IBD – Intergiciels et Bases de Données

Introduction

Fabien Gaud, fabien.gaud@inrialpes.fr

http://www-ufrima.imag.fr/ ⇨ Placard électronique ⇨ M1 Info ⇨ IBD

Objectives

- Introduction to distributed systems and middleware
- Conceptual and practical aspects of distributed systems and middleware
- Illustration through current distributed systems, e.g. web systems, database systems

Professors

- Middleware and distributed systems
  - Fabien Gaud (Fabien.Gaud@inrialpes.fr)
  - Fabienne Boyer (Fabienne.Boyer@inrialpes.fr)
- Database systems
  - Marie-Christine Fauvet (Marie-Christine.Fauvet@imag.fr)
  - Goran Frehse (Goran.Frehse@imag.fr)

Organization

Week date	Wednesday, 13:30 – 15:00, Lecture, room F018	Wednesday, 15:15 – 18:30, Practical work), rooms F202/F204	Friday, 13:30 – 15:00, Lecture, room F022
22 Sep. 2008	Lecture 1 on Middleware		Lecture 1 on DB
29 Sep. 2008	Lecture 2 on Middleware	Practical work 1 on DB	Lecture 2 on DB
6 Oct. 2008	Lecture 3 on Middleware	Practical work 1 on Middleware	Lecture 3 on DB
13 Oct. 2008	Lecture 4 on Middleware	Practical work 2 on DB	Lecture 4 on DB
20 Oct. 2008	Lecture 5 on Middleware	Practical work 2 on Middleware	Lecture 5 on DB
27 Oct. 2008			
3 Nov. 2008	Lecture 6 on Middleware	Practical work 3 on DB	Lecture 6 on DB
10 Nov. 2008	Lecture 7 on Middleware	Practical work 3 on Middleware	Lecture 7 on DB
17 Nov. 2008	Lecture 8 on Middleware	Practical work 4 on Middleware	Lecture 8 on DB Practical work 4 on DB
24 Nov. 2008		Project	
1 Dec. 2008	Project	Project	Project
8 Dec. 2008	Project demonstration	Project demonstration	

Organization

- Conceptual and practical aspects of middleware, distributed systems, and database systems
  - Lectures: 24h
  - Practical work: 36h
- Prerequisites
  - Java programming
  - Structured query language (SQL)
- Perspectives
  - Master 1 Info, Semester 2
    - sp. SRIL (*Systèmes, Réseaux et Ingénierie Logiciels*)
  - Master 2 Professional
    - M2P – GI
  - Master 2 Research
    - M2R – Distributed Systems
    - M2R – Database systems
    - M2R – Information systems

Web site and evaluation

- Web site
  - http://www-ufrima.imag.fr/
    - ⇨ Intranet
    - ⇨ Services pédagogiques
    - ⇨ Placard électronique
    - ⇨ M1 Info
    - ⇨ IBD
- Evaluation
  - Practical work
  - Mid-term evaluation
    - A project on middleware, distributed systems and database systems
    - Demonstration of project results
    - Evaluation of the demonstration
  - Final evaluation
    - Final exam

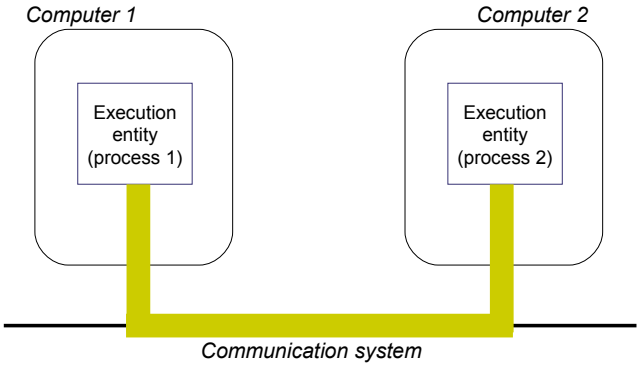
Outline of lectures and practical work on middleware

- Lectures
  - Introduction to distributed systems and middleware
  - Socket-based distributed systems
  - RMI-based distributed systems
  - Servlet-based distributed systems
  - Introduction to multi-tier distributed Internet services
- Practical work
  - Programming distributed systems with Sockets
  - Programming distributed systems with RMI
  - Programming distributed systems with Servlets
  - Project on multi-tier Internet services

