CONTENTS AND METHODOLOGY OF MIDDLEWARE PROGRAMMING FOR DISTANCE LEARNING IN MASTER PROGRAMS

Felipe Garcia-Sanchez, Antonio-Javier Garcia-Sanchez and Joan Garcia-Haro
Department of Information Technologies and Communications, Technical University of Cartagena
Campus Muralla del Mar, Cartagena, Spain

Keywords: Distributed Programming, Distance Learning, Master Program.

Abstract: This paper deals with a well-known problem, the learning of distributed programming languages, especially when they are included in distance programs. In particular, the course is focused on middleware learning in a master program, in which students that access to it have heterogeneous programming level and skills. Therefore, a real scenario and a real case study are presented, where students must rationalize their effort. An emphasis is placed on those students that offer more weaknesses. Moreover, this paper presents the students evaluation process, and summarizes the results obtained after the course had been developed.

1 INTRODUCTION

Learning in the field of Object Oriented Programming (distributed OOP) and, in particular, learning middleware, is a well-known issue in software teaching. It gets worse when students face distributed OOP for the first time or when the course is included in a postgraduate degree where students come from different study programs. Sometimes, their degrees do not include subjects typical of traditional Computer Science Degree in their courses syllabus. Therefore, some of the contents and methodology of traditional courses for graduate degrees must be reviewed.

Knowledge of middleware technologies is commonly required by different market employees. Therefore, these related topics must be included in the program for Computer Science degrees, including master ones.

This document describes and emphasizes the contents and methodology followed to develop an intensive course of *distributed programming systems*, in particular regarding *middleware*, and it is oriented for distance learning. The motivation is to include this course in a professional master program where students normally have strict working schedules in their respective companies. In this paper, the experiences and the results achieved are presented and discussed.

The main objectives for the course are to advance in the knowledge of the programming issues and to provide an appropriate background for the students so that they can face their own professional tasks. Specifically, the goals are: (a) to get a good and sufficient knowledge for understanding and applying the working technologies, (b) to select the most appropriate technology for a particular problem, (c) to establish the requirements for adapting a given solution, (d) to develop applications and distributed systems using these technologies, and (e) to analyze and evaluate the application performance.

In this paper, we detail the learning for two main technologies (although the program includes additional topics), namely: the general-purpose middleware CORBA (*Common Object Request Distributed Architecture*) (Bose et al., 2001) and the commercial system based on .NET communications. On the one hand, CORBA is general-purpose, commonly free or non-licensed distributions are provided and it is open-platform (not limited to any operating system). On the other hand, the .NET technology is commercial, owned by *Microsoft Company* and operates over its operating system, the *Windows* family.

The course consists of 3 ECTS (*European Credit Transmission System*) (Dir.-Gen. Education, 2004) and the students are required to have basic programming knowledge. It is assumed that they

Garcia-Sanchez F., Garcia-Sanchez A. and Garcia-Haro J. (2009).
CONTENTS AND METHODOLOGY OF MIDDLEWARE PROGRAMMING FOR DISTANCE LEARNING IN MASTER PROGRAMS.
In *Proceedings of the First International Conference on Computer Supported Education*, pages 230-234
DOI: 10.5220/0001976702300234
Copyright © SciTePress

were instructed in the common programming language, Java (Bose et al., 2001), and they are thus, familiarized with objects and classes. Technology degrees required to access the master course should include it; however, the master program includes a previous levelling course for those students with low skills in any discipline as programming in our case. CORBA learning is developed on Java programming language. However, as .NET does not include any interface for it, the course includes learning in C# language.

Material accessible on-line is available for students. In particular, slides, class notes, exercises and referenced books for theoretical concepts, and pieces of developed software (complete and partial solutions for diverse cases of study) for practical experiences. In fact, students have unlimited open access to a computer room where they may develop their own solutions. In it, computers have installed Java programming environments as *NetBeans* (Boudreau et al., 2003) and *Eclipse* (Pluta, 2003) and the .NET one, the *Microsoft Visual Studio* (Petzold, 2002).

The rest of the paper is organized as follows: section 2 outlines previous work on this topic, section 3 defines and details the contents included in the course, section 4 shows the methodology employed for distance learning, leaving section 5 for concluding and showing the results obtained for students.

2 PREVIOUS WORK

As far as the authors know, a recent paper summarizing contents and methodology included in a middleware course can not be found. The closest one may be the Brownsmith (Brownsmith, 2007) where the topics included in a middleware course are detailed. It is aimed at offering a critical insight in order to be able to select the right commercial middleware for determinate professional tasks.

