UNIVERSITY OF TARTU
Institute of Computer Science

LTAT.06.007 Distributed Systems

Lecture 4 – Processes II (From Systems to Descriptive Models)

Huber Flores, PhD

ASSOCIATE PROFESSOR

Tartu, Estonia 09/03/2020

Recap

- Learned how processes are related to distributed systems
- Explored performance modelling for client/server applications
  - Availability
  - Throughput
  - and others…

Agenda

- Processes
  - Migration
- Modelling
  - Performance laws
  - Quantifying performance

Process migration

Basic idea

- Move a piece of code execution from one place to another
  - Cyber-foraging
  - Code offloading
  - Code migration
  - Compution offloading

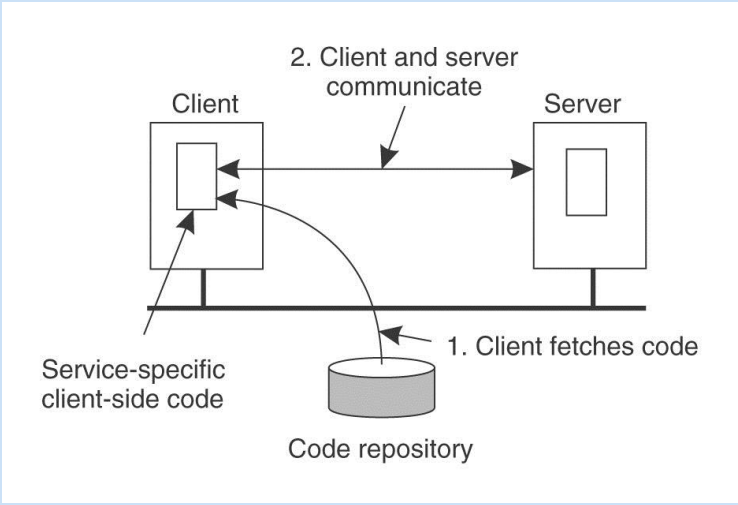
Process migration

Reasons to migrate code

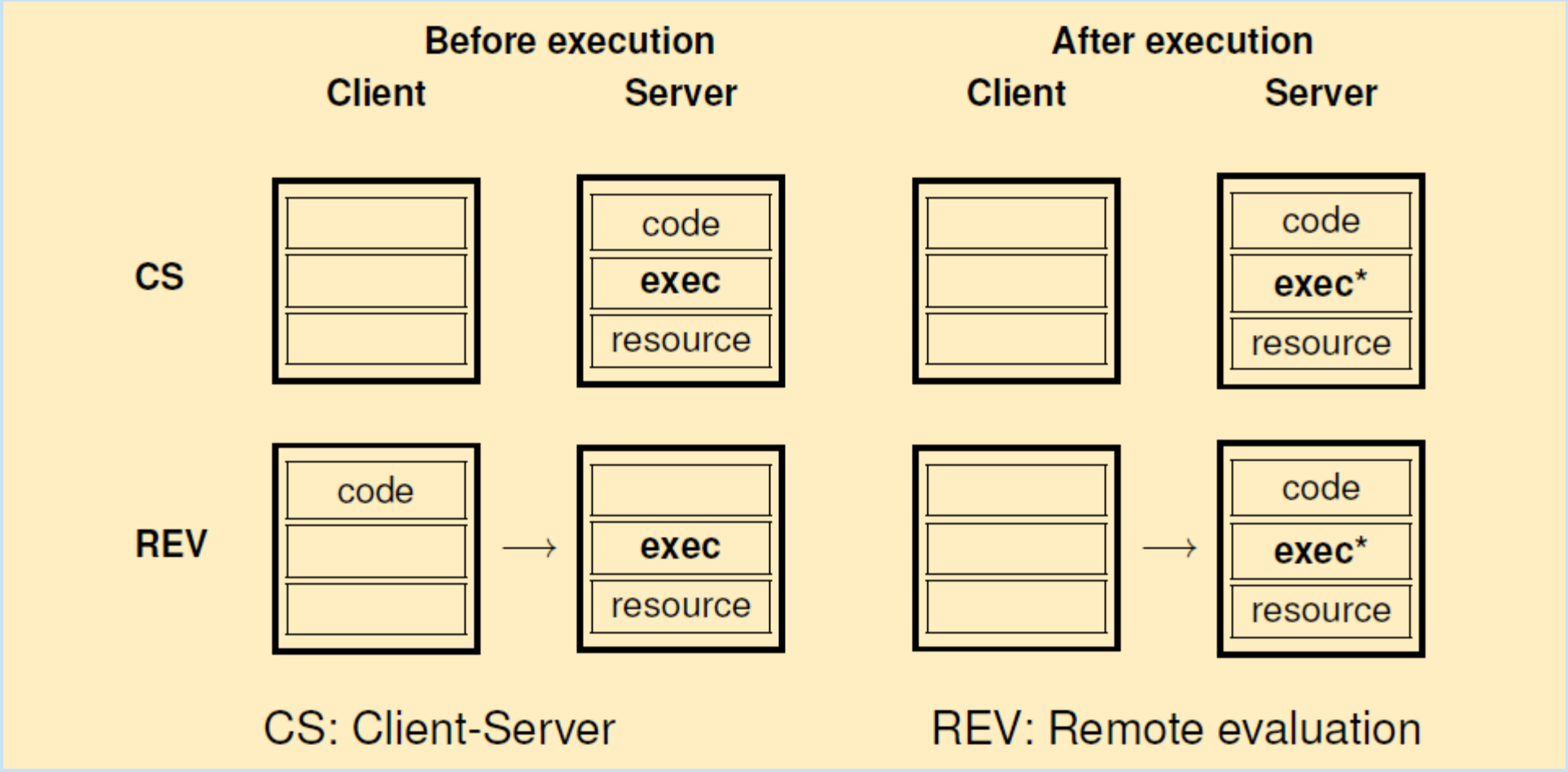
Load distribution

- Ensuring that servers in a datacenter are sufficiently loaded (e.g., to prevent waste of energy)
