Distributed Objects and Components

Loïc Chabardès

Outline

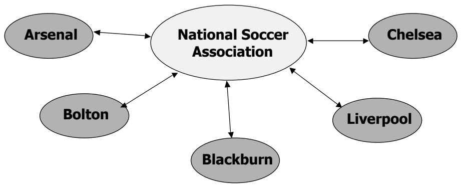
- Objects
- Components
- Distributed Objects
- Developing Distributed Objects
- Distributed Components

Objectives

- To provide a brief overview of Distributed objects and how they are developed with the use of object-oriented middlewares.
- To provide an overview of distributed components

A simple example

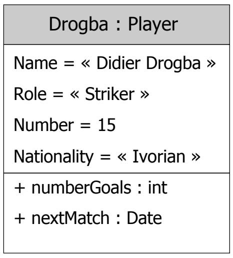
- Soccer League management system :



What is an Object?

- « *An element that combines data (properties) and behaviour (methods) in a single container of code. Objects inherit their properties and methods from the classes above them in the hierarchy and can modify them to suit their own purposes* »
- Has an internal state
- Has an unique identifier
- Equality ≠ Identity

Soccer Player Object


Drogba : Player

Name = « Didier Drogba »

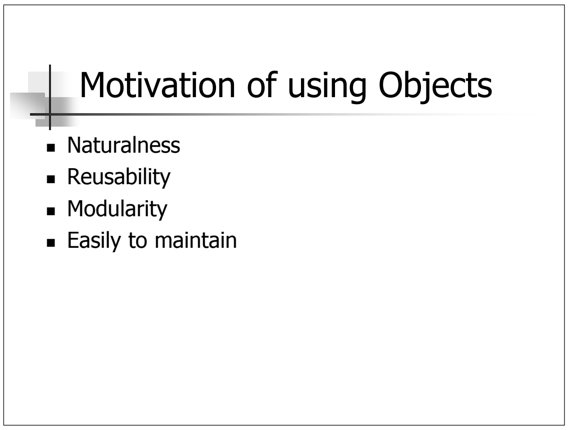
Role = « Striker »

Number = 15

Nationality = « Ivorian »

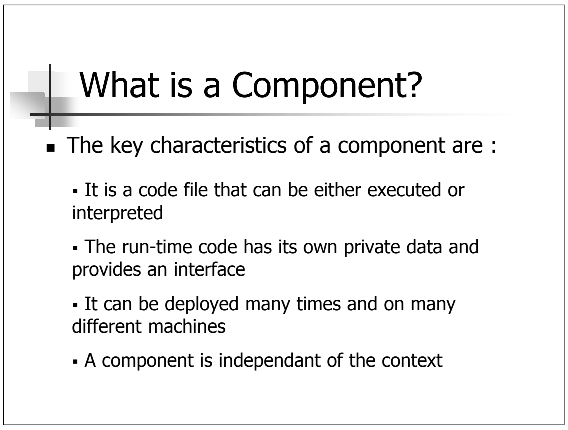
+ numberGoals : int

+ nextMatch : Date



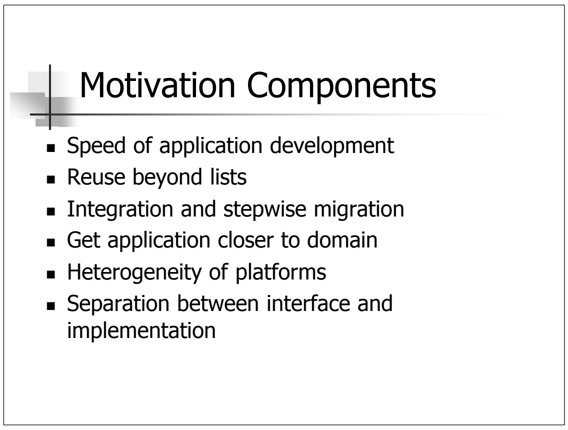
Motivation of using Objects

- Naturalness
- Reusability
- Modularity
- Easily to maintain



What is a Component?