Following the same idea, many universities, postgraduate institutions, academies, etc. offer a wide range of courses in middleware. They may be oriented for enterprise solutions and focused on a particular or proprietary option. When the course is aimed for a general-purpose technology, an existing middleware is selected and the course is entirely devoted to it. This paper presents the contents for an integrated course where the most representative technologies are chosen and developed.

3 CONTENTS INCLUDED IN THE MASTER PROGRAM

3.1 CORBA Middleware

CORBA teaching is always a complex task. The density, tools, options, and possibilities that this technology offers cannot possibly be developed in a single course assigning 1 ECTS to it. For this reason, the main objective is to provide the basic knowledge, management and development. Contents are categorized into three types of concepts:

Basic Concepts: Middleware definition. Heterogeneity. Extensibility. Security. Scalability. Failure Management. Concurrency. Transparency. ORB (*Object Request Broker*) Architecture. Invocation System. The IIOP (Interface Inter-ORB Protocol). The Object Definition Language IDL (Interface Description Language). Application Programming.

Applied Concepts: Remote Invocation Handle. ORB Handle. Input and Output Options. Compilation and Execution. Implementation and Invocation Types. Service Implementation. Client Implementation, etc.

Divulging Concepts: CORBA Programming Environment. CORBA Data Structure. Three-Band Services. Collection Service. Concurrency Service. Event Service. Notification Service. Name Service. Trade Service. Time Service, etc.

All topics have not the same relevance, as it depends on their difficulty and the desired learning level. For instance, the ORB invocation system is retaken in different lessons, at the beginning of them. The enumerated CORBA services are just commented or referenced because only the *Name Service* is used by the students for the practical exercises. As for basic concepts, students must understand the difference between using CORBA and using a common client-server system and its invocation model. They have a first topic about the motivation to use an invocation system of objects and to develop a distributed system. In particular, they know the competitive advantages that CORBA offers with respect to other solutions: its capability as “global communicator” for different systems, languages and application environments. For this reason, students develop a communication between a client and an object or service placed in a remote server. Concepts as *stub, skeleton POA (Portable Object Adapter)* or the *IDL definitions* are used for this purpose.

As for the applied contents, most of them are used for emphasizing the basic contents previously introduced. However, students improve from a simple application such as “HelloWorld” to the implementation of complex IDL’s, including generation of own types, input-output attributes, etc. The master timetable allows for a fast evolution.

Finally, the divulging contents are referred to in just one lecture, with several working examples. Their detailed architecture is not considered, although students know their applications, especially the most commonly ones used. In particular, the *Event*, *Notification* and *Trade Services* are related to different systems as peer-to-peer, brokering, etc.

3.2 .NET Architecture

In comparison to CORBA, the .NET platform has a more commercial orientation. Regarding the Internet Services field, *business-to-business* (b2b) or *business-to-client* (b2c) are examples of what providers have offered to users in recent years. The objective of the .NET platform is to give a solution to Internet Services, developing communications and applications easily.

.NET is the middleware solution offered by *Microsoft* for developing distributed applications among servers or servers and clients. Many concepts and contents previously mentioned in the CORBA subsections must be also applied here, and they are not explained again. However, .NET does not only allow communication and information transmission. It also integrates applications with other oldest platforms as DCOM, for upgrading the knowledge of old programmers to b2c or b2b services.

Therefore, the students’ qualification has three main objectives. First, they need to understand the .NET architecture. .NET offers different possibilities that must be known (*ADO.NET, ASP.NET, Windows Form, Web Forms, etc.*) and the CLR management (*Common Language Runtime*) for designing more efficient applications. Secondly, students develop interfaces or client applications. It is the base for the b2b and b2c services. Thirdly, students work on interfaces for transparent user access.

These three main goals involve different concepts and competences that students must acquire. The students have class notes, tutorship sessions and referenced books for their learning or search. Moreover, they perform intensification assignments about topics proposed by the lecturer and related to the three following parts:

.NET Architecture. Students must know the relevance of the .NET platform previously to develop their applications. Several questions arise and have to be answered: What are the compatible devices? What is the description language? What are the developing tools? What are the service types designed? What is the server for executing the services? Etc. All of them are requirements for learning this technology.