- Minimizing communication by ensuring that computations are close to where the data is (think of mobile computing).

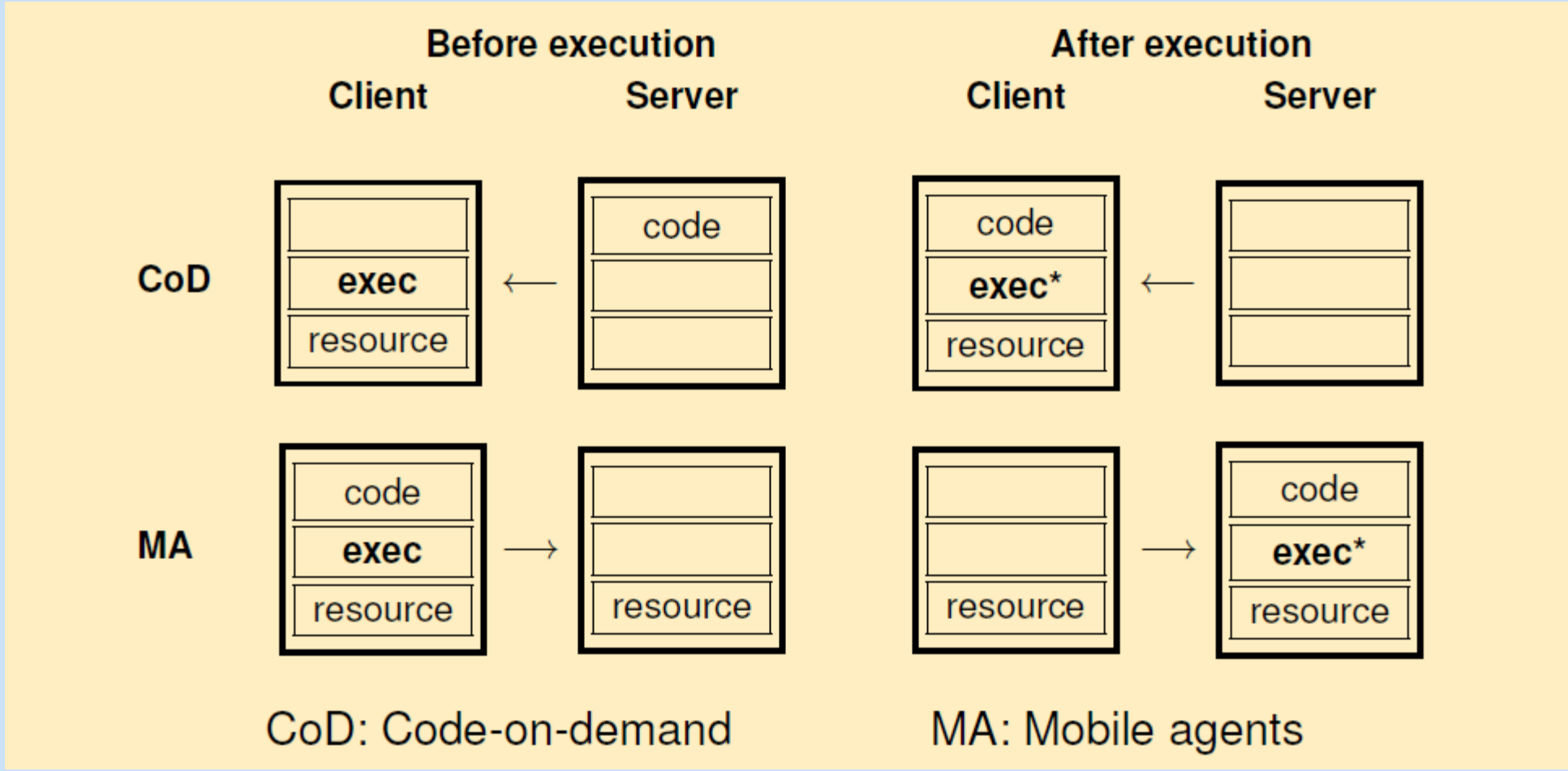
Flexibility: moving code to a client when needed



Models for code migration



Models for code migration



Strong and weak mobility

Object components

- **Code segment:** contains the actual code
- **Data segment:** contains the state
- **Execution state:** contains context of thread executing the object's code

Weak mobility: Move only code and data segment (and reboot execution)

- Relatively simple, especially if code is portable
- Distinguish code shipping (push) from code fetching (pull)

Strong mobility: Move component, including execution state

- **Migration:** move entire object from one machine to the other
- **Cloning:** start a clone, and set it in the same execution state.

