Journal of Software Engineering and Applications, 2013, 6, 48-52
http://dx.doi.org/10.4236/jsea.2013.62008 Published Online February 2013 (http://www.scirp.org/journal/jsea)

Deadlock Detection and Avoidance in Static Step Topology under Distributed Environment

Taskeen Zaidi[1], Vipin Saxena[2]

Department of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, India.
Email: taskeenzaidi867@gmail.com, vsax1@rediffmail.com

Received December 6th, 2012; revised January 5th, 2013; accepted January 14th, 2013

ABSTRACT

During the past years, the distributed computing approach has become very popular due to various advantages over centralized approach. In the distributed approach, the execution of a process has reduced and also it requires low cost for installation. Many of the researchers are using the modeling approach for solution of the software and hardware architecture research problems. The most popular approach of modeling is known as Unified Modeling Language based on the object-oriented technology. In the present work, a method of deadlock detection is explained for the newly proposed static step topology for the distributed network. In the step topology, the processes are taken as a task, sub task, macro, subroutine, etc which are executed in reflexive and symmetric manners when the systems are interconnected to each other under distributed environment and avoidance technique is also presented for the same. The deadlock detection technique is presented through a UML class model.

Keywords: Distributed System; UML Class Model; Step Topology; Deadlock Detection; Deadlock Avoidance

1. Introduction

In the current scenario, many of the software companies are changing the layout of computing labs from the old centralized computing approach towards the distributed computing approach as it involves low cost and saves the execution time of the tasks. Another advantage is that distributed computing system does not share the global clock. In the centralized approach, computing time of the tasks is much more in comparison of the distributed approach as the deadlock occurs frequently in the centralized approach. In the distributed computing approach deadlock situation can be handled by the use of mutual exclusion *i.e.* synchronization of the tasks. The present work is based upon a distributed kind of network connected through the static step topology designed by the authors; deadlock situations are detected and methods for avoidance of these situations are elaborated. In the present work, a static step topology is used for the high speed National Knowledge Network (NKN) connectivity and deadlock detection technique is defined for the execution of the processes/tasks or one transmits the information in terms of files from one device to another device attached through the topology. A method of avoidance is also explained. On the another aspects, modeling is necessary before executing the research problems as it saves the time and errors can be detected at the early stages of the software development, therefore, in the current scenario, many of the researchers are using the object-oriented Unified Modeling Language to construct the various models for the solution of the research problems, therefore, in the present work UML class model is also designed for finding the conditions of deadlock and a procedure to minimize the deadlock situations occurring in the distributed environment.

2. Related Work

In the current scenario and due to evolution of high speed network connectivity, many of the researchers share the information at the remote by using the remote login or transferring the information over the distributed network. Sharing of information in form of files, data, audio, video as well as mutual exclusion of tasks are well explained in [1]. The static interconnection of the network plays an important role for sharing the files and it is known as the topology and an important reference Tanenbaum [2] has described various types of topologies used to interconnect the devices under distributed computing environment. Various kinds of topologies like centralized, decentralized and hybrid, which are used for modeling, and internetworking of distributed systems are explained in [3]. Another important reference is Hwang [4] which describes all the aspects related to the software and hardware architecture including the parameters affecting the execution of the processes under the distributed computing environment. Due to evolution of the windows programming, the tasks called as processes are synchronized when executed on the devices and obviously deadlock will occur according to the conditions of deadlock and these are well described by Milenkovic [5] and it consists of necessary conditions for deadlock as well as avoidance, prevention and detection of deadlock under distributed computing environment. For execution of the processes under distributed environment, mutual exclusion takes place and processes are executed according to the clock of the device and one of the important references is Lamports [6] which has explained the execution of processes under distributed environment using time, clock and ordering of events. Later on Ricart and Agrawala [7] have modified the Lamport's algorithm. Symmetric execution of processes using parallel operations is explained by Maekawa [8]. Various Scientists have solved the problem of mutual exclusion for the distributed environment by taking the concepts of graphs, sets, etc. By the use of set theory, it is solved by Agrawal [9]. Suzuki and Kasami [10] have described a token-based ring algorithm for execution of processes.