Applications and Service Development. The main .NET objective is the development of web services (ASP.NET) or databases (ADO.NET) and the design of applications for interconnecting enterprises. Therefore, the first main topic is the standard definition for information interchange; and the protocols XML, WDSL, UDDI and SOAP have to be introduced.

Due to the heterogeneity of the programming language provided by .NET platform, C# is chosen as the language used during this course. The reason for this is that many libraries and examples are written in this language and it is the most similar to Java, that was employed in the CORBA section. Students learn concepts in C# such as classes, objects, arrays, polymorphism, inheritance, etc. and they also extend their knowledge about other interesting concepts such as work cycle, the garbage collector, etc. The lecturers remark the main differences between C# and other languages, mainly Java and C++. The material provided allows students to know and search for any particular topic that they require about this language. At the end of the course, students recognize all the abilities provided by *Microsoft* for communications among enterprises and clients.

Information Transmission. .NET offers different solutions for the communication among servers and clients, ranging from simple classes for developing sockets (simplest communication way) to most advanced mechanisms such as *.NET Remoting* (Conger, 2003) (Rammer 2002). In particular, the latter permits the distributed communication supporting different options.

4 METHODOLOGY FOR DISTANCE LEARNING

4.1 CORBA Learning

As mentioned in the contents section, CORBA teaching is a difficult task. As it is the case with most programming learning, the best way to approach it is to try to resolve problems at different levels of complexity. However, distance learning and the limited access to the lecturer creates a particular scenario to this main idea.

In this case, the first task is the explanation of already resolved examples for the basic concepts mentioned in the previous section. They are shared among students, and the concepts are related to client-remote service applications. When a particular concept is studied, its associated code is emphasized.

The complete comprehension of these concepts is particularly critical. CORBA is very sensitive to code failures due to its different component coordination. A small programming error will be reproduced in several compilation and execution failures. This problem makes students become discouraged. They must face their problems alone when they have enough knowledge to do it. Before this, students solve the simplest exercises with a set of usual failures that they may look up. This makes them fluent enough in programming, gradually improving until their own exercises resolution.

When the basic concepts are fixed, the most advanced ones are introduced. They are first introduced theoretically by referenced books, prepared class notes and resolved questions. Practical exercises are included too, getting more relevant (in length and difficulty) when the course approaching its end.

Both theory and practice imply different tasks. In the theoretical part, students have a range of well-known exercises, such as the “HelloWorld”, a basic calculator, etc. The goal is for students to become familiar with the CORBA architecture. On these same cases, students have suggested improvements that they may conduct. Usually these improvements are simple and students may resolve them throughout the course. Students get confidence and lose the “fear” of this environment. They do not have to resolve any exercise from beginning to end at this level, since it could result in the students giving up.

In the practical part, students face small problems, using the material provided or searching for it. The lecturer does not have an important presence, and his mission is to make sure that all students achieve the desired knowledge level.

In a second level, students have more experience (even the heterogeneity of the group may cause some of them to even consider the first level easy). They implement more complex cases as a simple database, where the lecturer “presence” is more active. Doubts arise when they implement the different topics assigned.

Regarding the second level of practical exercises, students develop an introductory first practice where CORBA and its invocation system are presented. They compare a simple client-remote server CORBA application with an also simple sockets one (based on the communications interface included in the operating system). Students learn the code, compile it and execute it. Thereby, they know what a middleware means. Moreover, they learn other key issues, such as a benchmarking tool to analyze applications.

The second practice is slightly more complex. They perform a three-band communication. The central server is an application that reads data from a remote database, (which may be referred to by any service). Client application refers to the remote server for getting information from that database. Students learn more realistic cases, and they face new problems such as the concurrency one. The lecturer must follow the students progress.

Finally, the tutorship has special relevance along the course. Students with low background require a special dedication. More examples and cases are prepared for them. On the other hand, the most advanced students do not employ many hours. However, their monitoring is also required for ensuring the entire right comprehension.

4.2 .NET Learning

Once the students have acquired the theoretical concepts of this technology, the practical training sessions start, consisting of the development of applications. The objective is to put the previous competences into practice, acquiring new ones. These applications form a set of activities which will be reported in a document. Furthermore, they will be managed by a lecturer in an open access laboratory, in tutorship timetable, and via email.

The activities have an increasing difficulty, and so that, they start with simple applications programming. Students train with the development tool provided by the .NET platform. Subsequently, students implement codes based on the communications among clients and servers. The course program concludes with a .NET *Remoting* and a simple b2b service using this communication methodology.