Outline

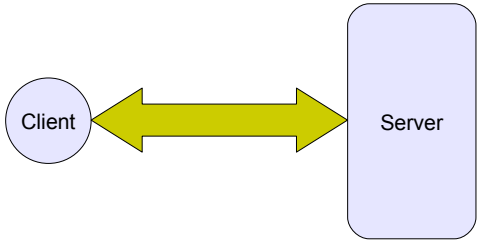
- Objectives and organization
  - Objectives
  - Chronological organization
  - Prerequisites and perspectives
  - Evaluation details
  - Web site information
  - Professors contact
- **What is a distributed system**
- What is a middleware
- Conclusion

What is a distributed system



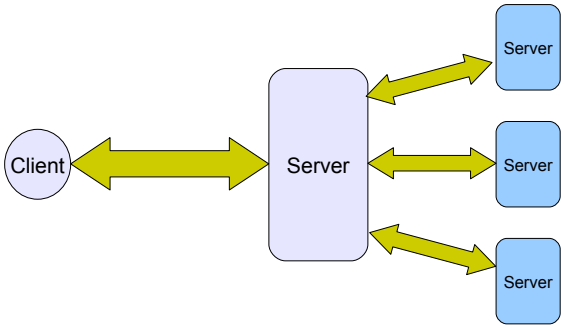
Example of distributed systems

- Client - Server



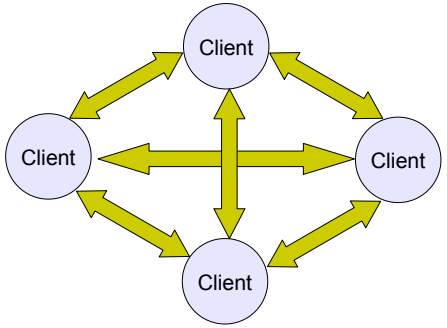
Example of distributed systems

- Multi-Tier



Example of distributed systems

- Peer to peer



Distributed systems

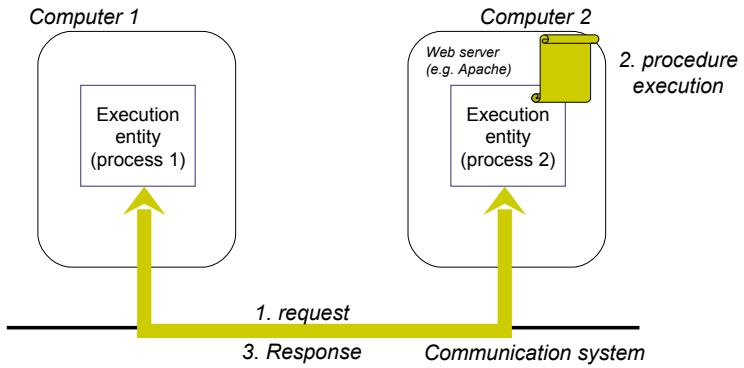
- Advantages
  - Storage/Computation capacity
  - Scalability
  - Fault tolerance
- Points to consider
  - Security
  - Failure
  - Synchronization
  - Heterogeneity

Communication mechanisms in a distributed system

- Direct
  - Program to program
    - E.g. remote procedure call
  - Program to database
    - E.g. distributed transaction processing
- Indirect
  - Message queuing

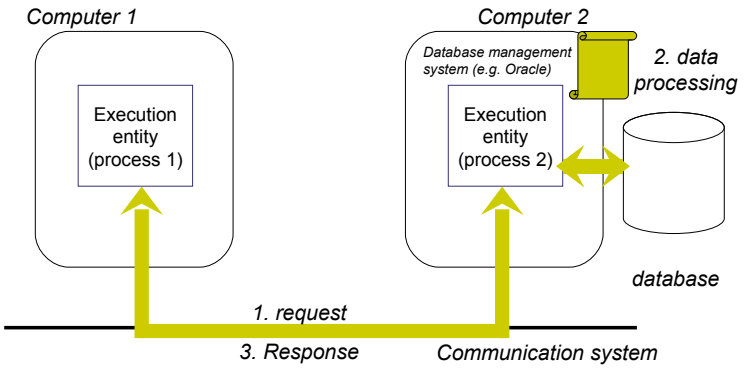
Communication mechanisms in a distributed system

- Remote procedure call (e.g. a web application)