Mobility types

Sender-initiated mobility vs receiver-initiated mobility

- Sender-initiated (remote evaluation, mobile agents)
  - Mobility initiated by machine where the code currently resides or is being executed
  - E.g. uploading programs to a computing server
- Receiver-initiated (code-on-demand)
  - Mobility initiated by target machine (client), e.g. Java applets downloaded from a web server
- Receiver-initiated mobility (code moves from server to client) often simpler to implement
  - Server's resources have to be protected from (malicious) clients
  - Server is not typically interested in clients resources

Migration in heterogenous systems

Main problem

- The target machine may not be suitable to execute the migrated code
- The definition of process/thread/processor context is highly dependent on local hardware, operating system and runtime system

Only solution: abstract machine implemented on different platforms

- Interpreted languages, effectively having their own VM
- Virtual machine monitors

Migrating a virtual machine

Migrating images: three alternatives

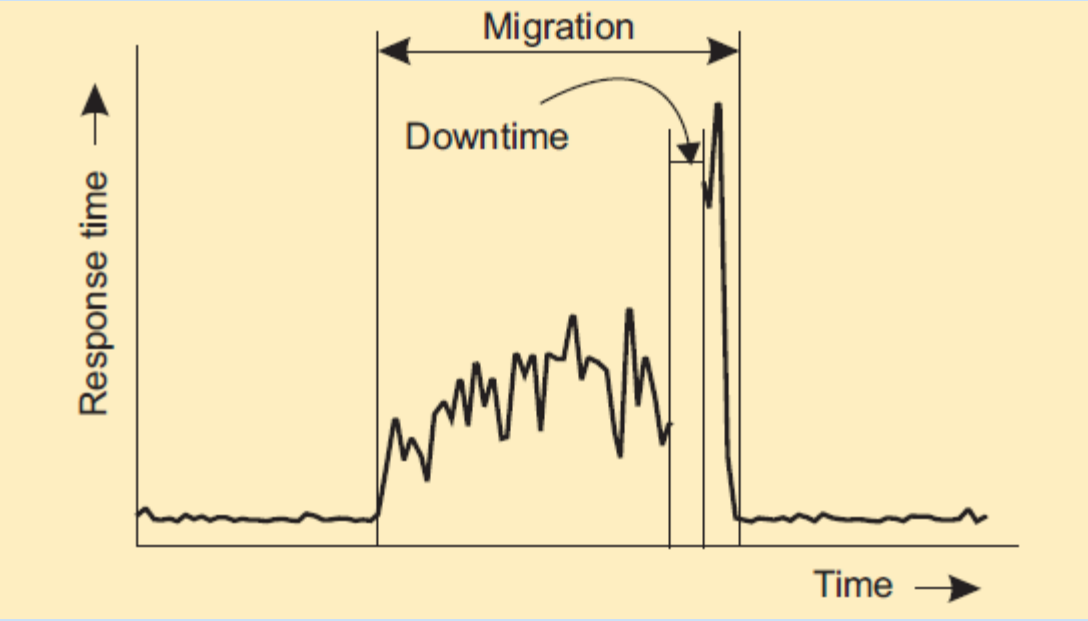
- Pushing memory pages to the new machine and resending the ones that are later modified during the migration process.
- Stopping the current virtual machine; migrate memory, and start the new virtual machine.
- Letting the new virtual machine pull in new pages as needed: processes start on the new virtual machine immediately and copy memory pages on demand.

Performance of migrating virtual machines

Problem

A complete migration may actually take tens of seconds. We also need to realize that during the migration, a service will be completely unavailable for multiple seconds.