Modeling plays an important role for the solution of the research problem. Many of the researchers have used the latest modeling language known as Unified Modeling Language (UML) based on the object-oriented technology to solve the software and hardware architecture problems. First time in the literature, Unified Modeling Language for solution of the hardware architecture problem is proposed by Gomma [11] for designing the concurrent, distributed, and real-time applications with UML. Later on Pllana and Fahringer [12,13] have suggested profiles for modeling of distributed and parallel applications by using UML. Let us describe some of the important references based on the present aspects. Saxena and Arora [14] have proposed UML modeling of a protocol for establishing mutual exclusion in the distributed system. Recently, Saxena and Zaidi [15] have proposed a new topology called as static step topology for interconnection of systems under distributed environment. Then, UML Modeling of newly developed static topology called as step topology is done by Saxena and Zaidi [16] and also compared with the well known bus topology. Saxena and Zaidi [17] have also studied objective and impact of NKN project as well as advantages, disadvantages and difficulty faced by this project. A case study has also been done by authors to compute percentage utilization of NKN over the step topology. The authors [18] have also modified the Lamport's algorithm for the step topology for execution of processes in reflexive and symmetric manners.

3. Background

3.1. Distributed System

A distributed system is an autonomous collection of computer systems, which connects the resources and users through message passing technique. It provides transparency, openness, reliability, scalability to the devices connected and also allows data sharing, resource sharing, audio and video sharing and reduces the cost of infrastructure. A representation of devices under distributed environment is shown in **Figure 1**.

3.2. A Process/Task

A Process as shown in **Figure 2** is defined as tasks, subtasks, macro, subroutine, and a computer program running concurrently with other programs executed by using processor. Process Execution Controller (PEC) *i.e.* processing_unit is fully responsible to execute the processes/tasks. A UML class diagram of process is represented below with attributes and methods.

3.3. Deadlock

In operating system, deadlock is defined as a condition in

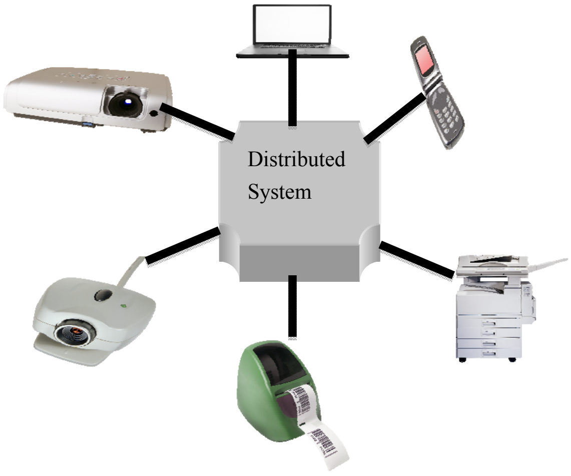
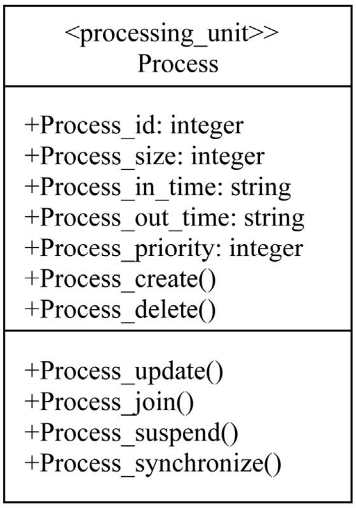


Figure 1. A distributed system.



