

Distributed Systems Basics

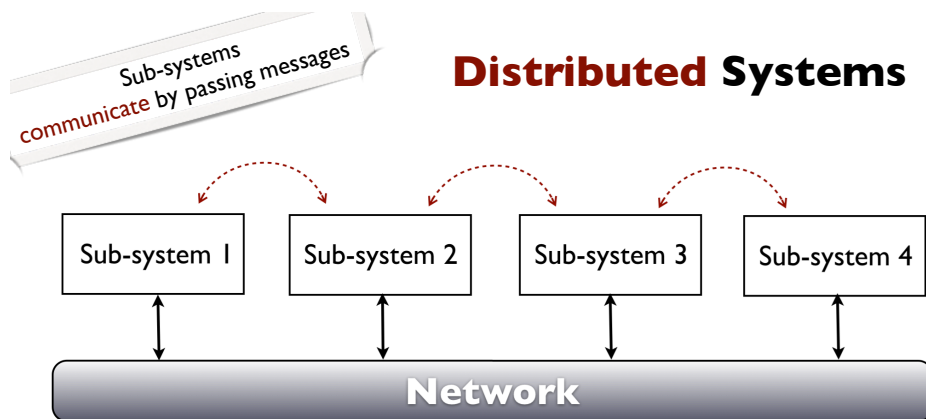
Dr. Petru Florin Mihancea

V20180301

1

Definitions and Challenges

Dr. Petru Florin Mihancea



Distributed software systems

a software **system** where the information processing is distributed over **several computers**

[Sommerville, Software Engineering, 2006]

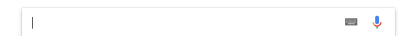
Dr. Petru Florin Mihancea



[<https://forrestercomputing.wikispaces.com/World+Wide+Web>]

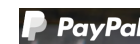
Examples

Google



Căutare Google Mă simt norocos

Google ofertă în: English magyar Deutsch



NETFLIX



G Suite

...

Dr. Petru Florin Mihancea

Challenges

Concurrency

Heterogeneity

Security

Scalability

Failure handling

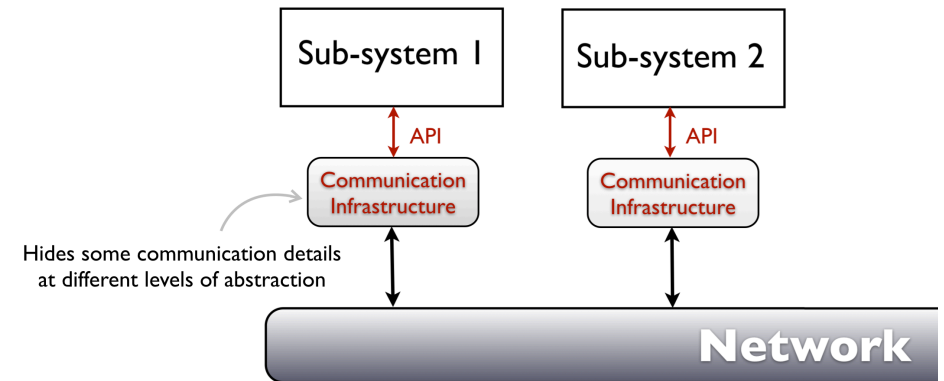
Openness

Transparency

QoS

Dr. Petru Florin Mihaies

Our Goal

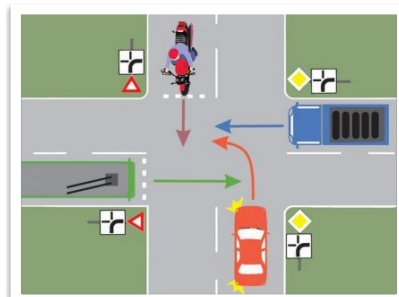


Implementing the communication based on the **interfaces (APIs)** of several existing communication infrastructures

Dr. Petru Florin Mihaies

Concurrency

the **same shared resource** may be accessed by **different users** at the **same time**



[<http://www.promotor.ro/>]

we must ensure correct operation in this concurrent environment

Dr. Petru Florin Mihaies

Heterogeneity

Differences as

- networks
- computer hardware
- operating systems
- programming languages
- developers

must be addressed!