Measurements regarding response times during VM migration



Modelling performance

Essence

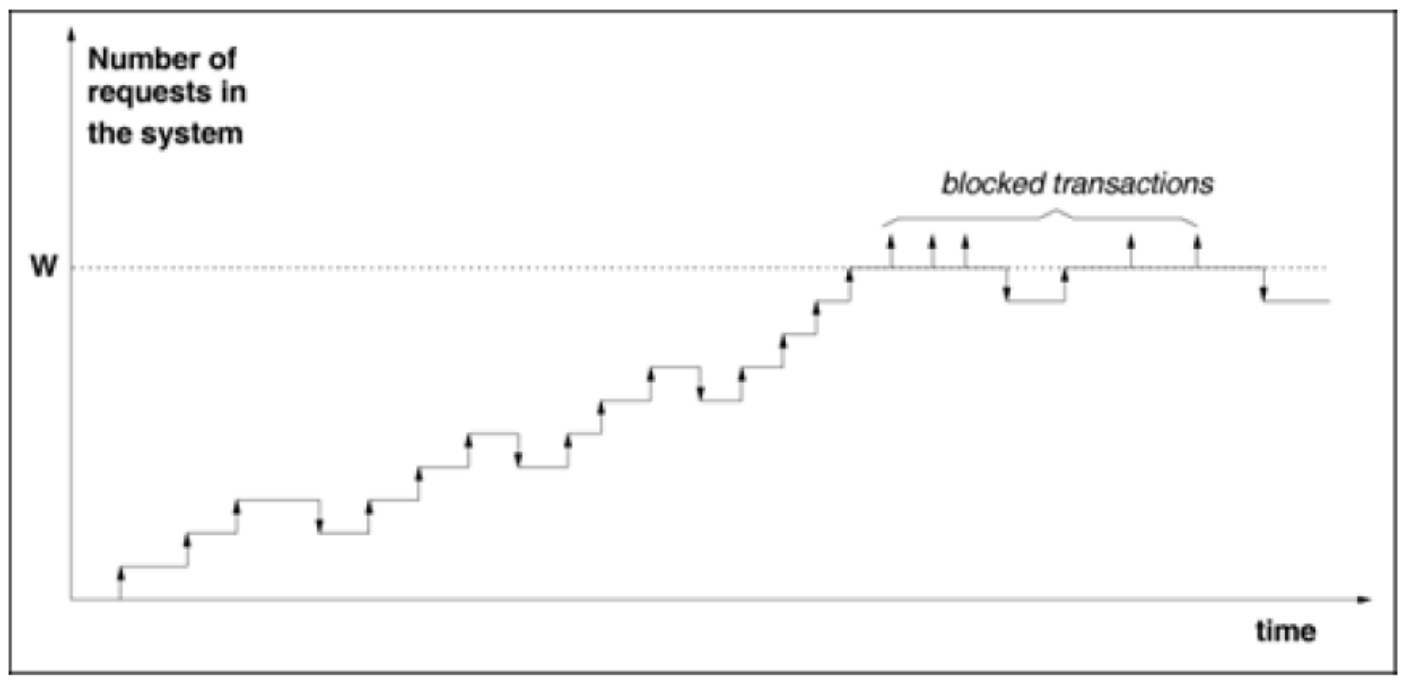
- Estimation of performance during runtime is critical to schedule pending processing tasks.
- Processing of tasks must be scheduled in such a way that does not disrupt nor degrades performance
- The performance of a (distributed) computer system can be balanced among its existing parts
- Performance is modelled as QN (Queuing Network)

Modelling performance: Blocking

Importance

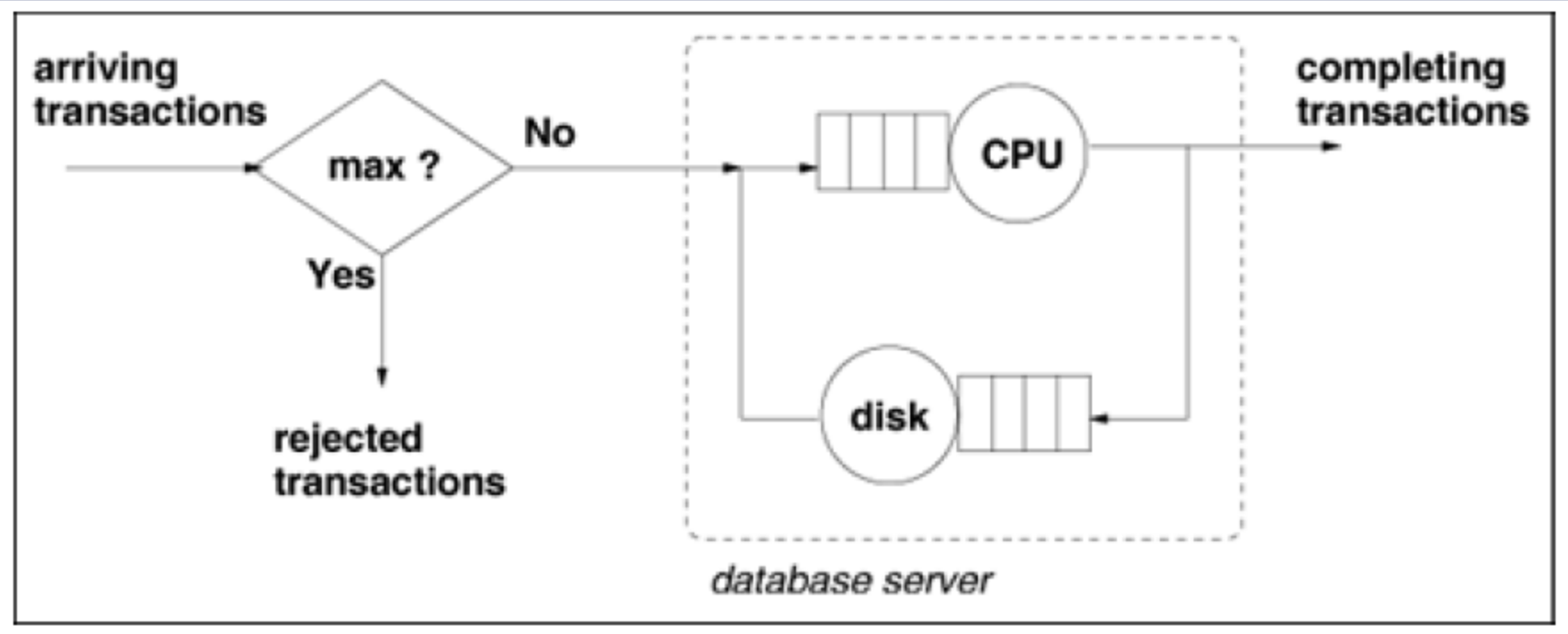
- Admission control policies to regulate the usage of resources

Example: Number of transactions in the database server vs. time with admission control.



