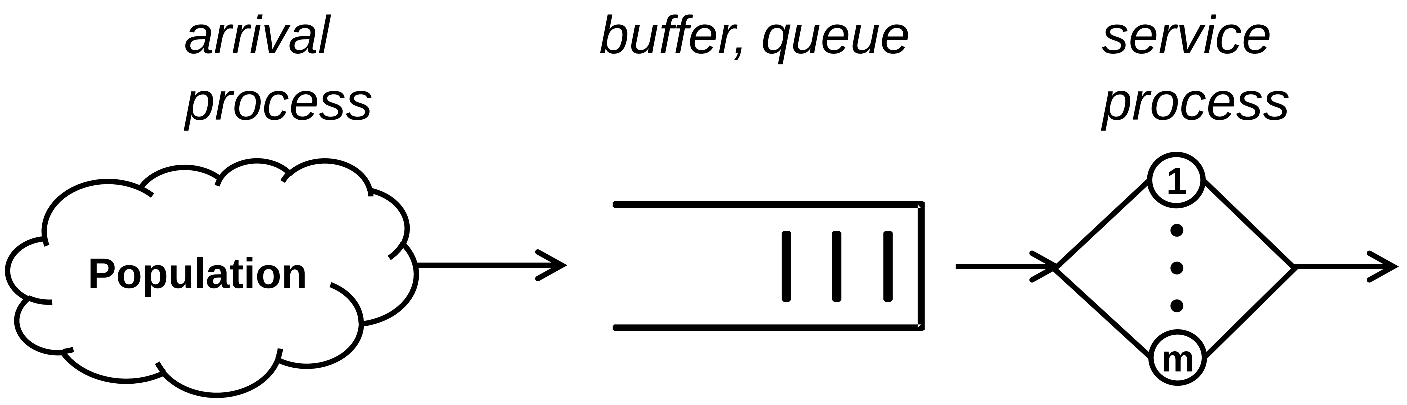


- Example:

  Router
  - Data packets arrive at the router via its wireless interface
  - A packet is forwarded immediately via the DSL interface if the buffer is empty and no packet is currently transmitted
  - Otherwise the packet is stored in the buffer if the buffer is below its maximum capacity
  - The service process simulates the time that is required by the router to write a packet on the DSL interface

Queuing Systems

- System Characteristics:



- Finite / Infinite
- Entity size
- Deadline

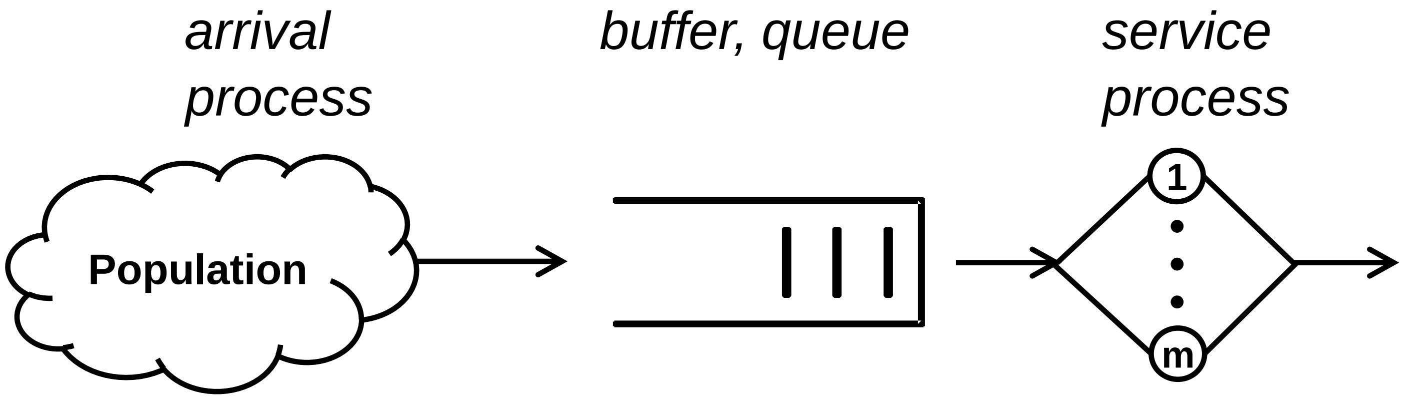
- Single queue / Multiple queues
- Finite / Infinite capacity
- Queuing strategy (FIFO, LIFO…)

- Single processor / multiple processor
- State-(in)dependent
- Vacation

- Typical Performance Characteristics:
  - Average/maximum customer waiting time
  - Average/maximum processing time of a customer
  - Average/maximum retention time of a customer
  - Average/maximum number of customers in the queue
  - Customer blocking probability
  - Utilization of the system / individual processing units

Queuing Systems – Kendall’s Notation

- System Characteristics:



- Parameter:
  - Arrival rate: $\lambda = \frac{1}{T_A}$ $T_A =$ mean interarrival time
  - Service rate: $\mu = \frac{1}{T_B}$ $T_B =$ mean service time
  - Number of servers: m

Queuing Systems – Kendall's Notation

- **Kendall's Notation**

$A^{[x]} / B / m - S$ queueing discipline

Number of Places in the Queue

S = 0 Loss/Blocking System

S = ∞ Waiting System

Number of Servers

Service Time Distribution

Batch Arrival Process

Arrival Process

Queuing Systems – Kendall's Notation

Queuing Discipline

FIFO / FCFS	First In First Out / First Come First Served
LIFO / LCFS	Last In First Out / Last Come First Served
SIRO	Service In Random Order
RR	Round Robin
PNPN	Priority-based Service
PS	Processor Sharing
EDF	Earliest Deadline First
LLF	Least Laxity First
Preemption	Jobs can be interrupted

Distributions

M	Markovian	Exponential Service Time
D	Degenerate Distribution	A deterministic service time
E_k	Erlang Distribution	Erlang k distribution
GI	General distribution	General independent
H_k	Hyper exponential	Hyper k distribution

