Teaching Queuing Systems Modeling Using UML

Athanasios Perdos
Ph.D. Student
Department of Applied Informatics, UOM
156 Egnatia st., GR- 54006
Thessaloniki Greece
+306944254662
perdos@uom.gr

Alexander Chatzigeorgiou
Lecturer
Department of Applied Informatics, UOM
156 Egnatia st., GR- 54006
Thessaloniki Greece
+302310891886
achat@uom.gr

George Stephanides
Assistant Professor
Department of Applied Informatics, UOM
156 Egnatia st., GR- 54006
Thessaloniki Greece
+302310891872
steph@uom.gr

Abstract

In this paper we suggest a new approach in the way that a network modeling and simulation or a queuing theory course can be developed. Beyond the mathematical model that sometimes is too difficult for students to understand, this paper introduces the use of Unified Modeling Language as the mean to teach modeling of discrete event systems such as queues and networks. The basic course scheduling and teaching material are presented too.

1. Introduction

The great majority of courses relevant to network modeling and simulation or queues require very good knowledge of Probabilities Theory and Stochastic Processes. Lots of problems have arisen because of this requirement. Students usually lack the demanded background in mathematics. Thus, teachers are obliged to explain and clarify the mathematical equations something that requires a lot of valuable time. On the other hand students are disorientated from their objective goal because of the difficulties that face in understanding the equations. The teaching method that has been followed in this area, involved the following aspects:

1. Presentation of basic concepts of queuing theory
2. Presentation of the mathematical model.
3. Evaluation of some simulation results.

The disadvantages of this method are the following:

1. It requires a very good knowledge of mathematics to understand the equations that describe the theoretical model.
2. Some types of queues are very difficult to model using mathematics.
3. It demands a great amount of efforts to create a simulation software program that can be used as a supplement to a network or queuing theory course providing a basis for a laboratory work.

Our emphasis in this paper is to propose and present an innovating teaching method which purpose is two fold. First, it can serve as the basis for a course released from the problems that have been mentioned above. This method is based on the Unified Modeling Language and we believe that can ease the work both of educators and students because anyone that studies informatics has obtained the demanded knowledge for object oriented methods since the first or second year. Secondly, since UML is the tool to model it is possible to generate code for a simulation program using a tool such as Rational Rose [1].

2. Course Schedule

2.1 Basic Concepts of Queuing Theory

In the beginning of the course the basic concepts of queuing theory must be presented. There is no need to present the mathematical equations that have been used until now to model queuing systems.

Queuing theorists in order to classify queues use a shorthand notation called the Kendall notation [2] which defines the class A/S/m/c/p/SD as follows: “A” is the interarrival time distribution. “S” describes the service time distribution. “m” denotes the number of servers available to give service to jobs in queue. "c" defines the capacity of the buffer. If c is unspecified, it is assumed to be infinite. "p" defines the maximum number of jobs. If p is unspecified, it is assumed to be infinite. "SD" stands for service discipline which is a very important aspect of queues and is related to the way that a server decides which job in the queue to pick next for service.

Some other significant variables of queuing theory are: "τ" = interarrival time, "λ" = mean arrival rate, "s" =service time per job, "μ" = mean service rate per server, "n" = number of jobs in the system, "nq" = number of jobs waiting to start service, "ns" = number of jobs currently being served, "r" = response time, "w" = waiting time. Since all these concepts will be used in a simulation program, random number generators for the mean arrival time and the mean service time must be used. We propose that these generators should use the following distributions: Normal, Exponential, Hyper – Exponential and Erlang.

Good references for the above queuing concepts are the following: [3], [4], [5], [6]

2.2. UML Diagrams

All the queuing concepts that were discussed in the beginning of the course are involved now in the presentation of the UML diagrams. We believe that a short presentation of the diagram types and their characteristics is necessary before queues modeled using UML [7], [8], [9].

