Introduction to Distributed Systems

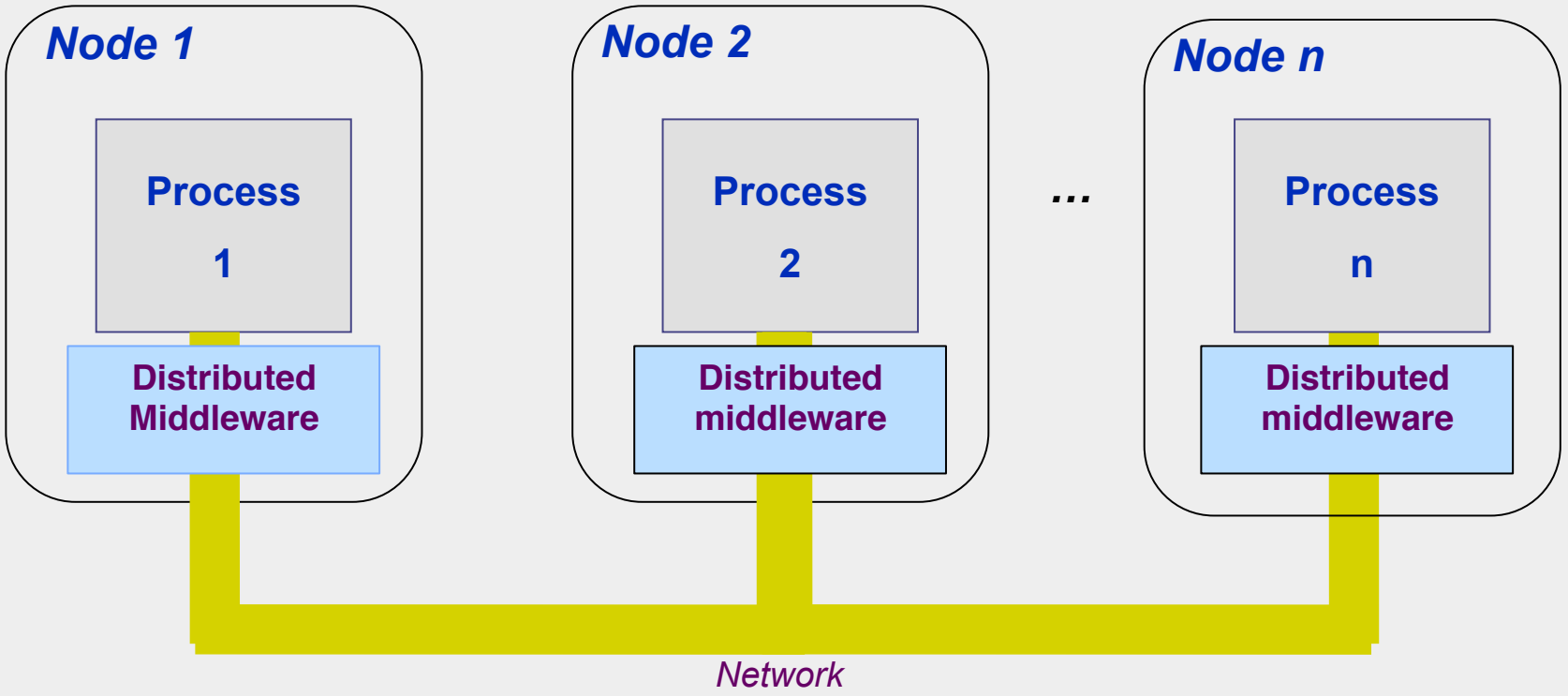
Polytech/INFO 4, 2022-2023

Fabienne Boyer
UFR IM2AG, LIG, Université Grenoble Alpes
Fabienne.Boyer@imag.fr

What is a distributed system?

- **A distributed system is made of**
  - A set of processes running in separate address spaces (*either on the same machine or on distributed ones*)
  - Communicating through a network
    - Classically components communicate through a *distributed middleware* providing communication facilities
  - Collaborating to a common goal

What is a distributed system?



Examples of distributed systems

- **Information sharing**
  - Wikipedia
- **Business**
  - E-commerce
- **Collecticiels**
  - Workflow (BPM)
  - Audio/visio-conferences
- **Distributed games**
- **Streaming servers**
- **Naming server**
  - **DNS**
- **Network file servers**
  - **AFS, NFS**
- **…**