Modelling performance: Blocking

Database server QN with admission control.

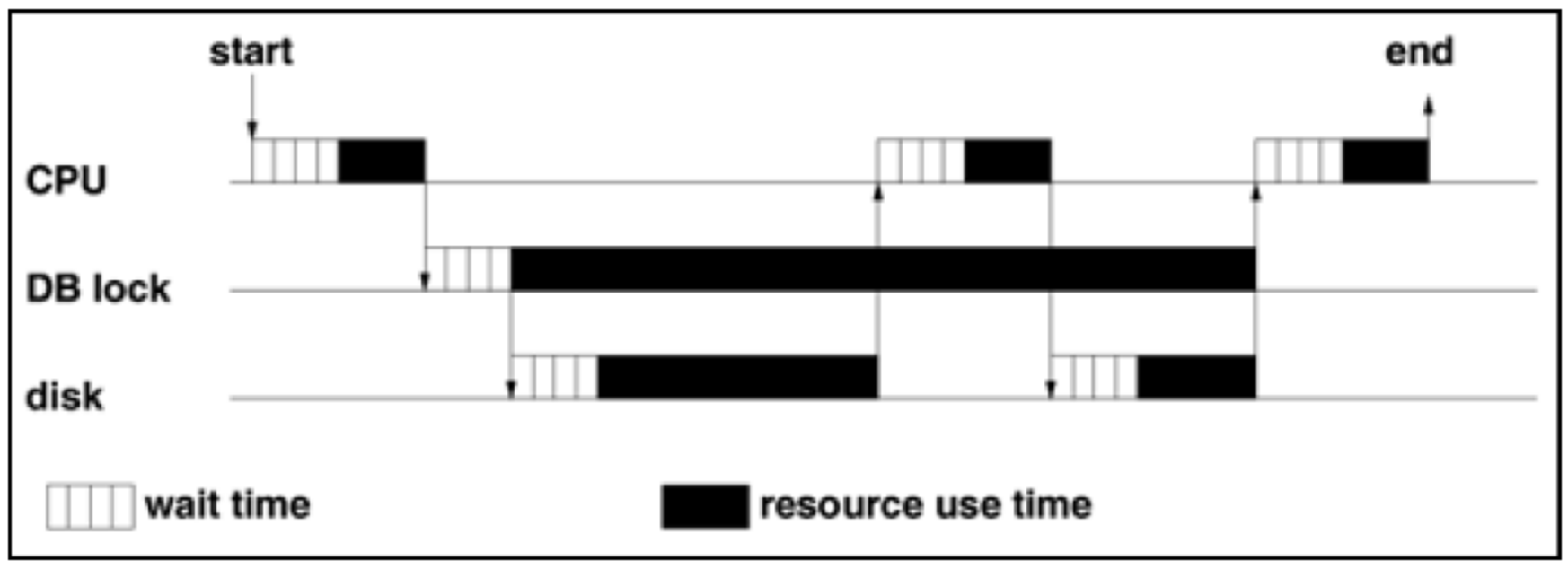


Simultaneous resource possession

Importance

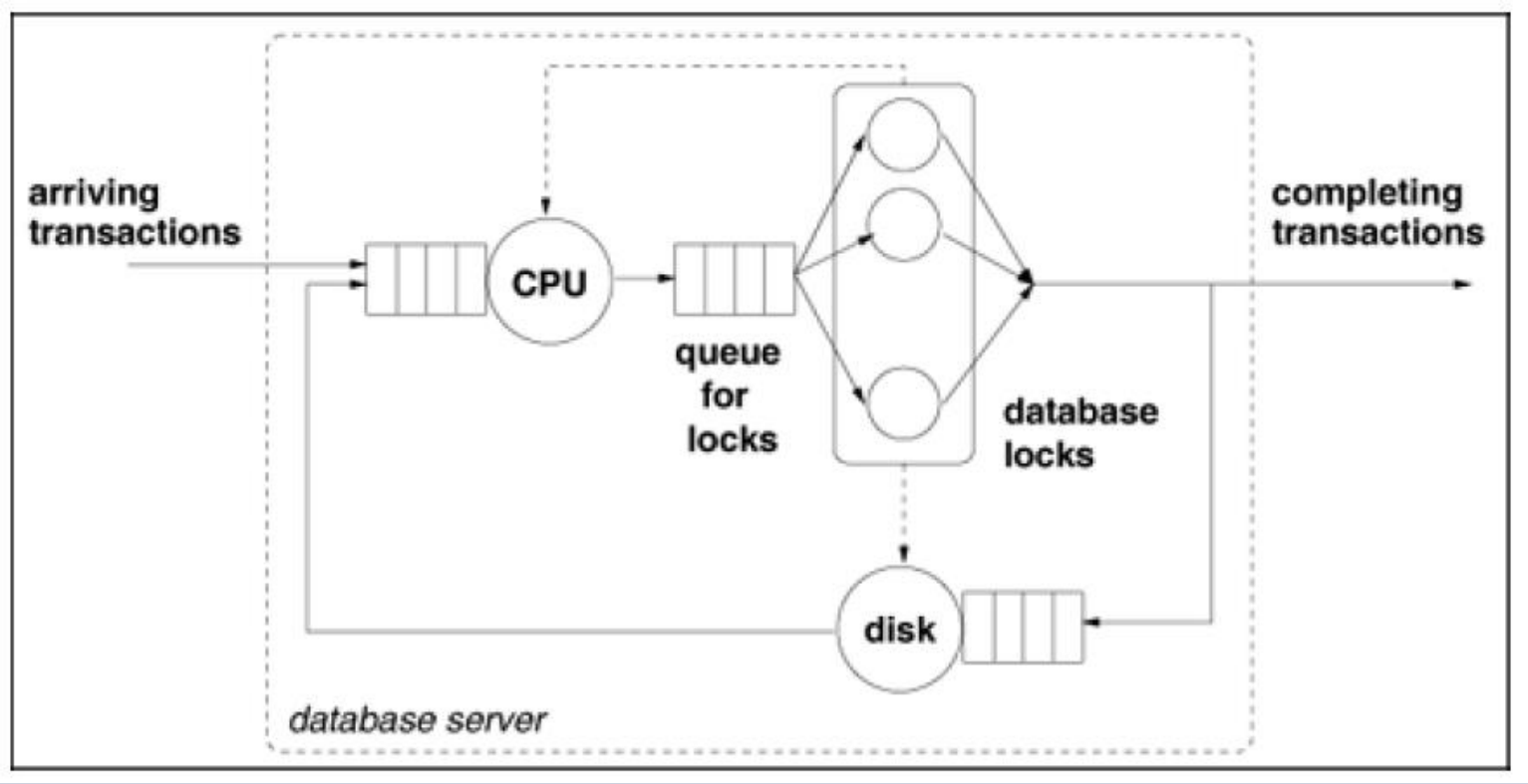
- Resources are locked to be use at the same time by the same request.