The class diagrams that model this system are:

Queue and its Service Disciplines are specified in the following diagram

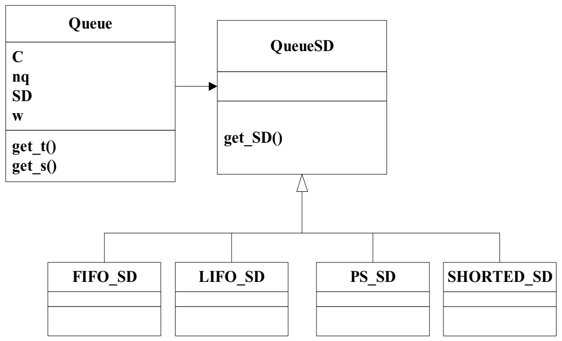

Figure 1. Queue Service Discipline Class Diagram

This is a typical example of the Strategy Design Pattern [10] according to which a family of algorithms (Service Disciplines) has been encapsulated and thus made interchangeable. This pattern lets the algorithms (QueueSD) vary independently from the client (Queue) that use it.

For arrivals and service times the diagram for the distributions that produce random times is the following:

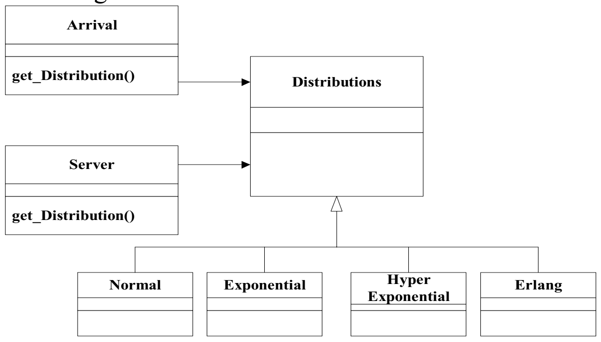

Figure 2. Distributions

As it can be observed from Figure 2, one axis of change [11] has been anticipated, namely the possibility for enhancing the system with new distributions. The clients (Arrival, Server) use distributions employing polymorphism: as a result, the addition of a new distribution will not demand any modification of the clients. Consequently the proposed Object oriented design employs the Open – Closed Principle, according to which a design should be open for extension but closed for modification.

Both Queue and Distribution Classes cooperate with Simulation (Figure 3) in order to model a queuing process.

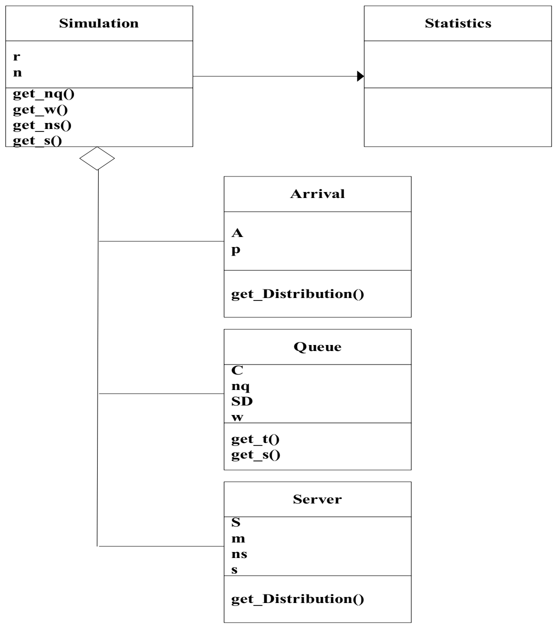

Figure 3. Simulation Class Diagram

Beyond the class diagrams it is useful to show how the system is modeled using activities, state and sequence diagrams.