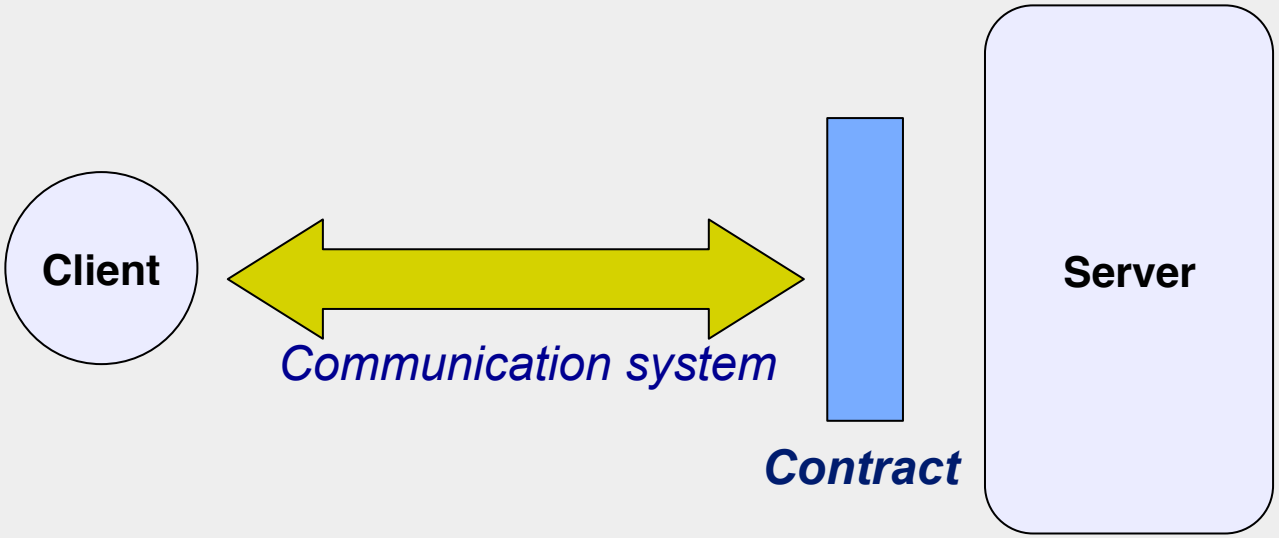
Objectives of the course

- **Study well-known distributed middlewares**
  - Message-based/Stream-based (Sockets)
  - Event-driven (Java/NIO)
  - Object-based (Java/RMI)
  - Web-based (Servlets/HTTP)

- **Program Client-Server distributed applications on top of these middlewares**
  - Client-Server - the most common architecture for distributed systems

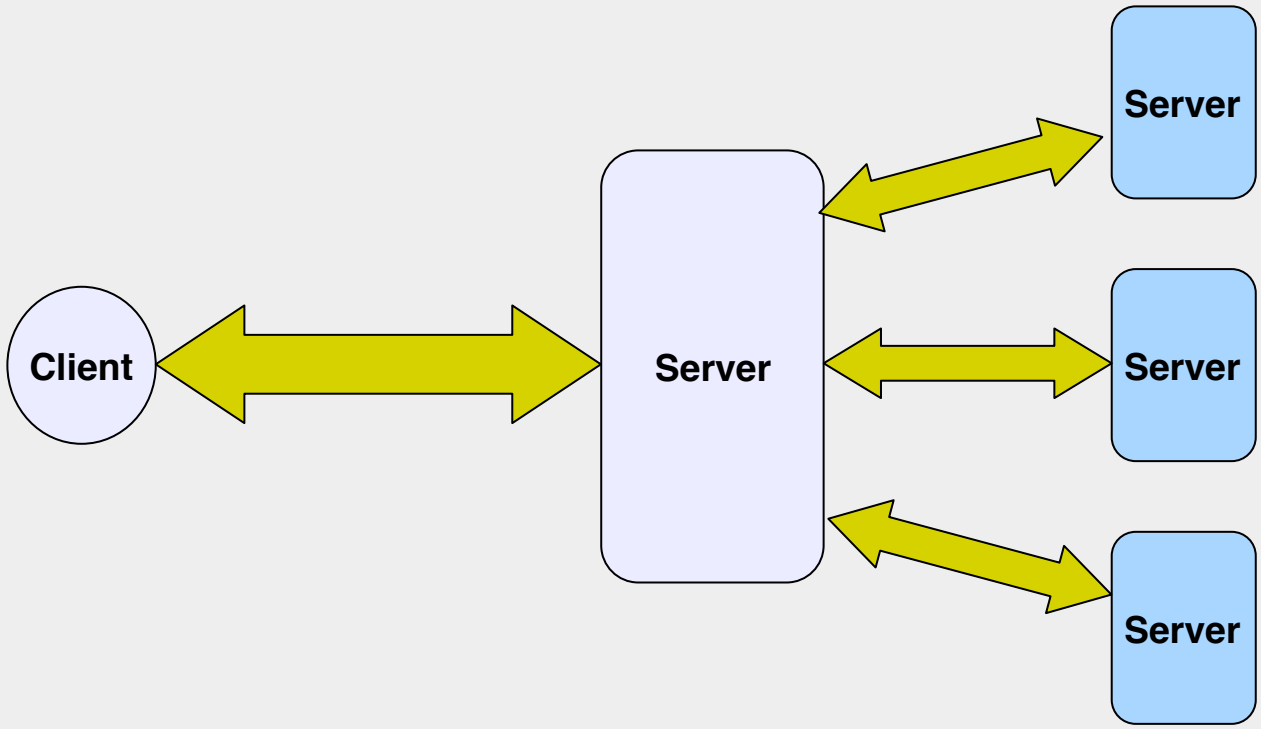
Basic Client-Server architecture

- A client uses some services provided by a server
- The services conform to a contract (usually specified as an *interface*)