Database server QN with admission control.



Simultaneous resource possession

QN with simultaneous resource possession. Database locks are held simultaneously with the CPU and with the disk.



Queuing disciplines

Many different queuing disciplines can be considered in a QN

- First Come First Served (FCFS)
- Priority Queuing (PQ)
- Round Robin (RR)
- Processor Sharing (PS). – RR but concurrently

Quantifying performance models

Operational analysis with QNs

It is used to establish relationships among quantities based on measured or known data about computer systems.

Motivation

Suppose that during an observation period of 1 minute, a single resource (e.g., the CPU) is observed to be busy for 36 sec. A total of 1800 transactions are observed to arrive to the system. The total number of observed completions is 1800 transactions (i.e., as many completions as arrivals occurred in the observation period). What is the performance of the system (e.g., the mean service time per transaction, the utilization of the resource, the system throughput)?

Quantifying performance models

Operational analysis notation (Measurable quantities)

- T: length of time in the observation period
- K: number of resources in the system
- B_i: total busy time of resource i in the observation period T
- A_i: total number of service requests (i.e., arrivals) to resource i in the observation period T
- A_0: total number of requests submitted to the system in the observation period T
- C_i: total number of service completions from resource i in the observation period T
- C_0: total number of requests completed by the system in the observation period T

Quantifying performance models

Operational analysis notation (Derived quantities)

- S_i: mean service time per completion at resource i; $S_i = B_i/C_i$
- U_i: utilization of resource i; $U_i = B_i/T$
- X_i: throughput (i.e., completions per unit time) of resource i; $X_i = C_i/T$
- λ_i: arrival rate (i.e., arrivals per unit time) at resource i; $\lambda_i = A_i/T$
- X_0: system throughput; $X_0 = C_0/T$
- V_i: average number of visits (i.e., the visit count) per request to resource i; $V_i = C_i/C_0$

Quantifying performance models

Note

The previous notation can be easily extended to the multiple class case by considering that R is the number of classes and by adding the class number r ($r = 1, \cdots, R$) to the subscript. For example, $U_{i,r}$ is the utilization of resource i due to requests of class r and $X_{0,r}$ is the throughput of class r requests.

Quantifying performance models

Exercise

Suppose that during an observation period of 1 minute, a single resource (e.g., the CPU) is observed to be busy for 36 sec. A total of 1800 transactions are observed to arrive to the system. The total number of observed completions is 1800 transactions (i.e., as many completions as arrivals occurred in the observation period). What is the performance of the system (e.g., the mean service time per transaction, the utilization of the resource, the system throughput)?

Quantifying performance models

Exercise (Measurable quantities)

$$
\begin{aligned}
\mathcal{T} &= 60 \text{ sec} \\
K &= 1 \text{ resource} \\
B_1 &= 36 \text{ sec} \\
A_1 &= A_0 = 1800 \text{ transactions} \\
C_1 &= C_0 = 1800 \text{ transactions}
\end{aligned}
$$

Quantifying performance models

Exercise (Derivable quantities)

$$S_1 = \frac{B_1}{C_1} = \frac{36}{1800} = \frac{1}{50} \text{ second per transaction}$$

$$U_1 = \frac{B_1}{T} = \frac{36}{60} = 60\%$$

$$\lambda_1 = \frac{A_1}{T} = \frac{1800}{60} = 30 \text{ tps}$$

$$X_0 = \frac{C_0}{T} = \frac{1800}{60} = 30 \text{ tps}$$

Operational laws in performance models

Utilization law

Defines the utilization of a resource

$$U_i = \frac{B_i}{T} = \frac{B_i/C_i}{T/C_i}.$$

$$U_i = S_i \times X_i.$$

Utilization law

Example

The bandwidth of a communication link is 56,000 bps and it is used to transmit 1500-byte packets that flow through the link at a rate of 3 packets/second. What is the utilization of the link?