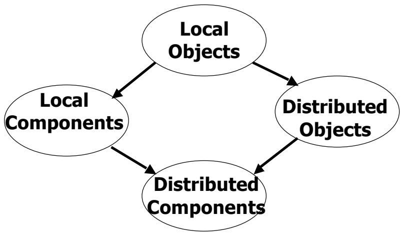
- The key characteristics of a component are :
  - It is a code file that can be either executed or interpreted
  - The run-time code has its own private data and provides an interface
  - It can be deployed many times and on many different machines
  - A component is independant of the context



Motivation Components

- Speed of application development
- Reuse beyond lists
- Integration and stepwise migration
- Get application closer to domain
- Heterogeneity of platforms
- Separation between interface and implementation

Components Infrastructure



Motivation Distributed systems

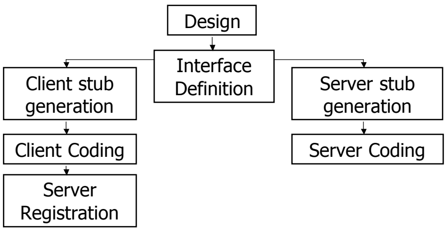
- Scalability
- Openess
- Heterogeneity
- Ressource sharing
- Fault-Tolerance

Local vs Distributed Objects

- Life cycle
- Objects references
- Request Latency
- Object activation
- Parallelism
- Communication
- Failures
- Security

Developing Distributed objects

- Object-oriented middlewares facilitate the development of distributed objects :



Design

Interface Definition

Client stub generation

Server stub generation

Client Coding

Server Coding

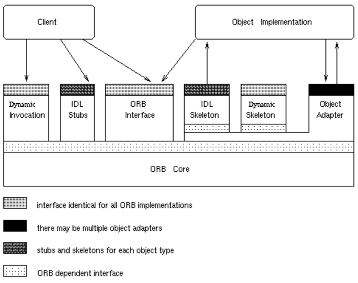
Server Registration

Object-oriented Middlewares

- OMG/CORBA

  =>Common Object Request Broker Architecture
- Microsoft COM
- JAVA/RMI

CORBA Architecture



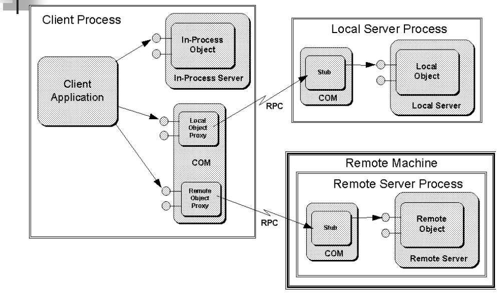
▤ interface identical for all ORB implementations

■ there may be multiple object adapters

▩ stubs and skeletons for each object type

⬚ ORB dependent interface

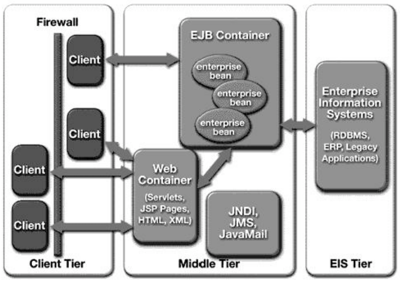
COM- part of the Architecture



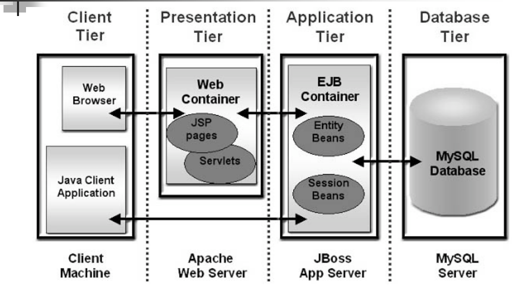
Distributed Components

- Motivation distributed objects +
  Motivation components
- Components will be configured to execute in remote locations
- A *Component Architecture* is a specification of a set of interfaces and rules of interaction that govern the communication among components and other necessary tools.

J2EE Architecture



EJB Architecture



Summary

- Distributed objects and components offer many advantages (Scalability, Openess, Heterogeneity, Ressource access and sharing, Fault-tolerance)
- Both enable to reuse remote codes, independantly of the programming languages (save of time)
- Middlewares facilitate the use of distributed objects (Corba, COM, Java/Rmi)
- Component Architectures facilitate the use of distributed components (J2EE, EJB)

References

- Engineering Distributed Objects – Emmerich, W: Wiley and sons, 2000
- J2EE Design Patterns – William Crawford & Jonathan Kaplan, O'Reilly, 2003

- www.cs.indiana.edu/~srikrish/talks/proposal.ppt
- www.pace.ch/cours/glossary.htm