COMP 315
Performance Evaluation

Dr. Yapeng Wang
Email: yapengwang@ipm.edu.mo
Tel: 85996432
Office: A313, Chi Un Building

Introduction and Welcome

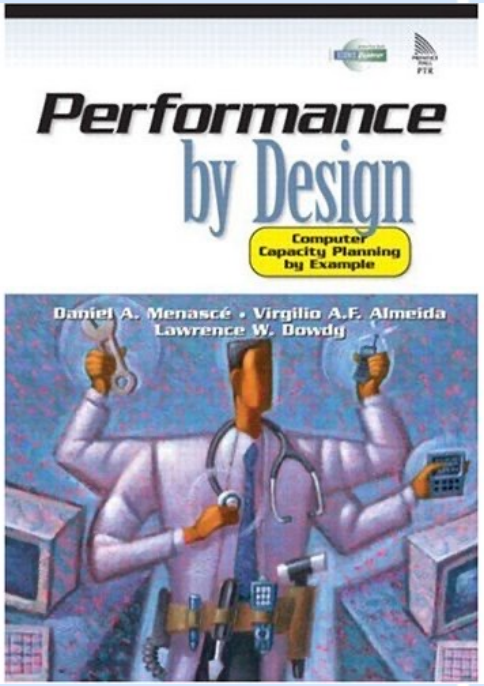
Brief Introduction of ME
Brief Introduction of YOU
Brief Introduction of this COURSE

Course Description

The aim of this course is to provide students with the main concepts and techniques needed to study the performance of computer systems, plan the capacity of computer systems, predict their future performance under different configurations, and design new applications that meet performance requirements. The course is mainly based on the use of analytic queuing network models of computer systems.

Main Textbook

- **Performance by Design: Computer Capacity Planning by Example,**
  - by Daniel A. Menasce, Virgilio A.F. Almeida, and Lawrence W. Dowdy
  - E-version will be available on canvas



Course Outline

1. Introduction
2. Computer System Lifecycle
3. Descriptive Models
4. Basic Performance Laws
5. Performance Engineering Methodology
6. Evaluation Database Service
7. Markov Models
8. Single Queue Systems
9. Single Class MVA (Mean Value Analysis)

Grading Method

Assignments 30%
Test 20%
Final Exam 50%

Chapter 1
Computer System Lifecycle

Performance by Design: Computer Capacity Planning by Example

Daniel A. Menascé, Virgilio A.F. Almeida, Lawrence W. Dowdy
Prentice Hall, 2004

Outline

1. Introduction
2. QoS in IT Systems
3. System Life Cycle
4. A Reference Model for IT Systems
5. Conclusions

Introduction – Why we need to take this course? The Importance of Performance in Computer Systems

- Mission-critical applications
- Life support applications
- Homeland security
- Battlefield situations
- Personal communication systems

QoS (Quality of Service) Metrics

- Response time
- Throughput
- Availability
- Reliability
- Security
- Scalability
- Extensibility

Response Time Breakdown

Browser Time		Network Time			E-commerce Server Time		
Processing	I/O	Browser to ISP Time	Internet Time	ISP to Server Time	Processing	I/O	Networking

CONGESTION

- Service time (does not depend on the load)
- Congestion (load-dependent)