Utilization law

Example

(K = 1)

throughput X_1 is 3 packets/second

Each packet has 1,500 bytes/packet x 8 bits/byte = 12,000 bits/packet.

12,000 bits/56,000 bits/sec = 0.214 sec
to transmit a packet over this link

S_1 = 0.214 sec/packet

Utilization Law $S_1 \times X_1 = 0.214 \times 3 = 0.642 = 64.2\%$

Operational laws in performance models

Service demand law

The notion of service demand is associated both with a resource and a set of requests using the resource. The service demand, denoted as D_i, is defined as the total average time spent by a typical request of a given type obtaining service from resource *i*.

$$D_i = \frac{U_i \times T}{C_0} = \frac{U_i}{C_0/T} = \frac{U_i}{X_0}.$$

Service demand law

Exercise

A Web server is monitored for 10 minutes and its CPU is observed to be busy 90% of the monitoring period. The Web server log reveals that 30,000 requests are processed in that interval. What is the CPU service demand of requests to the Web server?

Service demand law

Exercise

T is 600 (= 10 x 60) seconds

Web server throughput, X_0 = 30,000/600 = 50 requests/sec

CPU utilization is U_{CPU} = 0.9

service demand at the CPU is $D_{CPU} = U_{CPU}/X_0$

= 0.9/50 = 0.018 seconds/request.

Operational laws in performance models

The forced flaw law

It estimates the throughput of resource i, X_i, from the system throughput, X_0.

$$X_i = V_i \times X_0.$$

Forced flaw law

Exercise

Disk	Reads Per Second	Writes Per Second	Total I/Os Per Second	Utilization
1	24	8	32	0.30
2	28	8	36	0.41
3	40	10	50	0.54

If the server throughput is 3.8 tps, then what is the average number of I/Os on each disk?

Forced flaw law

Exercise

$V_1 = X_1/X_0$

$= 32/3.8 = 8.4$ visits to disk 1 per database transaction

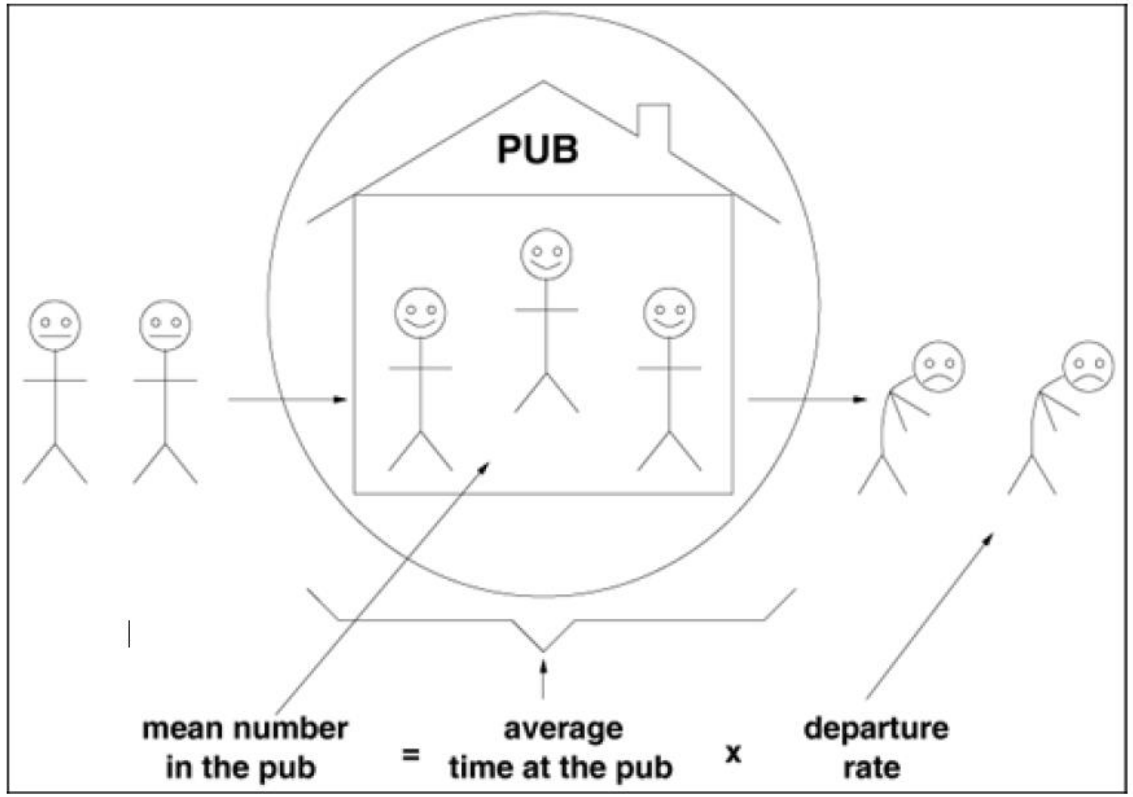
$V_2 = X_2/X_0 = 36/3.8 = 9.5$

$V_3 = X_3/X_0 = 50/3.8 = 13.2$

Operational laws in performance models

Little's law

Consider this analogy



$$N = X \times R$$

Little’s law

Exercise

Consider server throughput of a database is 3.8 tps, and during one hour measurement observation interval, the average number of database transactions in execution was 16.