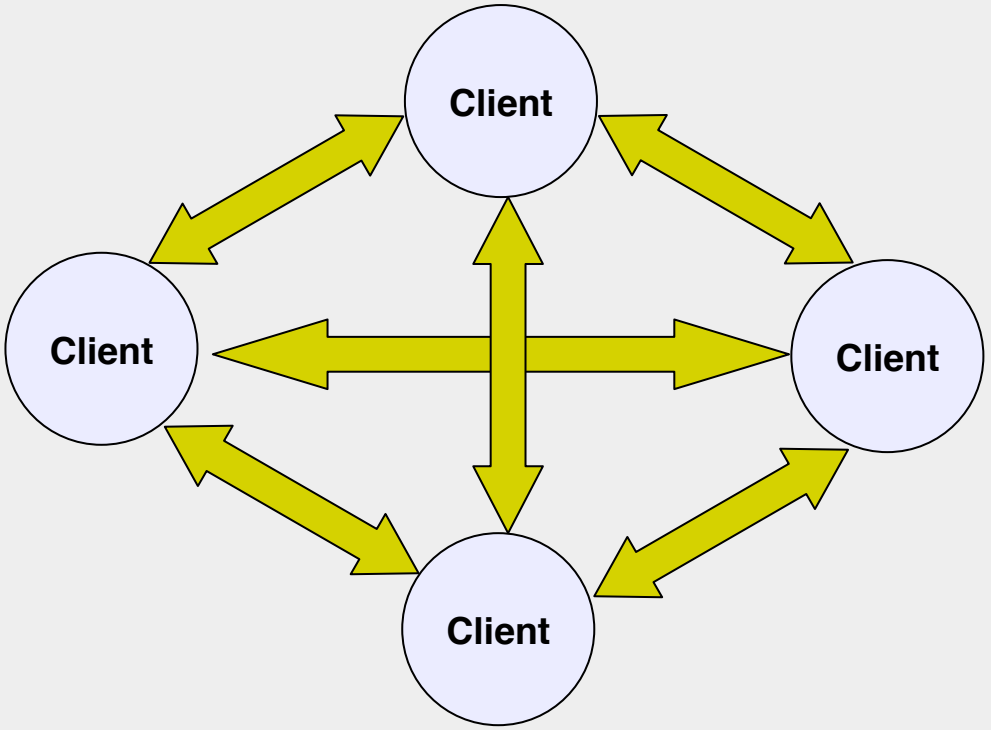
Multi-tiers architecture

- Distribute or replicate client requests on several servers
- Delegate client requests to specialized servers



Peer-to-peer architecture

- Decentralized architecture (no central server)
- Software components play the role of both client and server



Properties of distributed systems

Some expected properties

- **Availability:** the system aims to be usable 24/7
- **Reliability:** failures should not impact the system's availability
- **Scalability:** ability to support load variations
- **Manageability:** easiness of deploying, upgrading, maintaining the system over time
- **Security:** ability to resist to attacks
- …

Why it is difficult to ensure these properties

- **Network asynchronism:** no bounded delay for communication
- **Dynamicity:** components may come and leave at any moment
- **Heterogeneity:** components may be programmed in different languages
- **Size:** the number of components may be high
- …

REST Client-Server Architectures

- **Client-server applications are RESTful if they follow the REST rules**
  - Services follow a uniform interface (PUT, GET, POST, DELETE)
  - Servers are stateless: they do not keep any state between successive client requests
  - Any request brings the necessary data (e.g., a credit operation brings the accont number and the amount)
  - …
- **Favor reliability, scalability, manageability**
  - A server can easily be stopped-restarted (no state to manage)
  - A request can be served by any server

Planning

Séances	Focus	Practical work (**BINOMES**)
1, 2	Message-based / Stream-based distributed systems (Sockets)	File server over TCP/IP
3,4,5	Event-based distributed systems (Java/NIO)	File server over NIO
6,7	Object-based distributed systems (Java / RMI)	Chat server on top of Java/RMI
8	Web-based systems (HTTP/Servlets)	Servlet application on top of Apache HTTP server
--	Asynchronous, object-based distributed systems (MOM)	TO SEE
9, 10	Object-based distributed system (internals)	Programming your own Web server