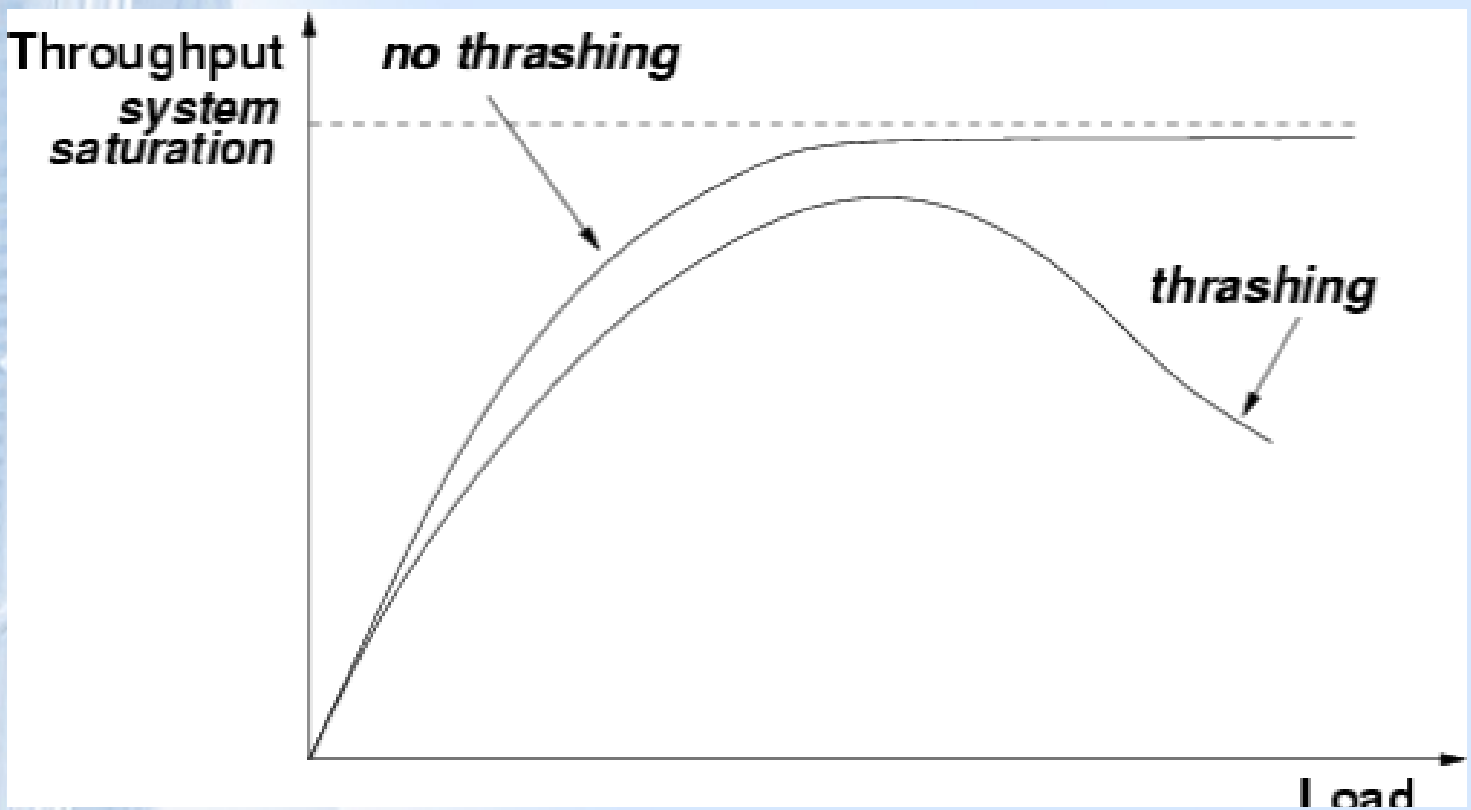
Throughput

- Measured in units of work completed over time. It's a rate.
  - I/O's/sec
  - Page downloads/sec
  - HTTP requests/sec
  - Jobs/sec
  - Transactions per second (tps)

Throughput Example

- An I/O operation at a disk of an OLTP system takes 10 msec on average.
  - What is the maximum throughput of the disk?
  - What is the throughput of the disk if it receives I/O requests at a rate of 80 requests/sec?

