CO7090 Distributed Systems and Applications

Credits: *15* **Convenor:** *Yi Hong* **Semester:** *2nd*

Prerequisites: *Essential: General knowledge on object-oriented programming, basic knowledge of Java, basic knowledge of software architectures*

Lectures:	*17*	*hours*	**Class Test Hours:**	*1*	*hours*
Surgeries:	*7*	*hours*	**Independent Study:**	*73.5*	*hours*
Laboratories:	*14*	*hours*			

Assessment: *Coursework: 40% + Three hour exam in May/June: 60%*

Subject Knowledge

Aims This course intends to equip students with notions for analysing/designing distribution of data and computations when designing and programming applications. The overall goal is to provide a critical understanding of distributed applications and middlewares.

Learning Outcomes Students will be able to: tackle distributed programming issues and analyse problems that require distribution of resources/computations; analyse and choose among the middleware models described in the course; understand and tackle issues like multi-threading and transactional interactions in distributed application; apply principles of component-based distributed programming (e.g., with respect to technologies like RMI, J2EE, etc.).

Methods Class sessions, textbook, worksheets, additional hand-outs and web support.

Assessment Marked coursework, traditional written examination.

Skills

Aims To teach students the basic principles of distributed computing and their middlewares.

Learning Outcomes Students will be able to: pinpoint the main features of middlewares for distributed applications; to identify essential elements that enable them to choose amongst the various type of middlewares; to apply prior knowledge on programming, designing and implementing to distributed applications development; to implement and evaluate a planned solution.

Methods Class sessions together with worksheets.

Assessment Marked coursework, traditional written examination.

Explanation of Prerequisites

A relevant aspect of the module is the reinforcement of material delivered in lectures with practicals involving students in the critical analysis of the middlewares presented in the lectures. Knowledge of Java, as provided in CO1003 and CO1004, is essential (inheritance, interfaces, exceptions). A basic knowledge of networks, client-server architecture and socket programming, as provided in CO2017, is required (and assumed).

Module Description

Computer networks and distributed applications have a paramount role in all-day life. Nowadays, it is hard to imagine stand-alone systems or applications. Practically, any modern computing device offers the possibility of being connected with other devices. At higher level, applications aim at exploiting networking capabilities of systems and tend to be more and more interconnected and communicating themselves.

Designing and programming this kind of distributed applications can result a hard task if not done at the appropriate level of abstraction. There are two main complex aspects to face: (i) distributed systems are frequently made of heterogeneous devices and interact through many different communication infrastructure; (ii) modern distributed systems have different tiers (e.g., TCP/IP level, operating system, network system, etc.). Middlewares provide an abstraction of many low-level details of systems. They are meant to simplify software development

and application integration by interfacing the application level with lower tier of distributed systems so that the programmer does not have to worry about implementation details. Also, middlewares allow the programmer to integrate applications developed for different execution context and in different times.

The course reviews some notions of concurrent and distributed programming (e.g., threads and RMI) and presents the main models and principles behind the middlewares that in the last years many vendors (Microsoft, IBM, Sun, Oracle) have proposed. In fact, these proposals differ each other not only with respect to the technologies or architectures adopted, but also with respect to the underlying coordination models.

Syllabus Introduction to distributed systems; Programming with threads; RPC and JAVA/RMI; Message oriented Middlewares; Event/Notification; Enterprise Java Beans; Distributed coordination.

Reading List

[B] A. S. Tanenbaum and M. Van Steen,, *Distributed Systems: Principles and Paradigms,* Prentice Hall, Inc..

[B] Rick Leander, *Building Application Servers,* Cambridge University Press, 2000.

[B] Daniel Serain, *Middleware and Enterprise Application Integration,* Springer, 2002.

Resources Course notes, web page, study guide, worksheets, handouts, lecture rooms with two OHPs.

Module Evaluation Course questionnaires, course review.