Queuing Systems – Kendall's Notation

- Example:

M / GI / 2 – 5 EDF Preemptive Resume (PR)

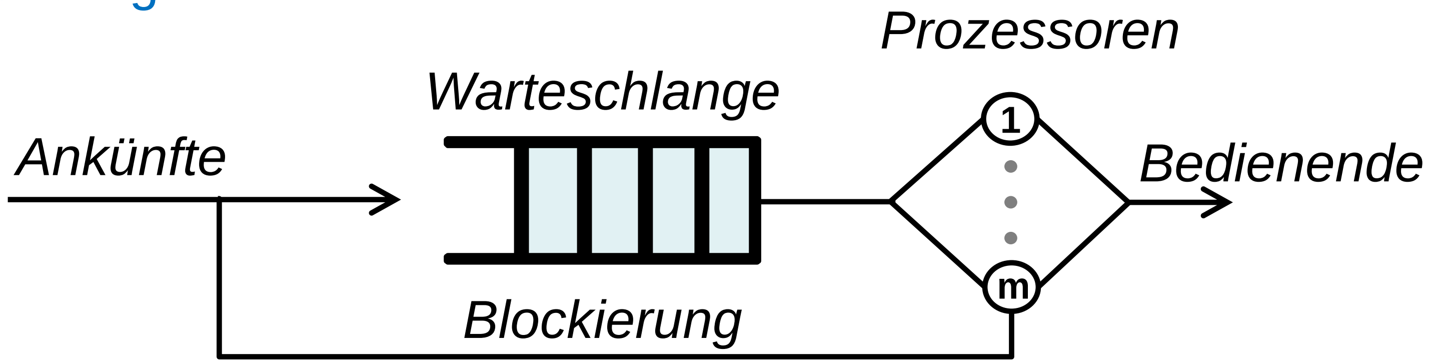
  - Interarrival time is exponentially distributed
  - Service time is arbitrarily distributed
  - Number of servers m=2
  - Number of waiting slots S=5
  - Queueing discipline: EDF (Earliest Deadline First)
  - Scheduling:
    - Preemptive means that an arriving job may preempt a job which is currently processed by the server
    - An interrupted job is resumed from the point when it was interrupted

Kriterien für das Scheduling

- Scheduling-Kriterien:
  - Hohe Prozessorauslastung $\Longleftrightarrow$ kurze Antwortzeit
- Schedulingziele:
  - Stapelverarbeitung:
    - Hoher Durchsatz, gute Auslastung
    - Priorisiere Prozesse die freie Ressourcen nutzen
  - Interaktiver Betrieb:
    - Kurze Antwortzeiten
    - Priorisiere Prozesse die auf E/A gewartet haben und rechenbereit sind
  - Echtzeitbetrieb:
    - Einhaltung von Zeitvorgaben
    - Priorisiere Prozesse deren Zeitschranken ablaufen

Schedulingverfahren

- Scheduling:

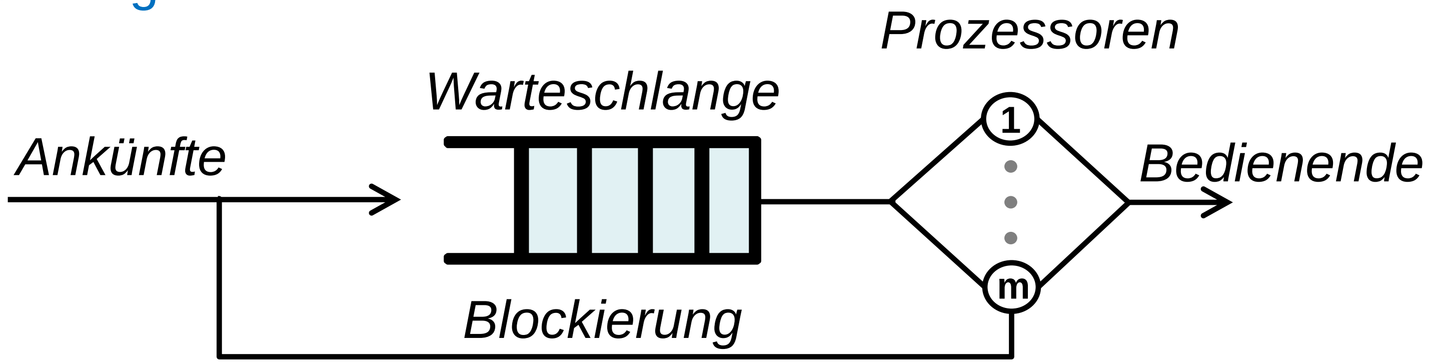


- Scheduling Strategien

FIFO / FCFS	First In First Out / First Come First Served
RR	Round Robin
EDF	Earliest Deadline First
SJF	Shortest Job First
LLF	Least Laxity First
⋮	⋮

Schedulingverfahren – First In First Out

- Scheduling:

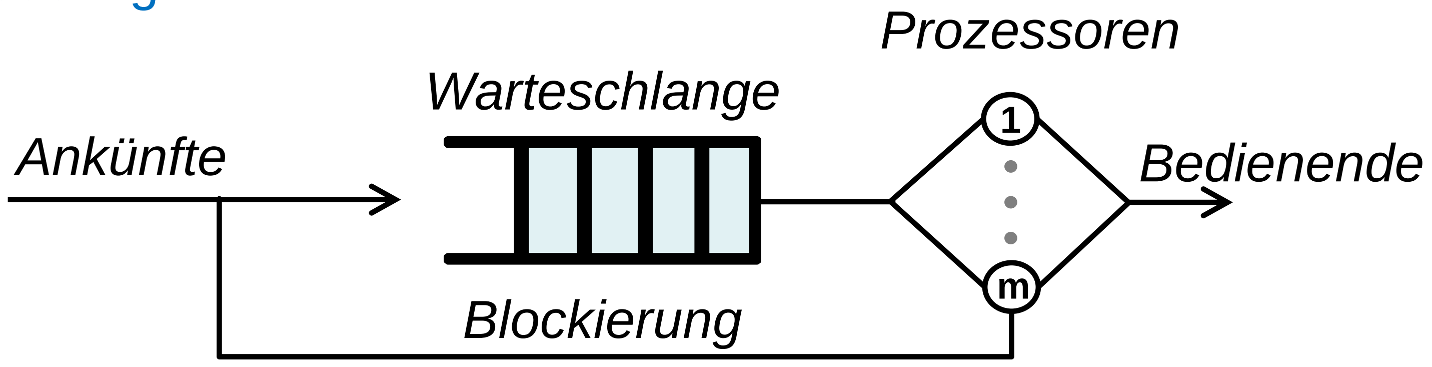


- First In First Out (FIFO):
  - Reihenfolge der Ankünfte bestimmt die Abarbeitung
  - Der am längsten wartende Job wird als nächstes bearbeitet
  - Bei Blockierung muss sich der Prozess wieder einreihen

- Eigenschaften:
  - Sehr einfacher Algorithmus mit hoher CPU-Auslastung
  - Geeignet für Stapelverarbeitung
  - Durchlaufzeiten nicht optimal
  - Probleme bei CPU-lastigen und E/A-lastigen Prozessen $\rightarrow$ **Convoy-Effekt**

Schedulingverfahren – Shortest Job First

- Scheduling:



- Shortest Job First (SJF):
  - Job mit kürzester Arbeitszeit wird zuerst bearbeitet
  - Präemptiv oder nicht-präemptiv

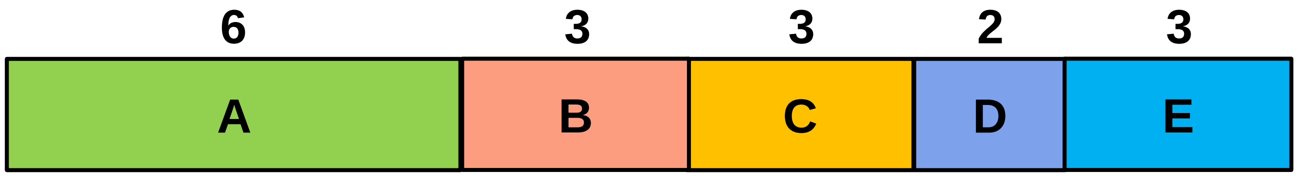
- Eigenschaften:
  - Einfacher Algorithmus
  - Optimale Strategie wenn alle Jobs gleichzeitig vorliegen und Arbeitszeit aller Jobs im voraus bekannt (i.d.R. Stapelverarbeitung)
  - Bei interaktiven Systemen muss die Dauer des CPU-Bursts (Zeit zwischen E/A) geschätzt werden

Schedulingverfahren – Vergleich FIFO und SJF

Beispiel:

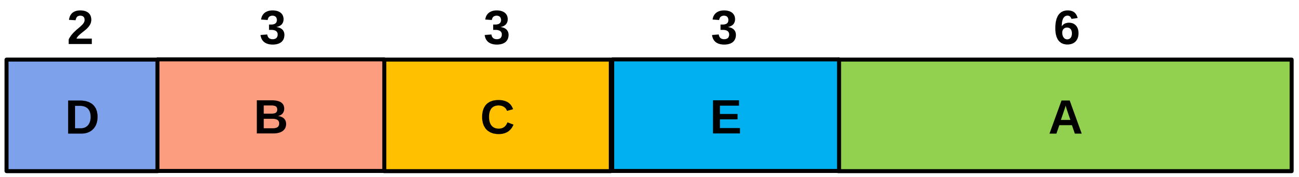
- 5 Jobs (A(6), B(3), C(3), D(2), E(3))
- Alle Jobs sind zum Startzeitpunkt bereit
- Verarbeitungszeit bekannt

FIFO-Reihenfolge



Mittlere Durchlaufzeit: $(6+9+12+14+17)/5=58/5=11.6$

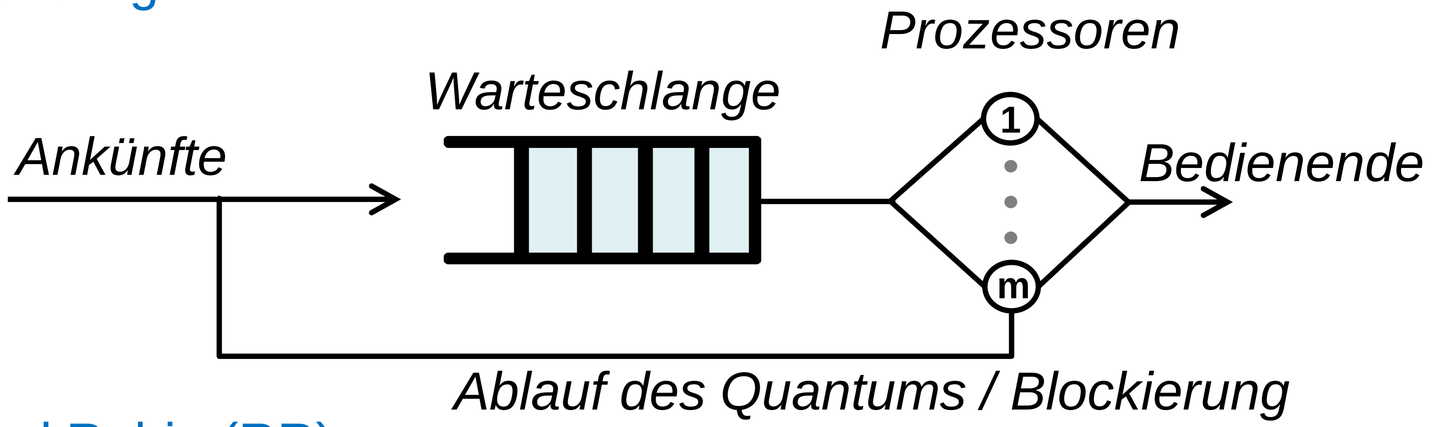
SJF-Reihenfolge



Mittlere Durchlaufzeit: $(2+5+8+11+17)/5=43/5=8.6$

Schedulingverfahren – Round Robin

- Scheduling:



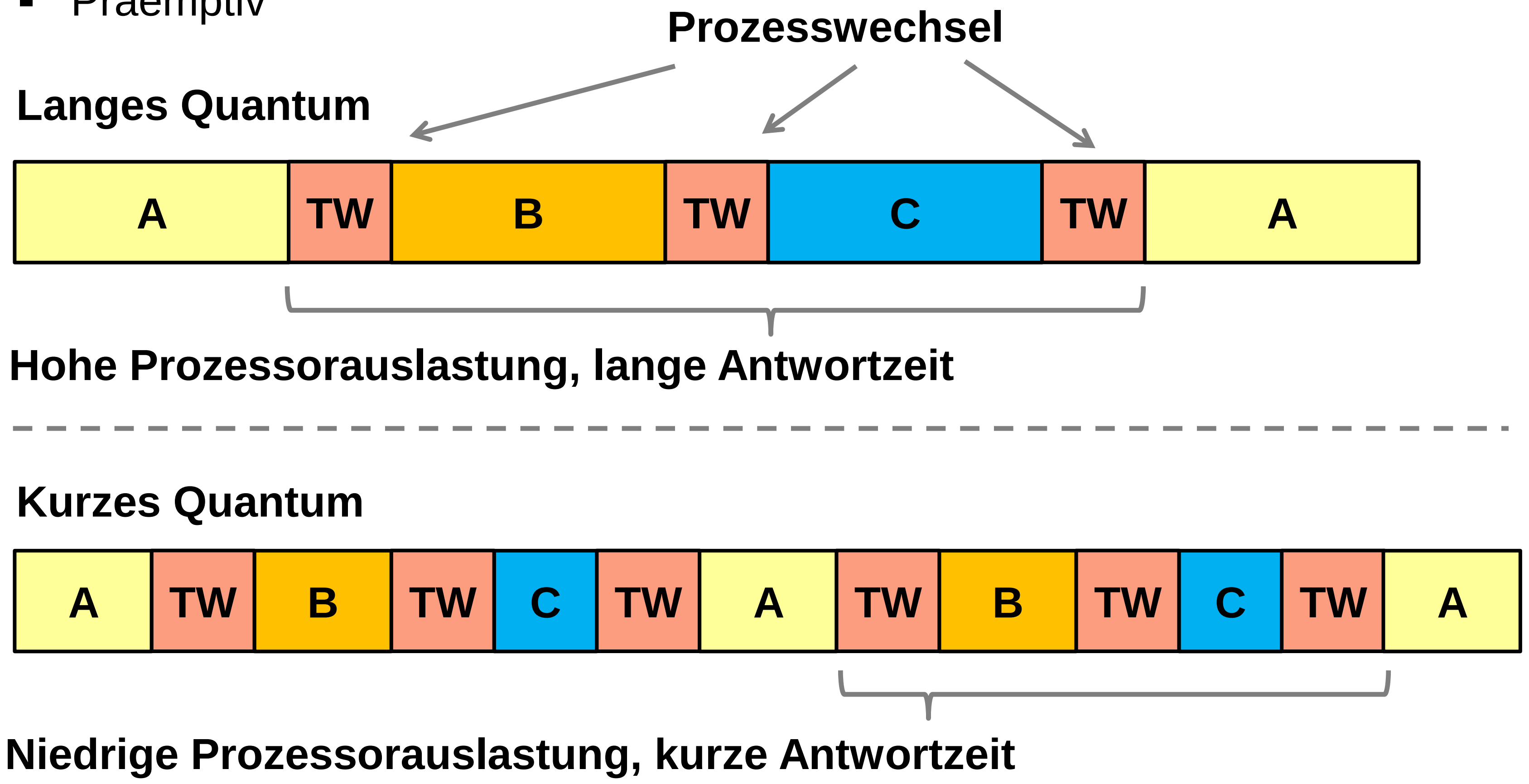
- Round Robin (RR):
  - Job darf höchstens eine bestimmte Zeit (**Quantum, Zeitscheibe**) rechnen
  - Jeder Job kommt der Reihe nach die CPU
  - Präemptive Umsetzung von FIFO

- Eigenschaften:
  - Einfacher Algorithmus (FIFO-Warteschlange und Timer-Interrupt)
  - Länge des Quantums beeinflusst die Effizienz (Taskwechsel)
  - Benachteiligung von E/A-lastigen Prozessen

Schedulingverfahren – Round Robin – Quantum

Beispiel:

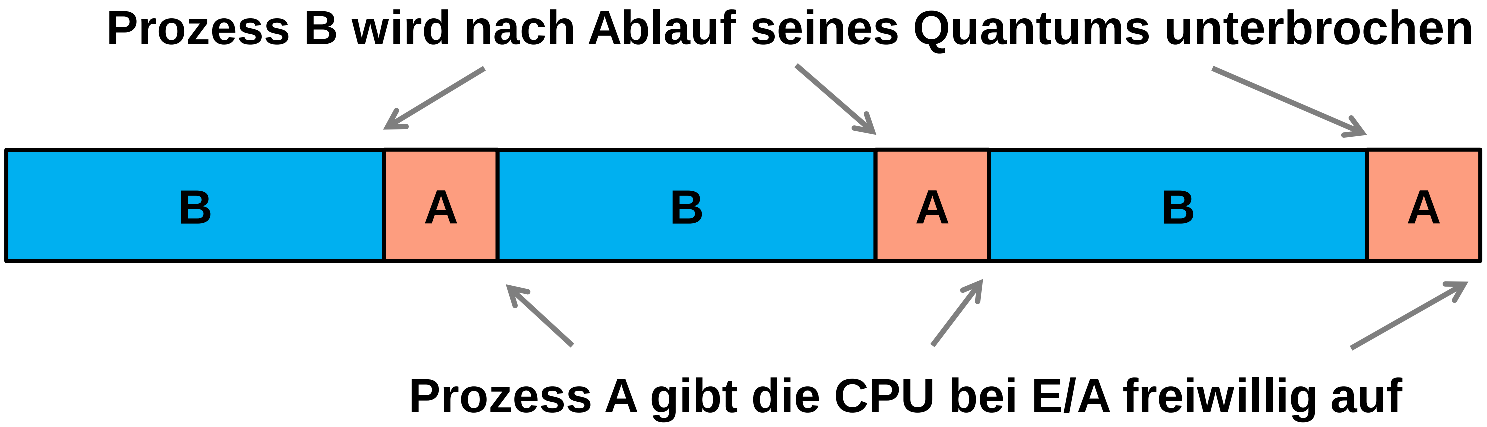
- Round Robin
- 3 Prozesse
- Präemptiv


Prozesswechsel
Langes Quantum
A
TW
B
TW
C
TW
A
Hohe Prozessorauslastung, lange Antwortzeit
Kurzes Quantum
A
TW
B
TW
C
TW
A
TW
B
TW
C
TW
A
Niedrige Prozessorauslastung, kurze Antwortzeit

Schedulingverfahren – Round Robin – Fairness

Beispiel:

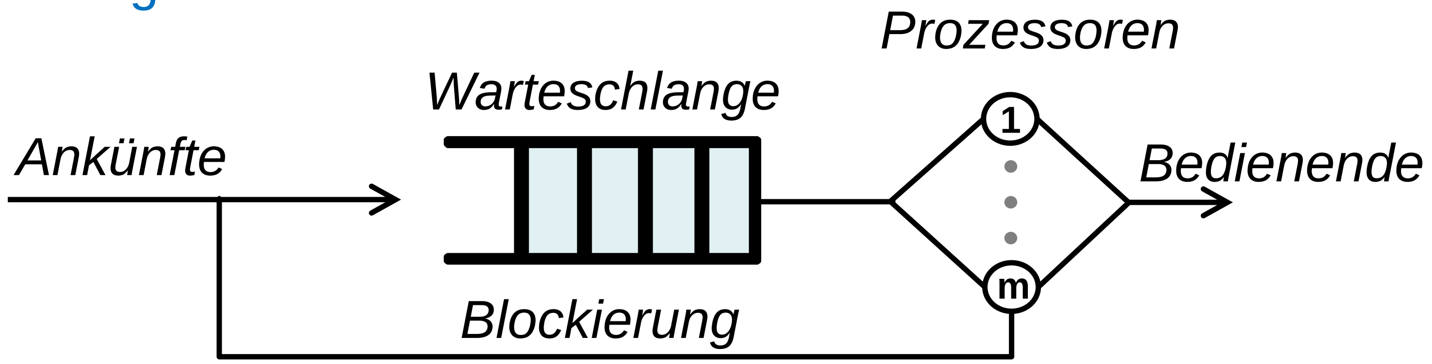
- Round Robin - Fairness
- 2 Prozesse
- Prozess A ist E/A-lastig, Prozess B ist CPU-lastig


Prozess B wird nach Ablauf seines Quantums unterbrochen
B
A
B
A
B
A
Prozess A gibt die CPU bei E/A freiwillig auf

⇒ **Prozess A wird im Vergleich zu Prozess B benachteiligt** ❗

Schedulingverfahren – Earliest Deadline First

- Scheduling:



- Earliest Deadline First (EDF):
  - Priorisierung erfolgt anhand der maximalen Zeitschranke
  - Prioritäten werden dynamisch zur Laufzeit vergeben

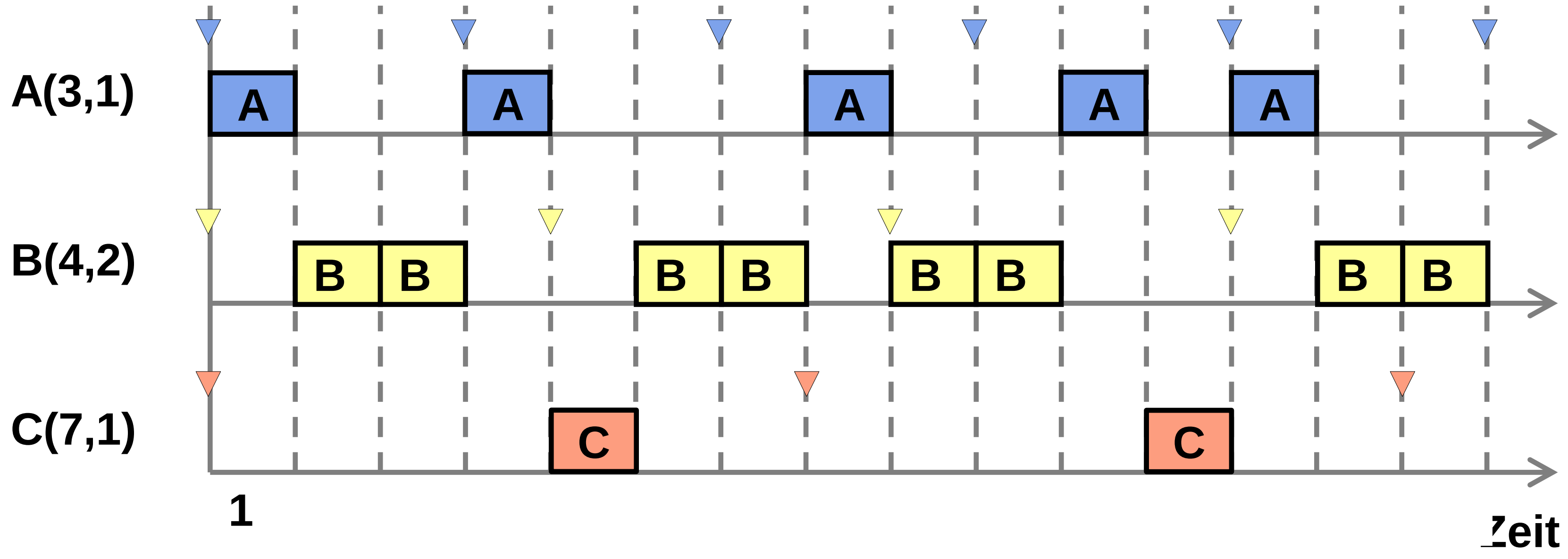
- Eigenschaften:
  - EDF ist ein optimales Schedulingverfahren
  - Dynamische Prioritätenzuweisung
  - Höherer Aufwand als bei statischer Prioritätenvergabe