Throughput example

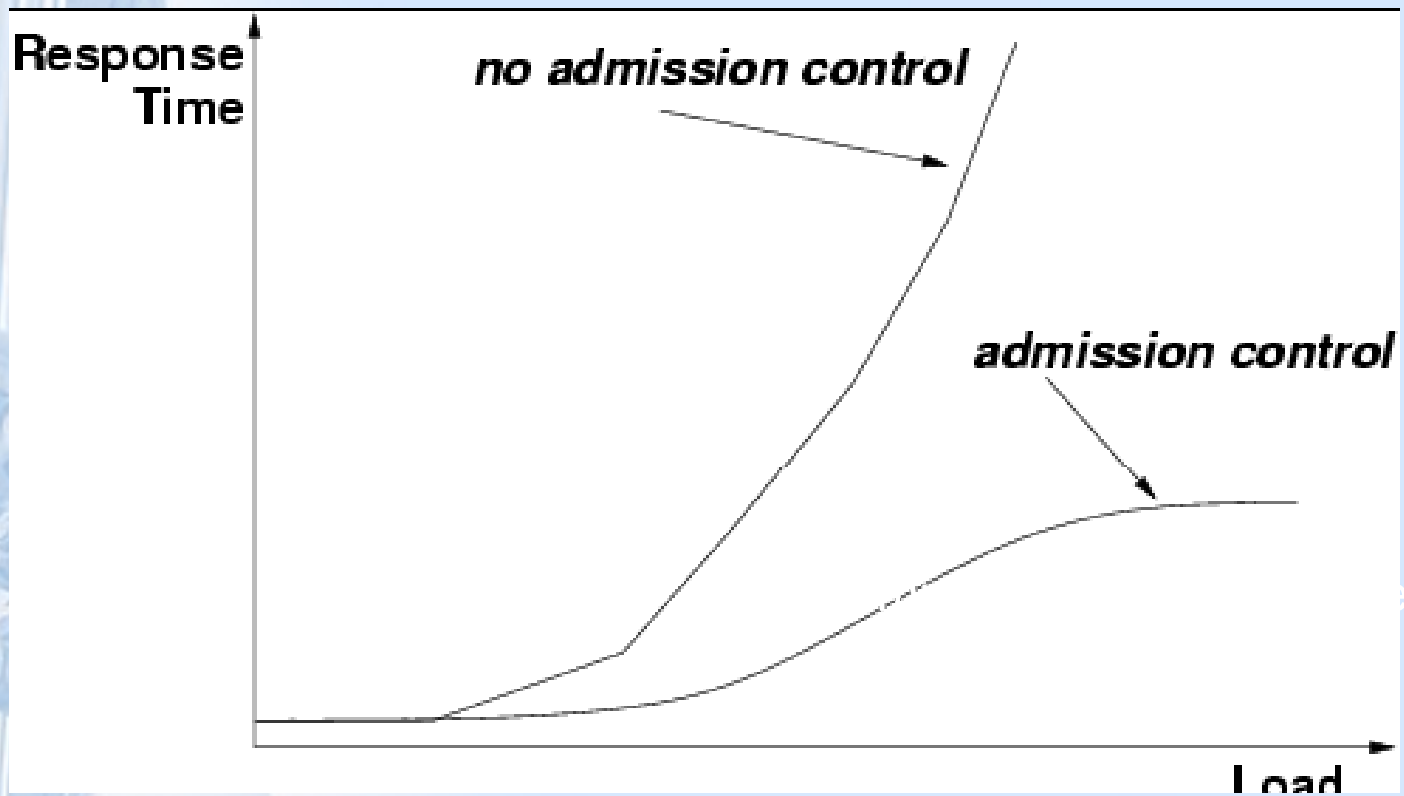


Availability

- Fraction of time a system is available (i.e.,operational).
  - Service interruptions can damage the reputation of a company, may endanger lives, and may cause financial disasters.
  - A system with 99.99% availability over 30 days is unavailable (1-0.9999) x 30 x 24 x 60 = 4.32 minutes.