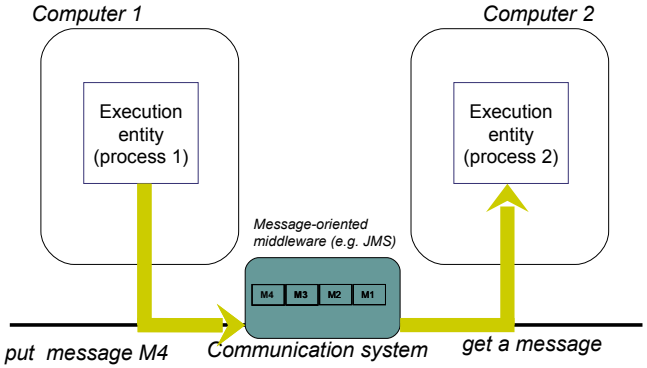
Communication mechanisms in a distributed system

- Distributed transaction processing (e.g. a database server)



Communication mechanisms in a distributed system

- Message queuing (e.g. a chat system)



Communication mechanisms in distributed systems - perspectives

- Direct
  - Program to program
    - E.g. remote procedure call
  - Program to database
    - E.g. distributed transaction processing
- Indirect
  - Message queuing

M1 Info – IBD – "I" Part

M1 Info – IBD – "BD" Part

M2P GI – Dist. Sys.
M2R Info – sp. SAR – Dist. Sys.

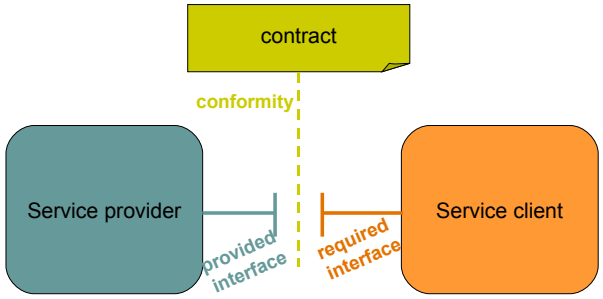
Outline

- Objectives and organization
- What is a distributed system
  - Communication mechanisms in distributed systems
  - **Services and interfaces in computing systems**
  - Client/server architecture
- What is a middleware
- Conclusion

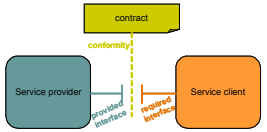
Services and interfaces in a computing system

- Service definition
  - A computing system is a set of (hardware and software) components
  - A component provides a service
  - "A service is a contractually defined behavior that can be implemented and provided by any component for use by component, based solely on the contract",
    Bieber el. al., Service oriented programming, http://www.openwings.org/
- Interface definition
  - A service is accessible via one or several interfaces
  - An interface defines the interaction between a service provider and its client

Interfaces (1/2)

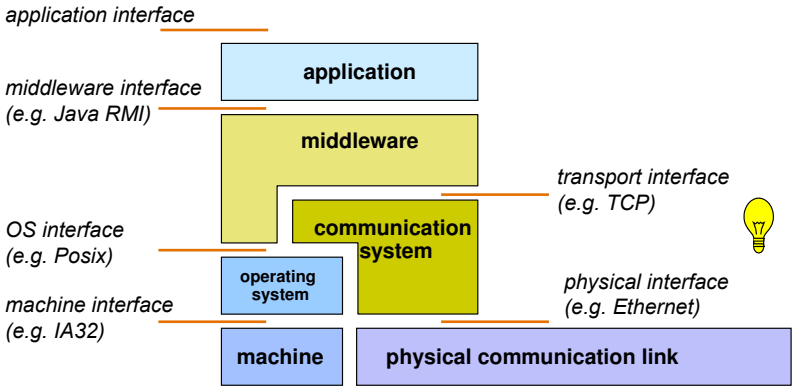


Interfaces (2/2)



- A service relies on two interfaces
  - Required interface (from the service client point of view)
  - Provided interface (from service provider point of view)
- Contract
  - The contract specifies the conformity between the provided and required interfaces
  - The service client and the service provider are considered as black-boxes; they might be replaced by other implementations as long as the contract is respected
- The contract may specify aspects that are not related to the interfaces
  - Non-functional properties related to QoS requirements