The first activity introduces students to the *Visual Studio* environment which is provided by *Microsoft.* All codes are implemented in C# language and the objective is to develop .NET applications. In this first activity, students perform the simulation of banking transactions, so actions as balance enquiry, expenditure, deposit, etc. are implemented. To this aim, concepts like classes, objects, arrays, inheritance, etc. are handled. This activity allows students to apply methods and functions learnt in theory lectures.

The second activity introduces students to the graphical interfaces, in *Windows Forms* of the .NET platform. It proposes developing an application that allows to start with visual objects (buttons, boxes, commands, etc.). For facilitating its programming, the exercise is divided into two parts: (i) students develop the code which allows to open an image file in a particular format (bmp, gif, jpeg or tiff) and (ii) the file opened in the previous format is converted into another format type. The objective is twofold. First, to acquire the knowledge of the components, objects and classes which are necessary for carrying out a graphical application. Secondly, students acquire the competences for searching in the help menus that *Visual Studio* .NET provides to them.

The third activity is based on the client-server communication. The purpose is to develop a visual environment and the same functionalities as the well-known "Messenger" application. Two users (irregardless of where they are placed) may communicate with each other. To carry out this goal, the lecturer identifies which objects are the most suitable and he shows the students the different methods and arguments that they may use for implementing the "Messenger" application. From this instruction and the .NET help, students program the code for the peer communication, performing sessions among users (via multicast) and implementing an appealing graphical interface. Finally, students check their programs in the laboratory (*Microsoft Visual Studio* and libraries are available), testing the code, correcting and improving it.

The last activity is the implementation of a remote application using the .NET *Remoting* library. It consists of creating a database where the user adds data in the fields implemented by students. These fields are stored in an object placed in a remote computer. Students develop the functions and methods for performing the query action in a local computer and the transmission of these queries to the remote computer. It returns the result of the search requested by the user. The lecturer trains students to gain all the necessary competences for doing this activity through traditional lectures and bibliography. In particular, students gain experience in the functions and methods that allows to implement a distributed communication via .NET *Remoting*. Furthermore, students apply the concepts learnt related to the XML and SOAP protocol.

5 CONCLUSIONS

In this paper, the authors have presented the concepts and methodology applied to the teaching of the distributed objects programming and middleware in a postgraduate master for Computer Science. The contents are standard in relation to specific middleware, but with two particularities: (i) it highlights the practical implementation and (ii) it offers a global vision to students, enabling them to easily resolve future problems based on these technologies. As for the employed methodology, it is limited due to the distance learning. Material and references are carefully selected, providing an appropriate scenario where students may found their own requirements and improvements.

Finally, the course started in the recent academic year of 2007/2008 with 30 students. Within this period, two evaluation dates were given: June and September 2008. In June, the marks were the following: 53% of students passed the subject, 30% got B marks, 7% obtained A and 10% of students did not take the exam. In September: the only student that took it passed the exam.

ACKNOWLEDGEMENTS

This paper has been supported by project grants TEC2007-67966-01/TCM (CON-PARTE) and TIC-TEC 07/02-0002 (PLEDAX) and it is also developed in the "Programa de Ayudas a Grupos de Excelencia de la Región de Murcia, de la Fundación Séneca, Agencia de Ciencia y Tecnología de la RM".

REFERENCES

Bose G., Vogel A.. Duddy K., 2001. *Java Programming with CORBA*. Ed. OMG Press John Wiley and Sons.
Directorate-General for Education and Culture, European Commision. *ECTS Users' Guide.* Brussels. 2004.
Boudreau T., Glick J., Greene S., Spurlin V., Wochr J. 2003. *NetBeans: The Definitive Guide*. Ed. O'Really.
Pluta J., 2003. *Eclipse Step by Step.* Ed. MC Press.
Petzold C., 2002. *Programming Windows with C#*. Ed. Microsoft Press.
Brownsmith J.P., 2007. Teaching and Learning Middleware: A new course. *ACM Journal of Computing Sciences in Colle., vol 22, 3, pp. 251-256.*
Parihar M., 2002. *ASP.NET*. Ed. Anaya.
Robinson S., 2003 *Professional C#.* Ed. Wrox.
Conger D., 2003. *Remoting with C# and .NET.* Ed. Wiley.
Rammer I., 2002. *Advanced .NET Remoting*. Apress.