Availability Problems

Admission Control



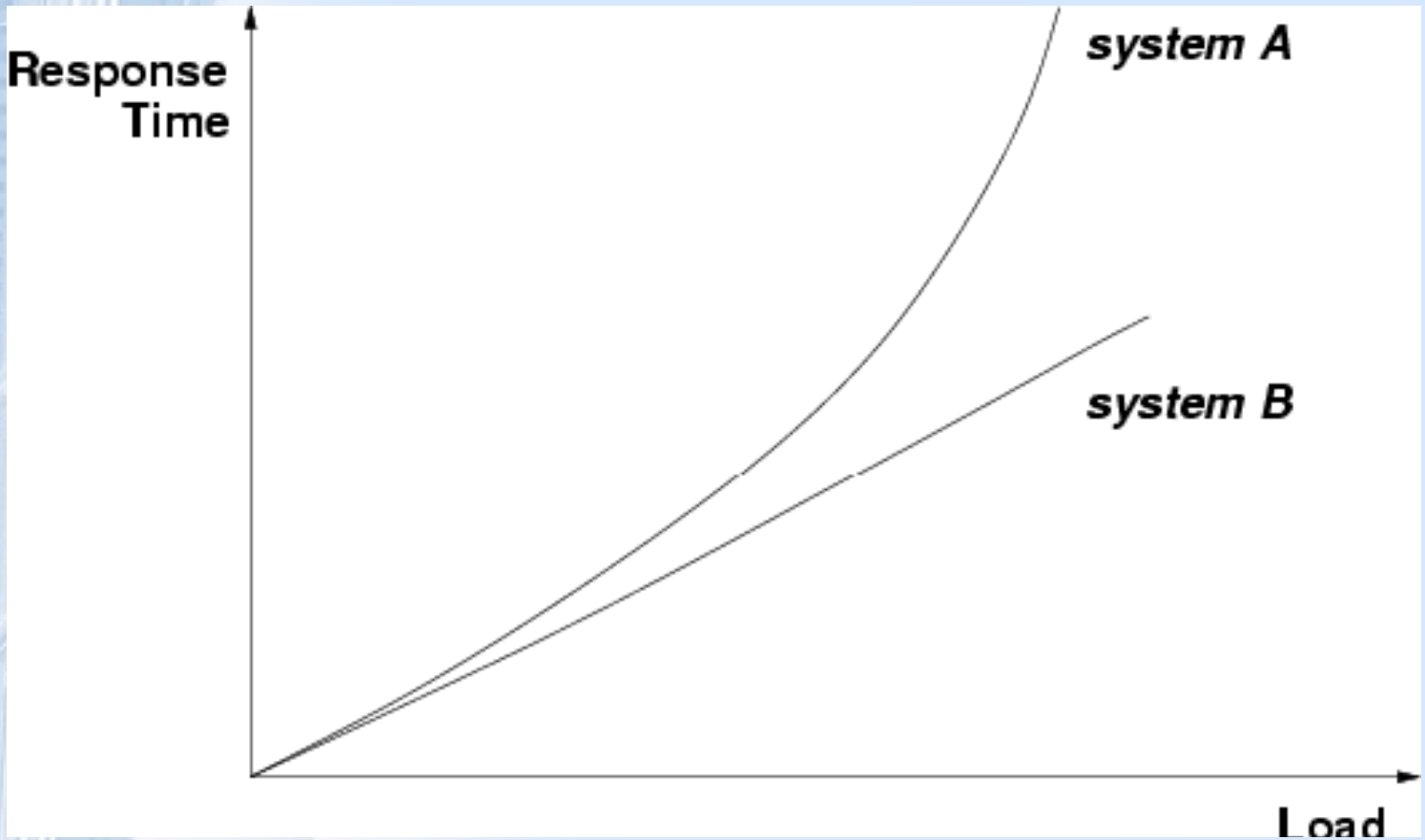
No admission control

with admission control

Reliability

The reliability of a system is the probability that it functions properly and continuously over a fixed period of time.

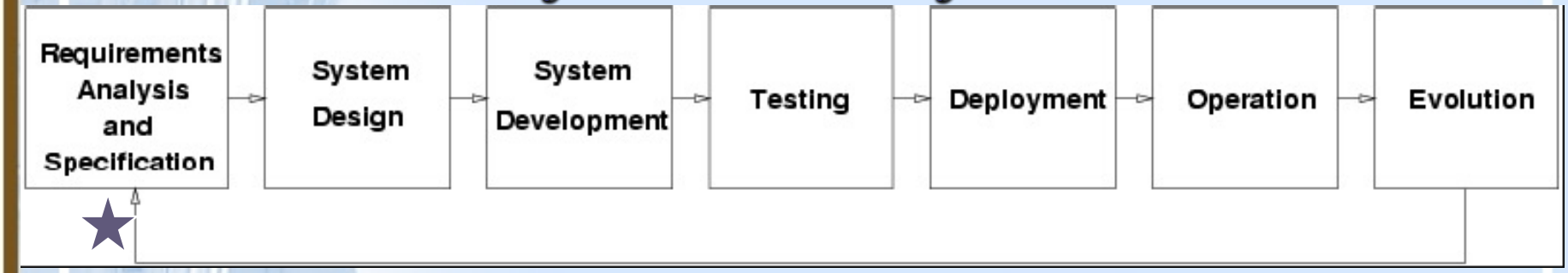
Scalability



Extensibility

- Property of a system to constantly evolve to meet functional and performance requirements.
  - Autonomic computing, self-managing systems, self-healing systems.

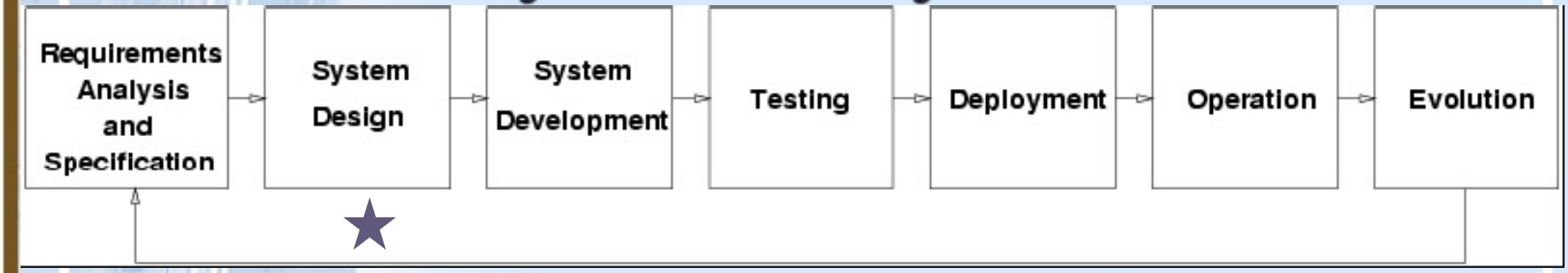
System Life cycle



•Functional requirements: what the system has to do and on what type of platforms.