Examples of important interfaces in computing systems

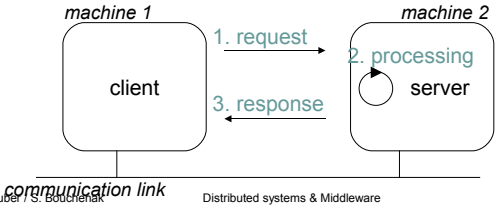


Outline

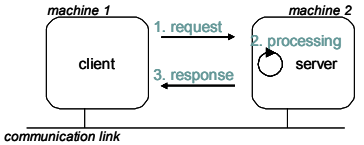
- Objectives and organization
- What is a distributed system
  - Communication mechanisms in distributed systems
  - Services and interfaces in computing systems
  - **Client/server architecture**
- What is a middleware
- Conclusion

Client/server architecture (1)

- Definitions
  - The client/server architecture is a general interaction model
  - The server provides a service
  - The client requests that service
  - The client and the server are usually (but not necessarily) hosted by two distinct machines
  - Examples of protocols based on the client/server architecture: RPC, Java RMI, Web Services, etc.



Client/server architecture (2)



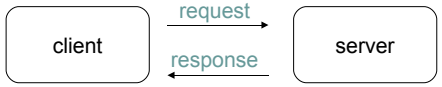
- Request message:
  - Sent by the client to the server
  - Specifies the requested service (a server may provide several services)
  - Contains parameters of the requested service
- Response message:
  - Sent by the server to the client
  - Results of service execution, or error message
- Synchronous communication between the server and the client:
  - When the client sends a request, it waits (it is blocked) until the server replies to its request

Client/server architecture (3)

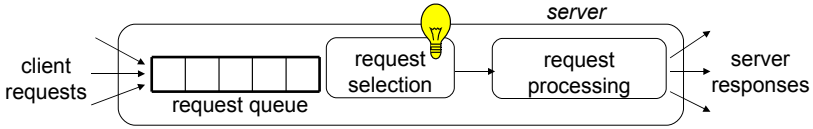
- Advantages of the client/server architecture
  - Structuring
    - Separation between the interface of a service and the implementation of that service
    - Based on this separation, the client and server implementations can be modified as long as the interface is kept unchanged
  - Protection/security
    - The client and server run in different protection domains
  - Resource management
    - A server may be shared by several clients

Client/server architecture (4)

- A server shared by several clients
  - The client point of view



  - The server point of view
    - Selecting a request among client requests
    - Request processing model (sequential or parallel)
    - Managing client sessions



Client/server architecture (5)

- Request processing model (resource management)
  - The client and server are executed by two distinct processes (asynchronous call)
  - The client waits until it receives a response to its request
  - Several requests may be processed concurrently by the server
    - real parallelism (e.g. multiprocessors, I/O)
    - pseudo-parallelism
  - Concurrency may take the form of:
    - multiple processes, or
    - multiple threads, or
    - Finite State Machine (FSM)

Threads & processes

- Basic unit of execution
- Threads / Processes are scheduled by the operating system
- On multiprocessor systems, multiple threads can be executed at the same time
- Threads from the same process can share memory

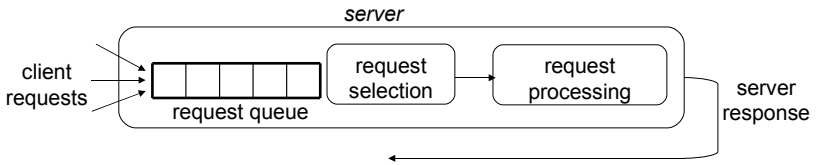
Threads & processes

- Threads
  - Advantages
    - Lighter than processes
    - Memory sharing
  - Drawbacks
    - Be careful with memory mutual accesses
- Processes
  - Advantages
    - Memory isolation
  - Drawbacks
    - Heavy

Client/server architecture (6)

- Server resource management – A unique thread

```
while (true) {
        receive(client_id,message);
        extract(message, service_id, params);
        results = do_service(service_id, params);
        send(client_id, results);
}
```

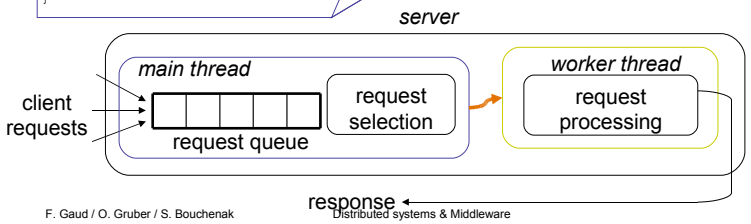


Client/server architecture (7)