What was the response time of database transactions during that measurement interval?

Little's law

Exercise

average number in the box is the average number N of concurrent database transactions in execution

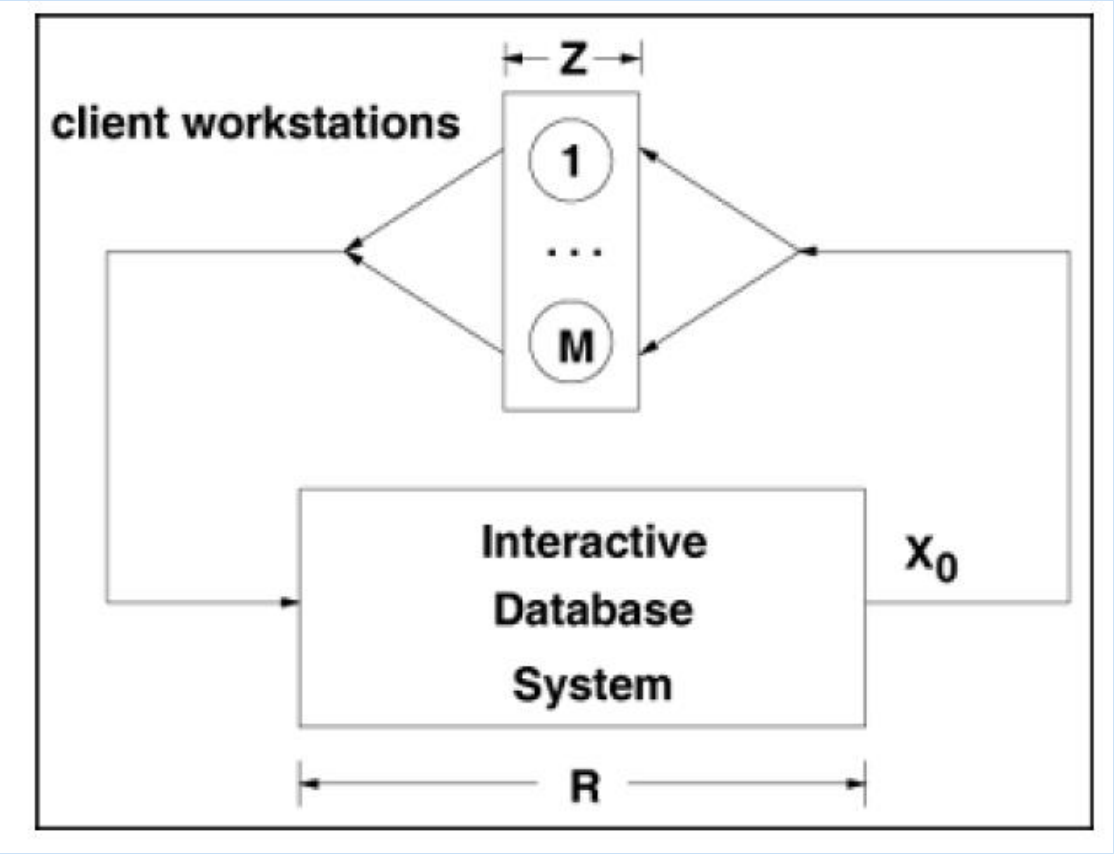
$= 16$

average time in the box is the average response time R desired

$R = N/X_0 = 16/3.8 = 4.2$ sec

Operational laws in performance models

Interactive response time law



$$R = \frac{M}{X_0} - Z$$

Interactive response time law

Exercise

If 7,200 requests are processed during one hour by an interactive computer system with 40 clients and an average think time of 15 sec, the average response time is?

Interactive response time law

Exercise

$$R = \frac{40}{7200/3600} - 15 = 5 \text{ sec.}$$

Operational laws in performance models

Summary of operational laws

Utilization Law:

$$U_i = X_i \times S_i = \lambda_i \times S_i$$

Forced Flow Law:

$$X_i = V_i \times X_0$$

Service Demand Law:

$$D_i = V_i \times S_i = U_i / X_0$$

Little's Law:

$$N = X \times R$$

Interactive Response Time Law

$$R = \frac{M}{X_0} - Z$$

Further readings

- Flores, Huber, Pan Hui, Sasu Tarkoma, Yong Li, et al***. "Mobile code offloading: from concept to practice and beyond."*** *IEEE Communications Magazine* 53, no. 3 (2015): 80-88.

- Flores, Huber, Pan Hui, Petteri Nurmi, Eemil Lagerspetz, Sasu Tarkoma, Jukka Manner, Vassilis Kostakos, Yong Li, and Xiang Su. ***"Evidence-aware mobile computational offloading."*** *IEEE Transactions on Mobile Computing* 17, no. 8 (2017): 1834-1850.

Summary

- Processes
  - Fundamental concepts
- Modelling
  - QoS attributes
  - Server performance
  - Client/Server applications

Next lecture

Communications

Questions?

E-mail: huber.flores@ut.ee