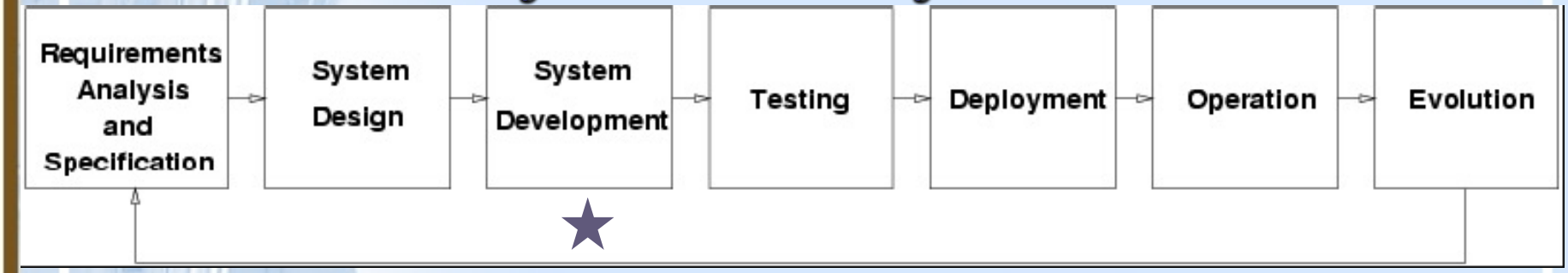
•Non-functional requirements: how well the system has to accomplish its functions. Service Level Agreements (SLA) are established.
In many cases, non-functional requirements have been neglected or considered only at system test time!

System Life cycle



How will the requirements be met?
- System architecture
- System broken down into components
- Major data structures, files, and databases are designed.
- Interfaces between components are specified

System Life cycle

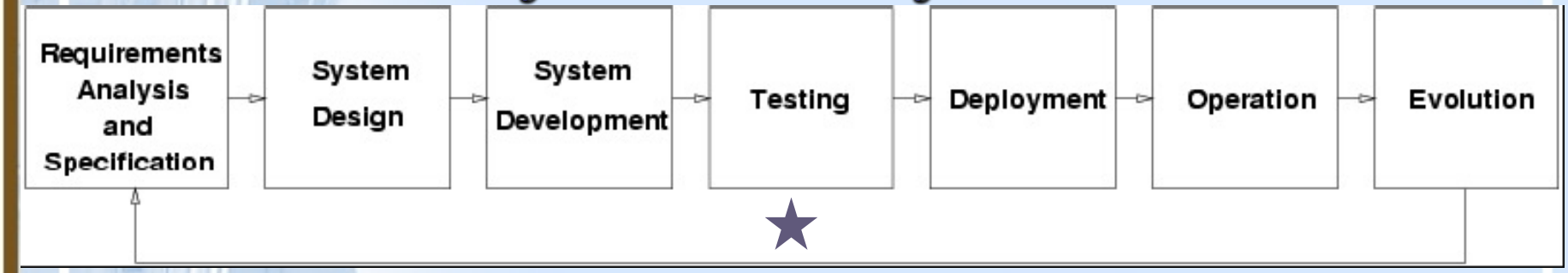


Components are implemented.
- some are new
- some are re-used
- some are adapted

Components are interconnected to form a system

Components should be instrumented as they are built

System Life cycle

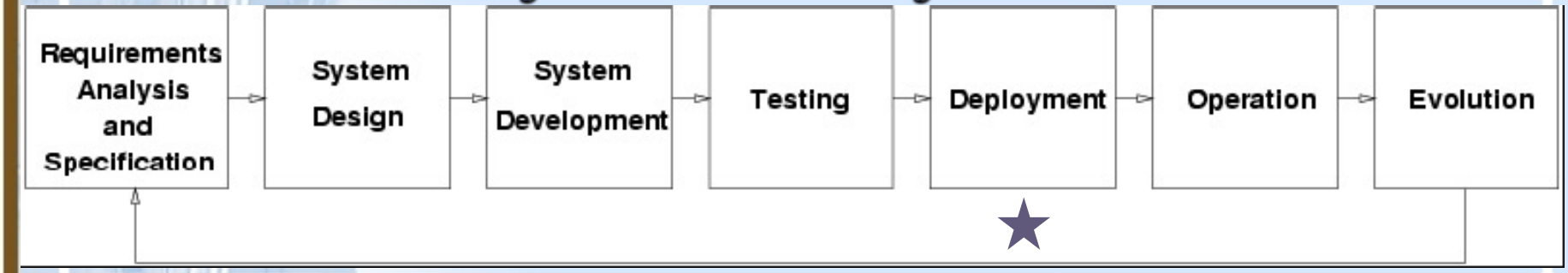


Concurrent with system development, as components become available (unit testing)

Integrated tests are carried out when the entire system is ready.