- Server resource management – Multiple threads

```
while (true) {
   receive(client_id,message);
   extract(message, service_id,
      params);
   thr = create_thread(client_id,
      service_id,params);
}
```

```
Program executed by thread thr:
   results = do_service(
      service_id, params);
   send(client_id, results);
   exit
```



Client/server architecture (8)

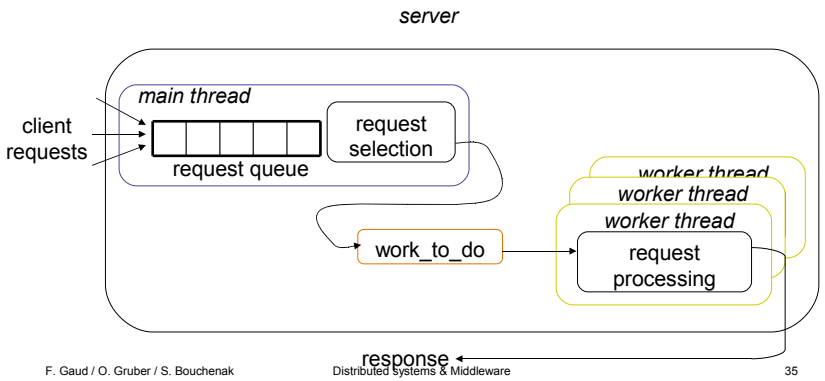
- Server resource management – A pool of thread

```
while (true) {
   receive(client_id,message);
   extract(message, service_id,
      params);
   work_to_do.put(client_id,
      service_id,params);
}
```

```
Pool of threads:
while (true) {
   work_to_do.get(
      client_id, service_id,
      params);
   results = do_service(
      service_id, params);
   send(client_id, results);
}
```

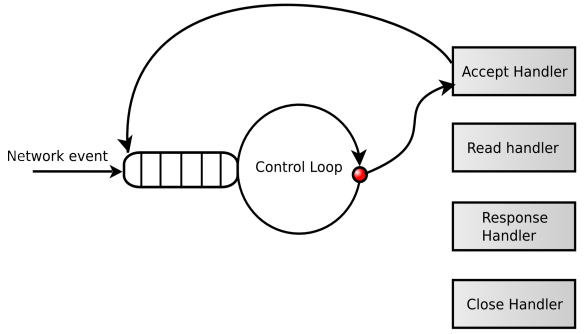

Client/server architecture (9)

- Server resource management – A pool of threads



Client/server architecture (9)

- Server resource management – A FSM



Client/server architecture (10)

- Application of the client/server architecture
  - With low level operations
    - Using functions of the communication system
    - Example: Sockets
      - TCP, connected mode
      - UDP, unconnected mode

Lecture 2

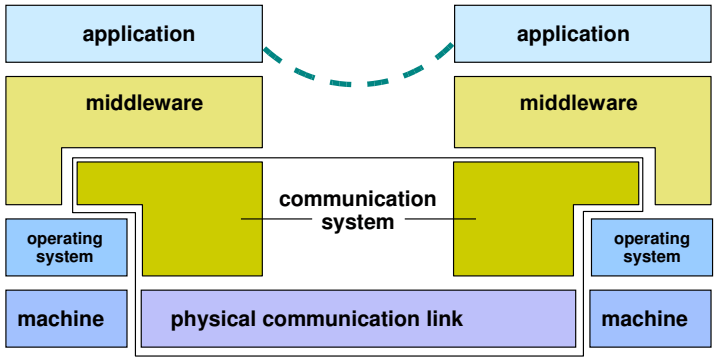
  - With high level operations
    - Using a middleware
    - Example: RMI in object-oriented middleware
      - Remote method invocation

Lecture 3

Outline

- Objectives and organization
- What is a distributed system
  - Communication mechanisms in distributed systems
  - Services and interfaces in computing systems
  - Client/server architecture
- **What is a middleware**
- Conclusion

What is a middleware



Functions of a middleware

- A middleware has mainly four functions
  - Make distribution as invisible (transparent) as possible
  - Provide a homogeneous view of underlying heterogeneous hardware and software systems
  - Provide services of common use for distributed systems
  - Provide a high-level interface or API (*Applications Programming Interface*) for programming distributed applications