<http://www.hrreview.co.uk/>

Protocols

Agreed message formats and their sequencing
Internet protocols hides network diversity

Middleware

Software infrastructure (over the primary network service) offering high level communication services for applications (hiding some heterogeneity aspects, etc.)

Dr. Petru Florin Mihaies

Security

Referring to

- confidentiality
- integrity
- availability



Message encryption

Concealing the content

Authentication

The communicating entities are the right ones

Dr. Petru Florin Mihances

2

Architectural Elements of Distributed Systems

Dr. Petru Florin Mihances

A

Communicating Entities

Nodes

usually processes

Higher level abstractions

objects
components
web services

Dr. Petru Florin Mihances

B

Kinds of Communication

Interprocess

lowest level

Socket programming (access to TCP/IP infrastructure)

Remote invocation

Request-reply protocols

small improvements over interprocess

Remote procedure call (RPC)

Remote method invocation (RMI)

Indirect communication

Publish-subscribe

Message queues

Distributed shared memory, etc.

Dr. Petru Florin Mihances

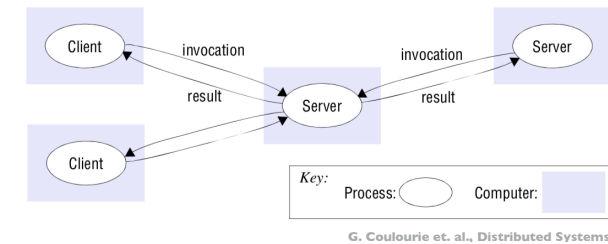
3

Distributed Architectural Styles

Dr. Petru Florin Mihances

A

Client-Server

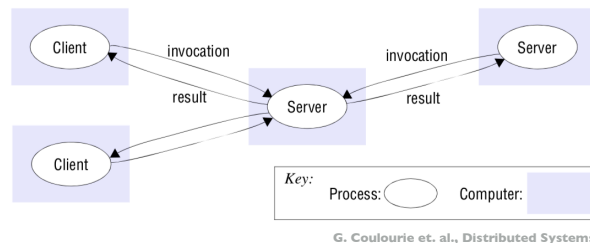


Servers offer services
Clients use these services

Dr. Petru Florin Mihances

A

Client-Server



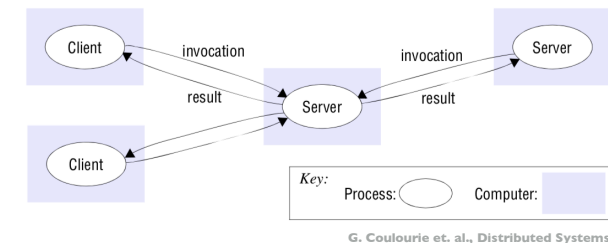
Clients know

- names of servers
- what services each server provides

Dr. Petru Florin Mihances

A

Client-Server



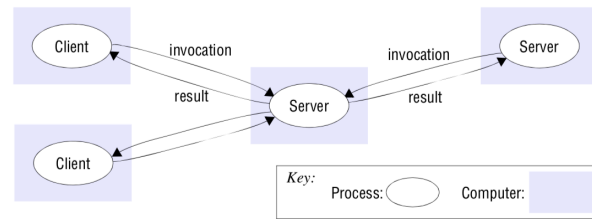
Servers

- a server may be client for another server

Dr. Petru Florin Mihances

A

Client-Server



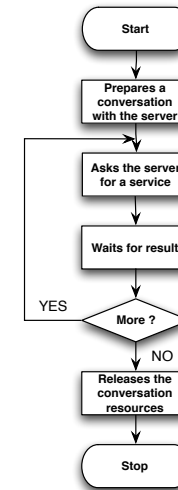
G. Coulourie et. al., Distributed Systems