Schedulingverfahren – Earliest Deadline First

Beispiel:

- 3 Jobs (A(3,1), B(4,2), C(7,1)
- Alle Jobs sind zum Startzeitpunkt bereit
- Verarbeitungszeit bekannt
- Deadline und Periodendauer sind identisch

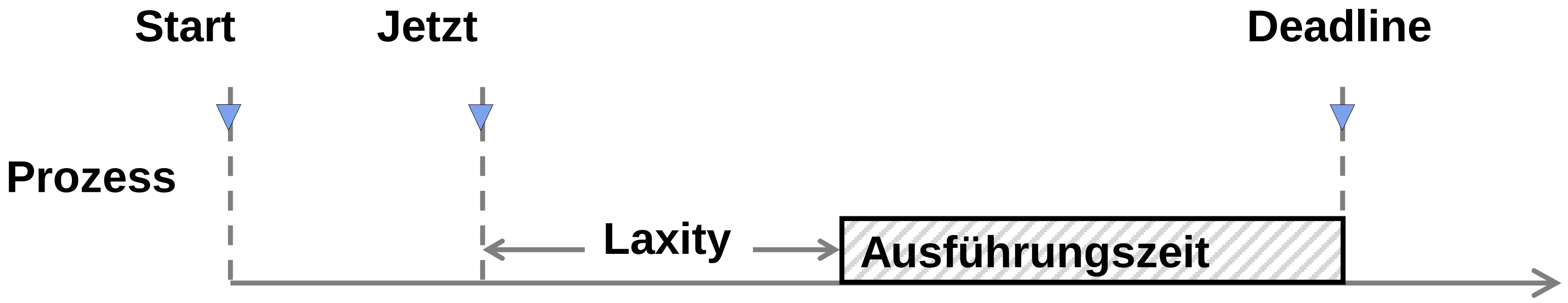
EDF-Scheduling



Schedulingverfahren – Least Laxity First (1/2)

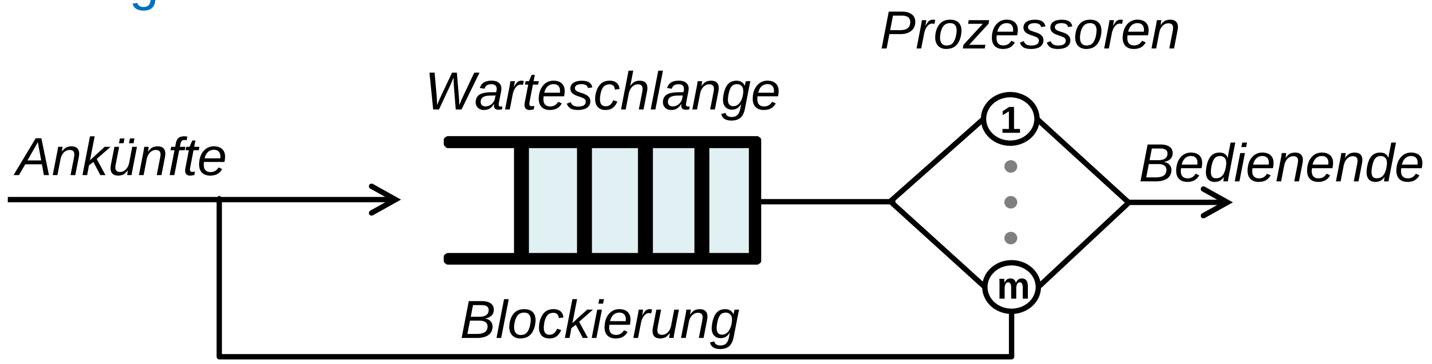
Laxity:

Laxity beschreibt die Zeit, welche einem Prozess bis zu seiner Deadline bleiben würde, wenn er ab sofort bis zu seiner Beendigung den Prozessor verwenden könnte.