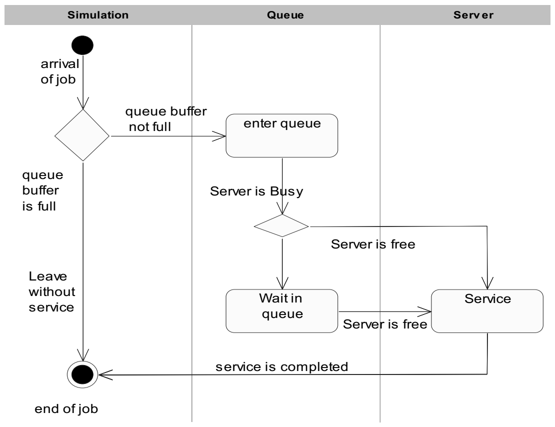

Figure 4. Activity Diagram Using Swimlanes

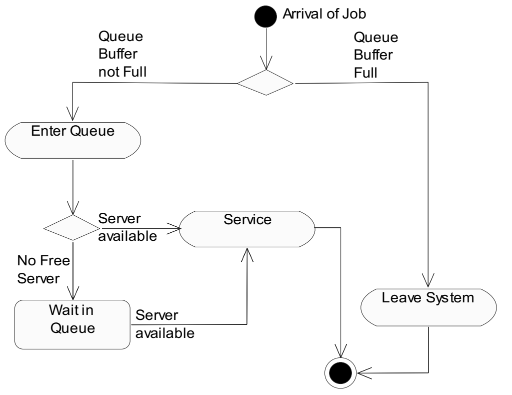

Figure 5. Activity Diagram of an Arrival Event

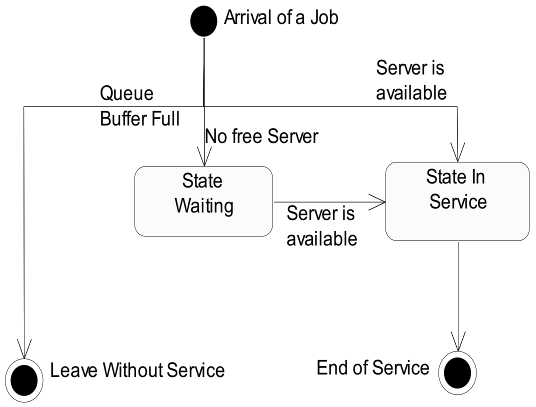

Figure 6. State Machine Diagram for a Job

2.3 Laboratory Assignment

Students are asked to model the M/M/1 queue(M stands for Markovian or memoryless (exponential)) and to develop an application that simulates it. They can use any object oriented programming language such as Java or C++. The aim of this laboratory assignment is to evaluate students' understanding of the course and see how they can pass from process modeling to simulation.

3. CONCLUSION AND FUTURE WORK

We believe that using UML to model not only queues and networks but any system gives a good understanding about the system properties and the interaction between its parts. Thus students obtain the appropriate knowledge easily and without facing problems with the high level of mathematics that is required. Beyond the teaching process our aim is to develop all the classes that the system consists of and give the students a methodology to create an integrated application.

4. References

[1] URL: http://www.rational.com/uml

[2] URL: http://www.omg.org/cgi-bin/doc?formal/03-03-01.

[3] Jain, Raj The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling by John Wiley & Sons, Inc. (1991)

[4] Leigh, J.R. (1983) Modelling and Simulation Peter Peregvinus Ltd London.

[5] Hoever Stewart V. and Perry Ronald F. Simulation, A Problem-Solving Approach by Addison-Wesley Publishing Company. (1989)

[6] Morse Philip M. Queues, Inventories and Maintenance John Willey. (1967)

[7] Quatrain, Terry Visual Modeling with Rational Rose and UML. Reading, MA. Addison Wesley Longman, Inc. (1998)

[8] Kruchten, Philippe. The Rational Unified Process – An Introduction. Reading, MA. Addison Wesley Longman, Inc. (1998)

[9] Booch, Grady; Rumbaugh, James; Jacobson, Ivar. The Unified Modeling Language User Guide. Reading, MA. Addison Wesley Longman, Inc. (1999)

[10] Gamma E., Helm R., Johnson R., Vlissides JDesign Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, Boston, MA. .,(1995)

[11] Martin R. C. Agile Software Development: Principles, Patterns and Practices, Prentice Hall, Upper Saddle River, NJ. (2003),