Pros/Cons

- easy to add a new server, transparently upgrade a server, possible different data models in different servers
- scalability issues

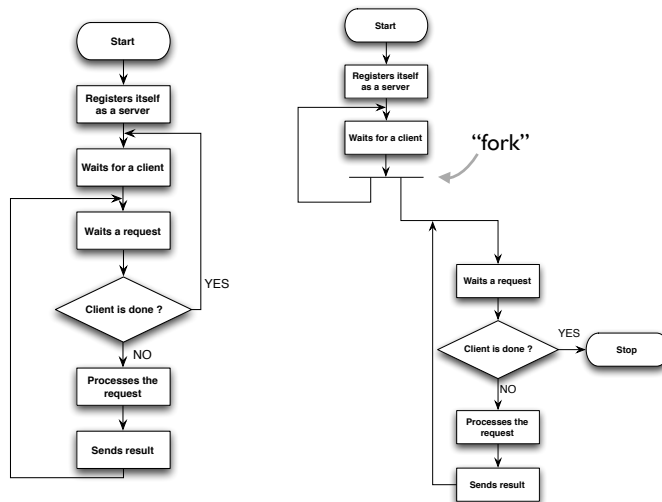
Dr. Petru Florin Mihances

Generic Client Actions



Dr. Petru Florin Mihances

Generic Server Actions



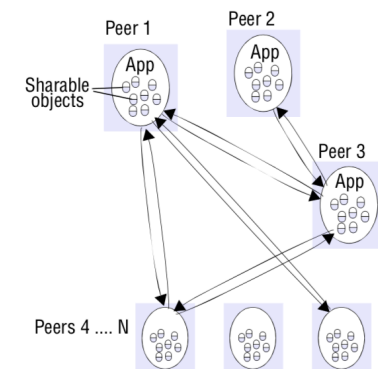
Iterative server

Concurrent server

Dr. Petru Florin Mihances

B

Peer-to-Peer



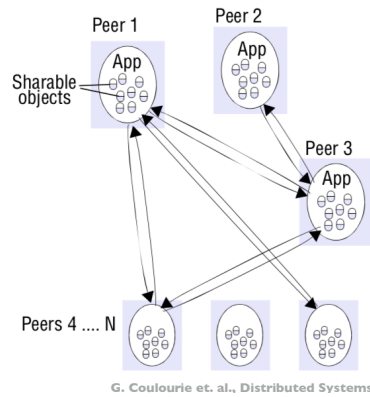
G. Coulourie et. al., Distributed Systems

Same process plays the server and the client role simultaneously
exploiting the resources (data and hardware) of each node

Dr. Petru Florin Mihances

B

Peer-to-Peer



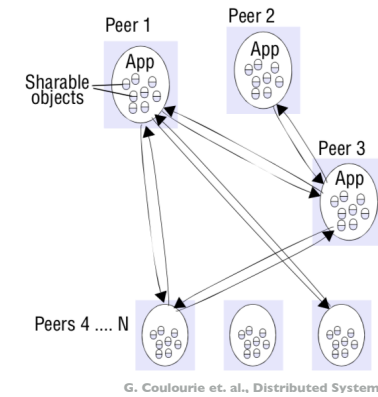
Pros/Cons

- **load** distribution, **resilience** in case of disconnection

Dr. Petru Florin Mihances

B

Peer-to-Peer



Pros/Cons

- **data replication** make the system **more complex**, **more difficult to control** due to the lack of centralisation