Middleware for distributed systems

- Middleware aims at simplifying programming distributed systems
  - Implementation, evolution and reuse of applications code
  - Inter-platform portability of applications
  - Interoperability between heterogeneous applications

Middleware layers

- OS
- JVM
- RMI
- Servlet / JSP
- Sun J2EE / EJB

Outline

- Objectives and organization
- What is a distributed system
- What is a middleware
  - What is a middleware
  - Functions of a middleware
  - Middleware for distributed systems
  - Examples of middleware solutions

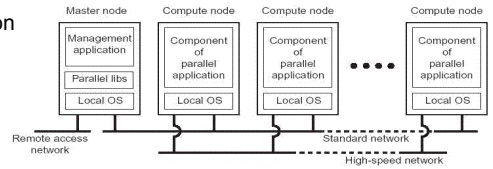
4. **Conclusion**

Typical distributed systems

- Distributed computing systems
- Distributed information systems
- Distributed pervasive systems

Distributed computing systems

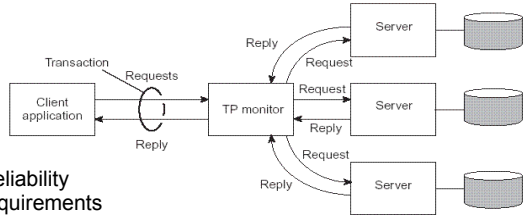
- Objective
  - Distributed systems configured for high performance computing
- Cluster computing
  - A group of high-end systems connected through a LAN
  - Homogeneous, i.e. same OS, hardware
  - Single managing node
- Grid computing
  - Heterogeneity
  - Geographical dispersion
- Applications
  - Video streaming
  - Web services
  - Scientific computing



M. van Steen, Lecture on Distributed Systems, Chapter 1, http://www.cs.vu.nl/~steen/

Distributed information systems

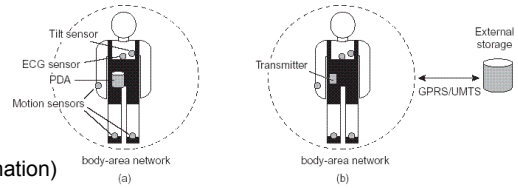
- Objective
  - Providing consistent access to (shared) data that can be distributed and accessed concurrenly
- Observation
  - Transactions
  - ACID properties
- Applications
  - Data access with reliability and consistency requirements



M. van Steen, Lecture on Distributed Systems, Chapter 1, http://www.cs.vu.nl/~steen/

Distributed pervasive systems

- Objective
  - Providing consistent access to (shared) data that can be distributed and accessed concurrenly
- Observation
  - Contextual change
  - Ad-hoc composition
- Applications
  - Domotics (home automation)



M. van Steen, Lecture on Distributed Systems, Chapter 1, http://www.cs.vu.nl/~steen/

Outline

- Objectives and organization
- What is a distributed system
- What is a middleware
4. Conclusion
   - Types of distributed systems
   - **Incoming lectures**
   - References
   - Web site visit

Incoming lectures and practical work on middleware

- Lectures
  - *Introduction to distributed systems and middleware*
  - Socket-based distributed systems
  - RMI -based distributed systems
  - Servlet-based distributed systems
  - Introduction to multi-tier distributed Internet services
- Practical work
  - Programming distributed systems with Sockets
  - Programming distributed systems with RMI
  - Programming distributed systems with Servlets
  - Project on multi-tier Internet services

References

- Chris Britton, Peter Bye. *IT Architectures and Middleware: Strategies for Building Large, Integrated Systems (2nd Edition)*. Addison-Wesley, 2004.
- George Coulouris, Jean Dollimore, Tim Kindberg. *Distributed Systems: Concepts and Design (4th Edition)*. Addison Wesley, 2005.
- Arno Puder, Kay Römer, Frank Pilhofer. *Distributed Systems Architecture: A Middleware Approach*. Morgan Kaufmann, 2005.
- Andrew S. Tanenbaum, Maarten van Steen. *Distributed Systems: Principles and Paradigms (2nd Edition)*. Prentice Hall, 2006.

- This lecture is mostly based on lectures given by Sara Bouchenak, http://sardes.inrialpes.fr/~bouchena/

Web site visit…

- http://www-ufrima.imag.fr/
  - ⇨ Intranet
  - ⇨ Services pédagogiques
  - ⇨ Placard électronique
  - ⇨ M1 Info
  - ⇨ IBD