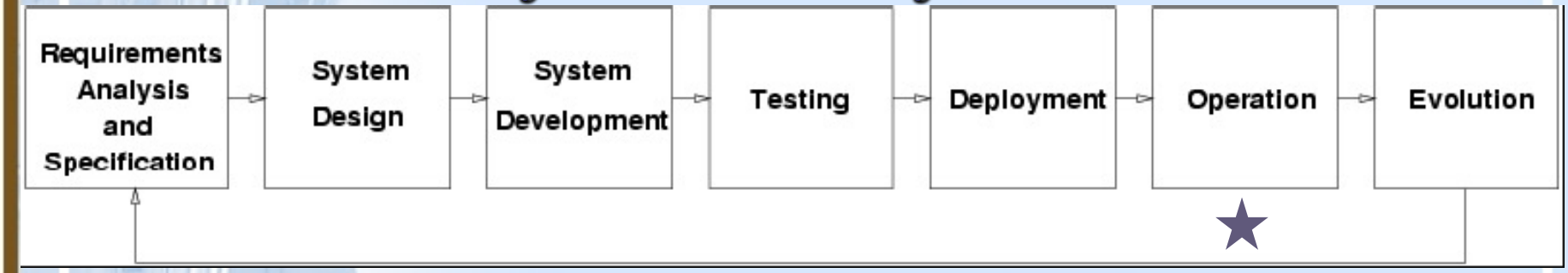
Often, more time is spent in testing functional requirements than in testing non-functional requirements.

System Life cycle



- Configuration parameters have to be set in order to meet the SLAs.
  - e.g., TCP parameters, database poolsize, maximum number of threads, etc.

System Life cycle

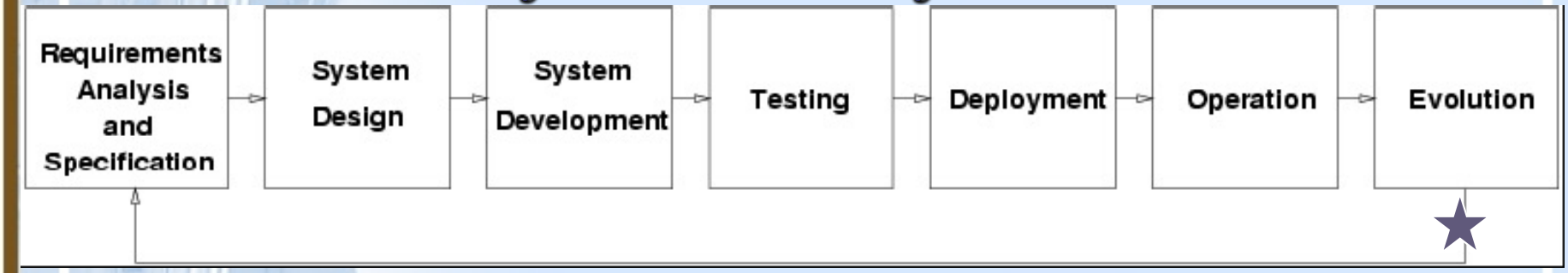


Constant monitoring to check if the system is meeting demands:

- workload (peak periods, unusual patterns)
- external metrics (user-perceived)
- internal metrics (help to detect bottlenecks and to fine tune the system)
- availability (external and internal)

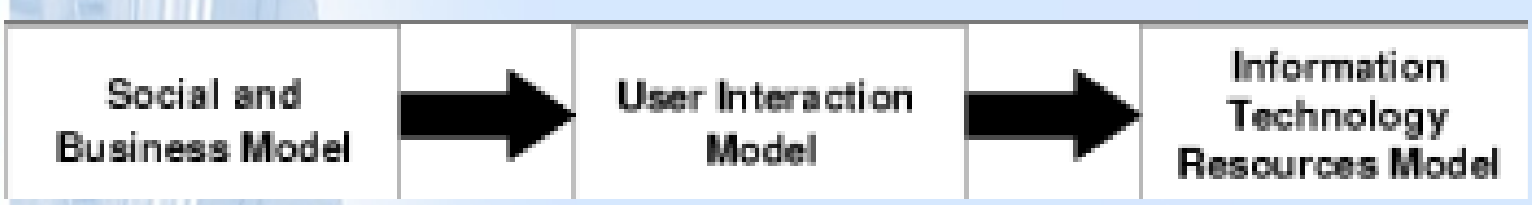
May need to dynamically adjust configuration parameters