Dr. Petru Florin Mihances

4

The Layered Style in Distributed Systems

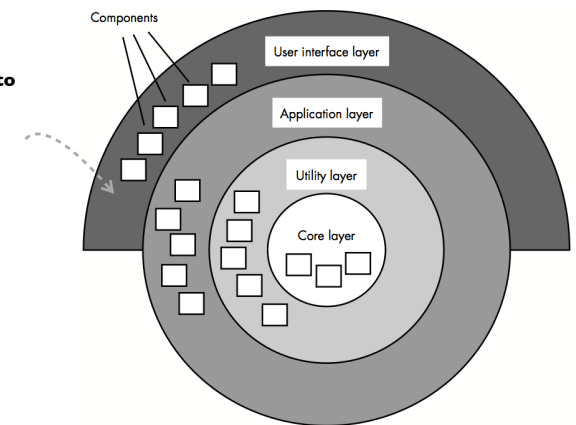
Dr. Petru Florin Mihances

Several Usages in Distributed Systems

Layered Style

Each layer is a sub-system similar to a “machine” whose “language” is represented by the services it provides to the upper layer

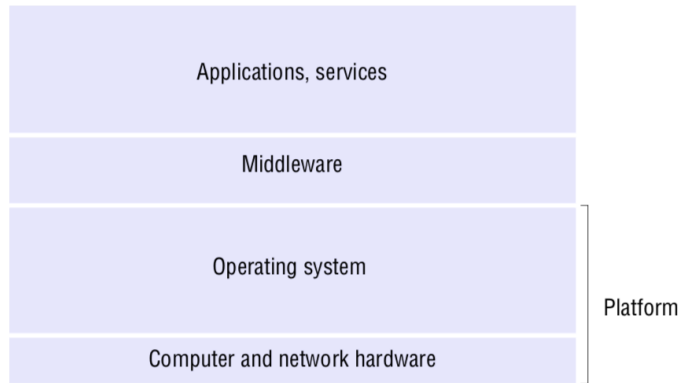
This “language”/services is implemented based on the “lower-language”/lower-level-services provided by the layer **immediately below** (i.e., layer **N** uses only the **N-1** layer)



Dr. Petru Florin Mihances

A

Middleware



G. Coulourie et. al., Distributed Systems

Software infrastructure offering higher level programming abstractions for the development of distributed systems

Dr. Petru Florin Mihances

Categories and Examples

Remote invocation between objects

Java Remote Method Invocation (RMI), CORBA

Indirect publish-subscribe communication

Java Message Service (JMS), CORBA Event Service

Indirect message queues communication

Java Message Service (JMS), Websphere MQ

Peer-to-peer

Gnutella, Openstore

application specific middleware

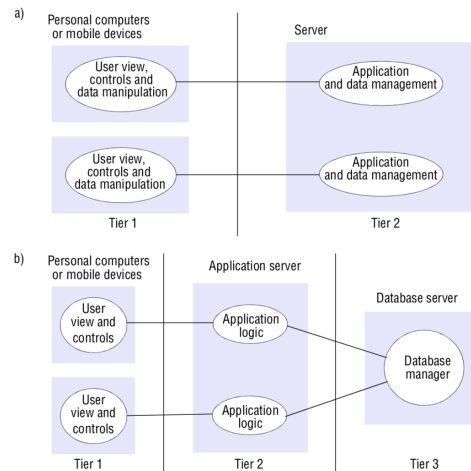
... and many others

Dr. Petru Florin Mihances

B

N-Tiere Applications

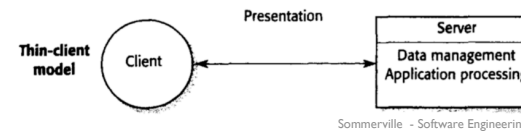
Map logical elements to physical servers



G. Coulourie et. al., Distributed Systems

Dr. Petru Florin Mihances

Client Variations

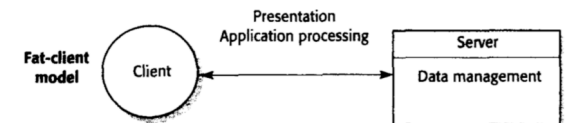


Sommerville - Software Engineering

Pros/Cons

Simple (not powerful) client devices

Higher load of server and network



Sommerville - Software Engineering

Pros/Cons

Better distribution of processing load

Maintenance issues due to application logic distribution

Dr. Petru Florin Mihances



Protocol Stacks

See **next** chapter ...