Schedulingverfahren – Least Laxity First (2/2)

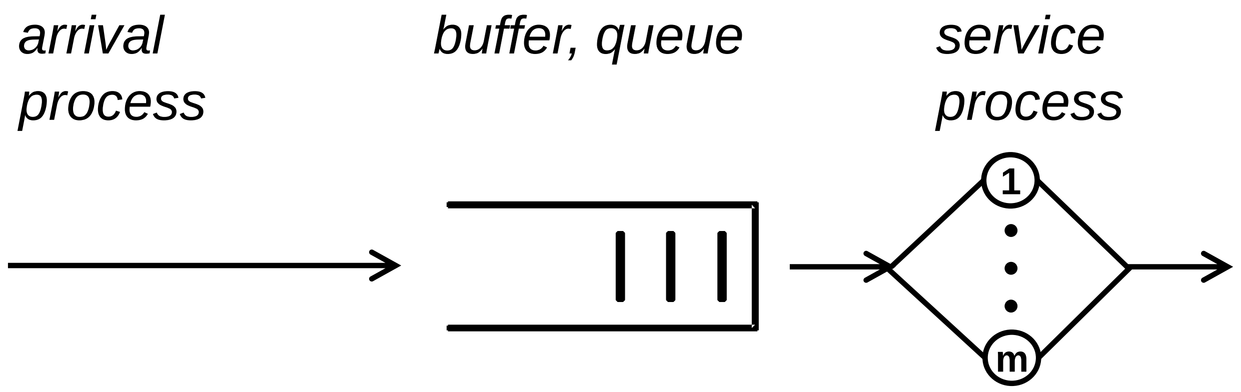
- Scheduling:



- Least Laxity First (LLF):
  - Prozess mit niedrigster Laxity bekommt höchste Priorität
  - Prioritäten werden dynamisch zur Laufzeit vergeben

- Eigenschaften:
  - Erkennt frühzeitig wenn eine Zeitschranke nicht mehr gehalten werden kann
  - Dynamische Prioritätenzuweisung
  - Höherer Rechenaufwand als EDF
  - Häufige Kontextwechsel

Queuing Systems



- Steady state probability π_K :

$$\pi_K = P[\text{there are } k \text{ jobs in the system}]$$

Waiting Systems

- Utilization ρ :

  - Single server $m = 1$:

$$\rho = \frac{\lambda}{\mu}$$

  - Multiple server $m > 1$:

    $m\mu =$ service rate of m servers

$$\rho = \frac{\lambda}{m\mu}$$

  - Condition for stability:

$$\rho < 1 \quad \Rightarrow \quad \lambda < m\mu$$

Waiting Systems

- Throughput η:
  - Number of served jobs per time unit (departure rate)
  - If $\rho < 1$ arrival rate = departure rate

$$\eta = m \cdot \rho \cdot \mu$$

- Response time T :
  - Total time in system (waiting time + service duration)

$$\overline{T} = \overline{W} + \frac{1}{\mu}$$

  - Sojourn time, system time

Waiting Systems

- Waiting time W:
  - Time a job spends in the queue
- Queue length Q:
  - Number of jobs in the queue

Waiting Systems

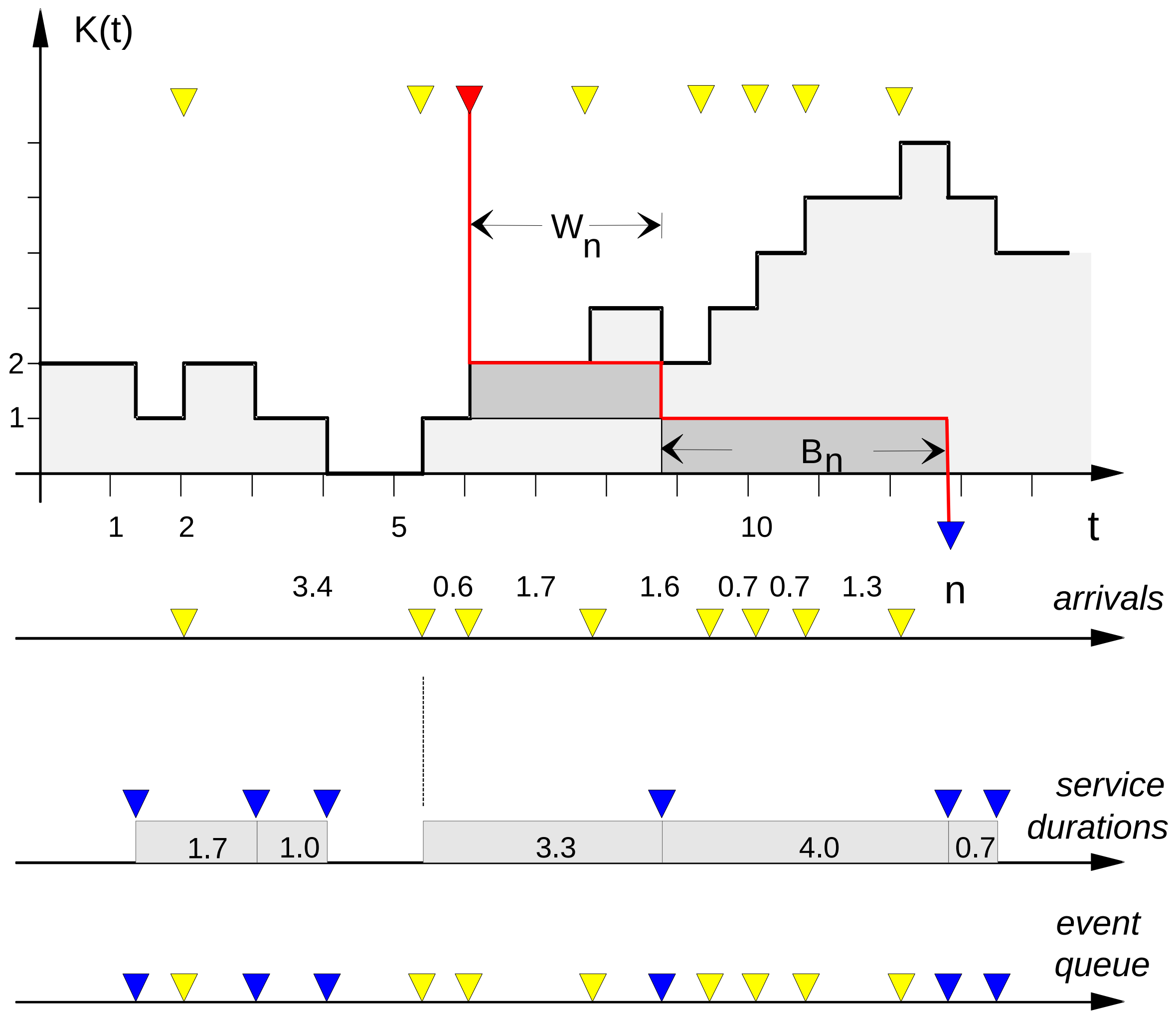
- Number of jobs in the system K :
  - Total number of jobs in the system (queue + servers)
  - Mean number of jobs in the system:

$$\overline{K} = \sum_{k=1}^{\infty} k \cdot \pi_K$$

- Little‘s theorem (Little‘s law):
  - Fundamental theorem of queueing theory
  - Can be used to calculate the mean number of jobs in the system and the waiting queue:
    - » $\overline{K} = \lambda \cdot \overline{T}$
    - » $\overline{Q} = \lambda \cdot \overline{W}$
  - Is one of the following measures known $\overline{K}, \overline{Q}, \overline{T}$ and $\overline{W}$, then the other three can be calculated.

Waiting Systems

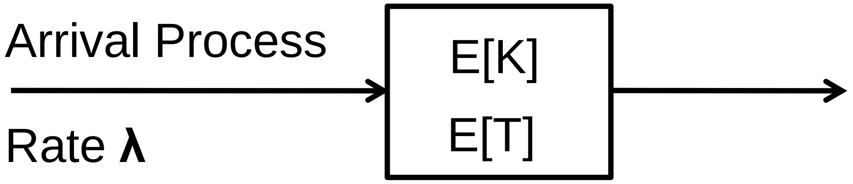
- Little Theorem



Waiting Systems

Little Theorem

- λ : average arrival rate
- E[K] : average number of jobs in the system
- E[T] : average retention time of packets in the system



$$\boxed{t_o \to \infty}$$

$$\bar{T} = \frac{1}{N}\sum_{i=1}^{N} T_i \approx \frac{1}{N}\int_0^{t_o} K(t)dt \qquad \lambda = \lim_{t_o\to\infty} \bar{\lambda} = \lim_{t_o\to\infty} \frac{N}{t_o}$$

$$\bar{K} = \frac{1}{t_o}\int_0^{t_o} K(t)dt \Longrightarrow \bar{K} \approx \frac{N}{t_o} E[T] \qquad E[T] = \lim_{t_o\to\infty} \bar{T} = \lim_{t_o\to\infty} \frac{1}{N}\sum_{i=1}^{N} T_i$$

$$\bar{\lambda} \approx \frac{N}{t_o} \Longrightarrow \bar{\lambda}\cdot\bar{T} \approx \bar{K} \qquad E[K] = \lim_{t_o\to\infty} \bar{K} = \lim_{t_o\to\infty} \frac{1}{t_o}\int_0^{t_o} K(t)dt$$

Waiting Systems

- Little Theorem:

  Average number of customers in the system is given by average arrival rate and the average retention time.

$$\overline{\lambda} \approx \frac{N}{t_o} \quad \Longrightarrow \quad \overline{\lambda} \cdot \overline{T} \approx \overline{K}$$

Characteristics:

- Little Theorem only holds for waiting systems.
- But it can be used as an approximation for blocking systems, if the number of blocked customers is much smaller than the number of served customers.

! Little Theorem applies for any scheduling strategy, inter-arrival time distribution and service time distribution.

Waiting Systems

- Important formulas:
  - Utilization: $$\rho = \frac{\lambda}{m\mu}$$
  - Little's Law: $$\overline{K} = \lambda \cdot \overline{T}$$ $$\overline{Q} = \lambda \cdot \overline{W}$$
  - Mean response time: $$\overline{T} = \overline{W} + \frac{1}{\mu}$$
  - Mean number of jobs: $$\overline{K} = \sum_{k=1}^{\infty} k \cdot \pi_K$$ $$\overline{K} = \overline{Q} + m\rho$$

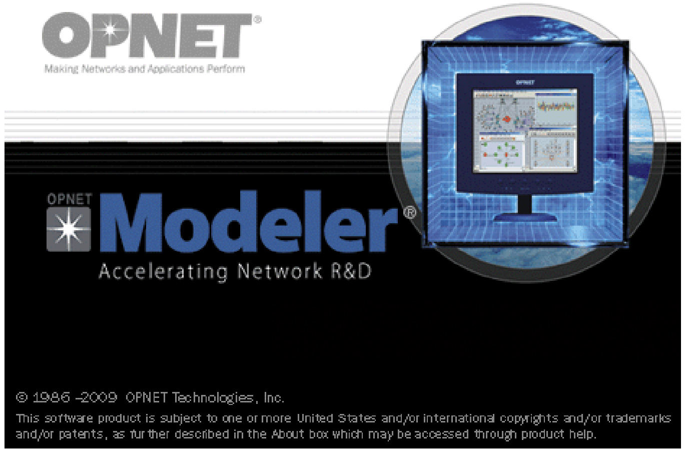
Waiting Systems

- **Questions:**
  - How does the number of service units affect the system?
  - What impact has a higher variance of the arrival and/or service process on the performance of the system?
  - Which system has a higher utilization? One with an unlimited number of waiting slots or one with a limited number?
  - Which system has a lower retention time? One with many slow serving units or on with a single but fast serving unit?
  - How does the queuing strategy (FIFO, LIFO, EDF) affect the average waiting time and the waiting time distribution?

Waiting Systems

- **System Characteristics**
  - Average customer waiting time
  - Average processing time of a customer
  - Average retention time of a customer
  - Average number of customers in the queue
  - Customer blocking probability
  - Utilization of the system / individual processing units

- Example



© 1986 -2009 OPNET Technologies, Inc.
This software product is subject to one or more United States and/or international copyrights and/or trademarks and/or patents, as further described in the About box which may be accessed through product help.

How to model and evaluate waiting queues in OPNET