System Life cycle



- Systems may need to evolve to cope with new laws and Regulations (e.g., HIPPA)
- Systems may need to evolve to provide new functions (e.g., sale of downloadable MP3 music in addition to CDs)
- How are the IT resources going to cope with evolution in terms of SLAs?

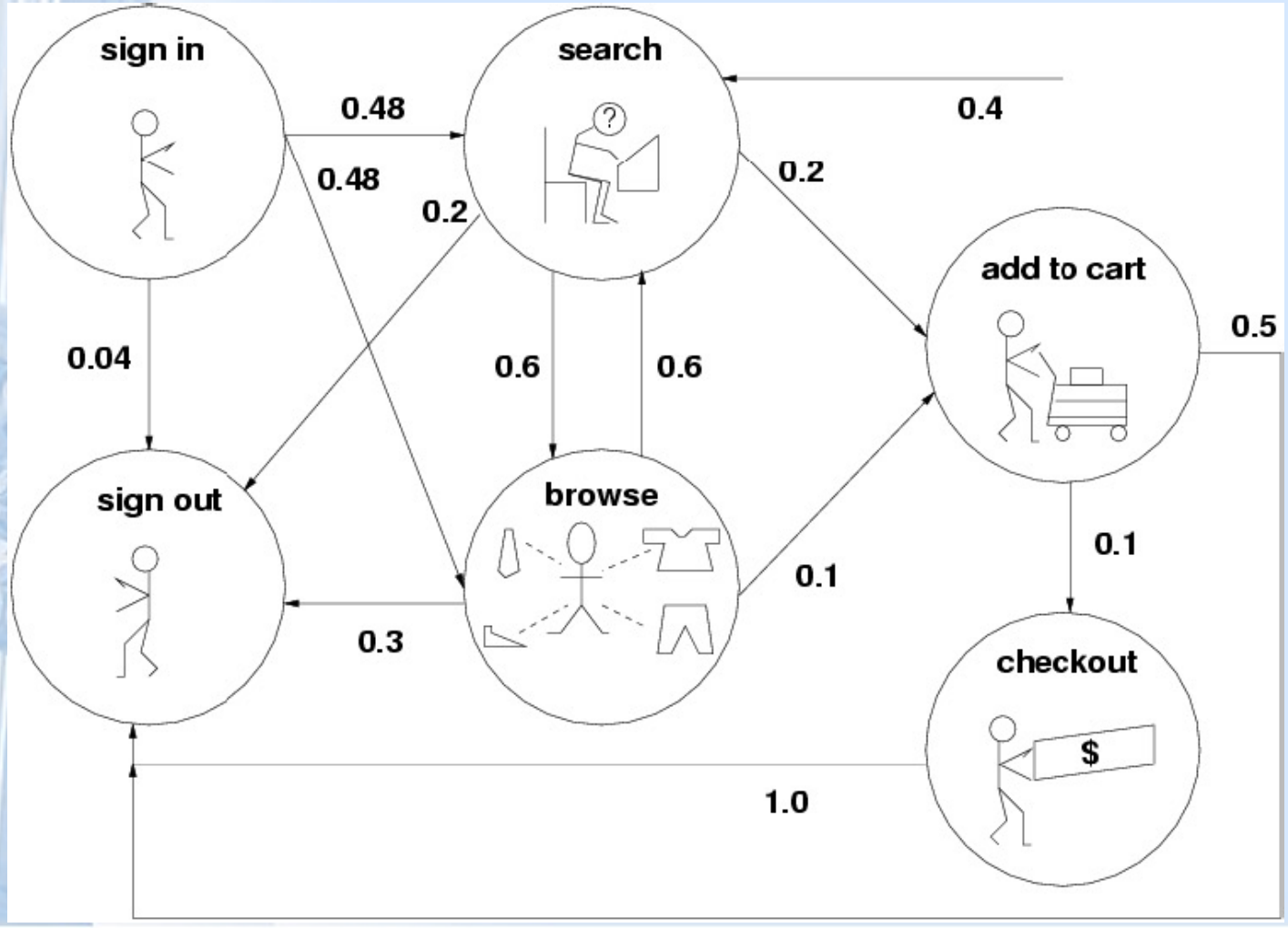
A Reference Model for IT Systems



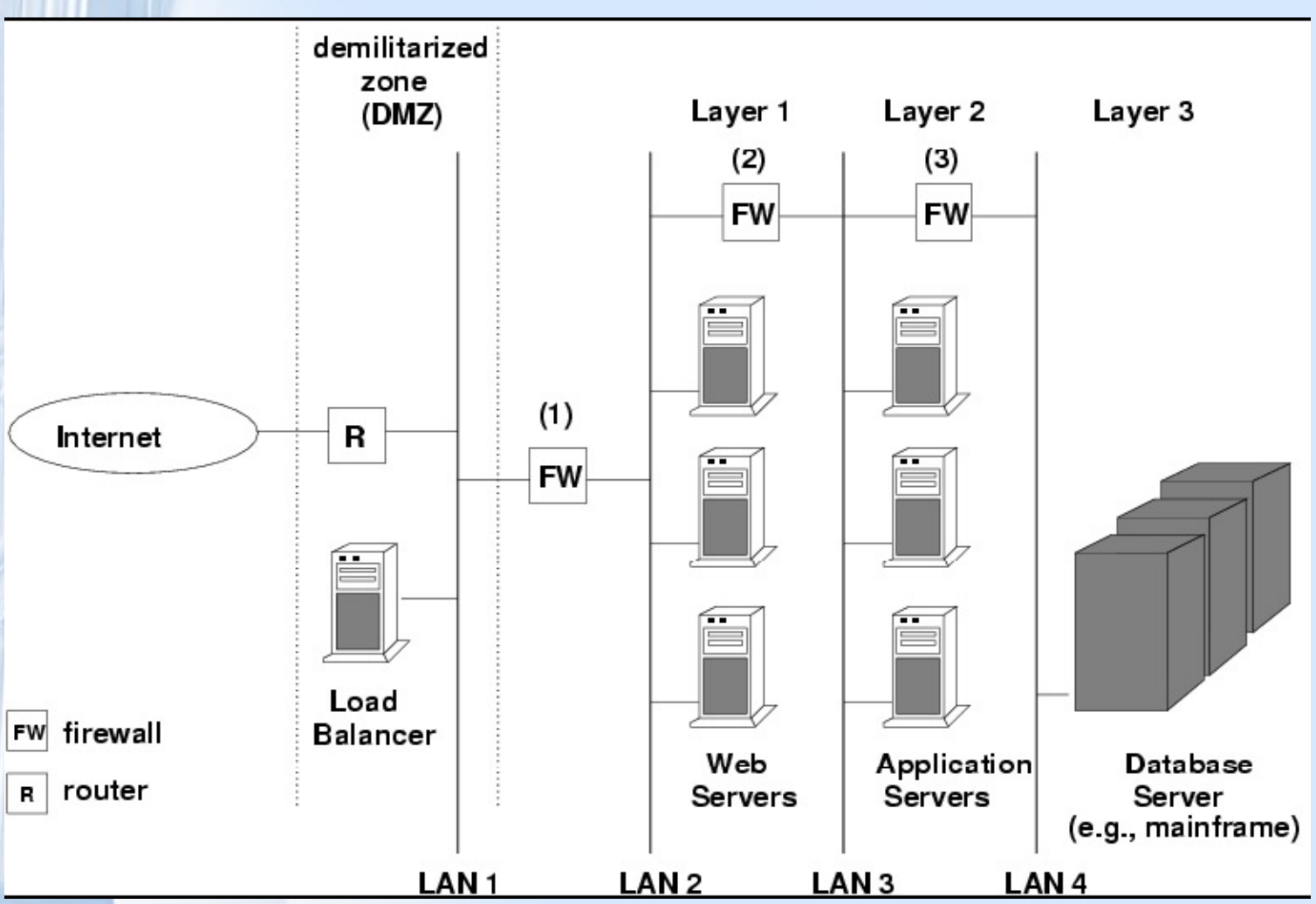
Business Model:
- number of branches
- number and location of ATMs
- number of accounts of each type
-business evolution plans (e.g., mergers)

Social Model
- privacy policy
- accessibility policy

User Model:
Customer Behavior Model Graph



IT Infrastructure: Example



Conclusions

- In this chapter we introduced several properties and metrics used to assess the quality of IT systems.
- We also discussed the various phases of the life cycle of a computer system and showed the importance of addressing QoS issues should be as early as during the design stage.