<processing_unit>>
Process

+Process_id: integer
+Process_size: integer
+Process_in_time: string
+Process_out_time: string
+Process_priority: integer
+Process_create()
+Process_delete()

+Process_update()
+Process_join()
+Process_suspend()
+Process_synchronize()

Figure 2. UML class for process.

which two or more processes are competing for resources and each waits for other to finish its task and thus deadlock occurs. It arises because one process is requesting for resources which is already taken by another waiting process and which in turn is requesting for resource that is held by another waiting process. It is a common problem occurs in distributed system. The necessary and sufficient conditions of deadlock [5] are given below:

1) Mutual Exclusion—only one process is allowed to use the resources at one time; other processes are put in the suspended queue;

2) Hold and Wait—a process is requesting for resources which is held by another process;

3) No Preemption—the resources which are held by one process can not release if it is being used by other process;

4) Circular Wait—a process is waiting for resources held by another process which is already waiting for resource by another process, a cycle formed and deadlock occur.

4. UML Modeling for Deadlock Detection

On the basis of above aspects, a static step topology as shown in the **Figure 3** is considered in which one step consists of three nodes and processes can share the resources by message passing technique.

In the above Figure N_1, N_2, …, N_M show the devices which are arranged as per the step. The detailed description of the topology can be found in [15]. In this figure, devices are attached on high speed NKN network connectivity. The speed of data transfer from one device to another device is 1 Gbps which can be extended up to 10 Gbps.

For detecting the deadlock conditions in the step topology over NKN network, a general UML class model is proposed in **Figure 4** which shows the static behavior of the problem in which attributes and operations are grouped together to form a class. The figure represents the class diagram for deadlock condition, the processes are executed by assigning threads controlled by Thread class, then the process enters into the PEC through a class named as Ready_queue; resources are allocated to each process, if the PEC is busy and resources are not allocated then the processes have to wait in the suspended queue controlled by Suspended_queue class; if any process waits for a long time for resources after sometime, time out condition occurs and either older process will die or rollback and then finally deadlock is removed and then process is synchronized and enters into PEC which is responsible for execution of the process and after being executed the resultant sends back to the memory controlled by Memory class.

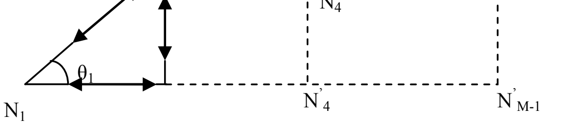


Figure 3. A static step topology.

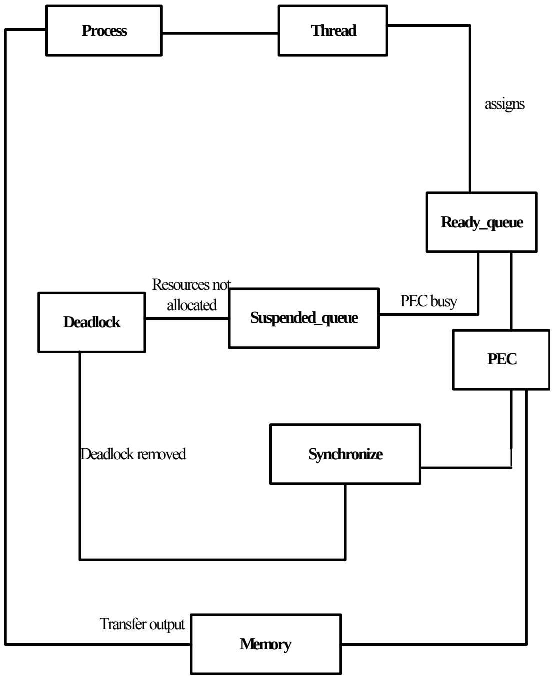


Figure 4. Class diagram for deadlock detection.

5. Deadlock Detection and Avoidance

On the basis of the above UML class model, the following cases arise on the NKN network in which devices are connected through static step topology as shown the **Figure 3**:

Case 1: Reflexive Situation of Deadlock

This situation of deadlock arises on the single node. Let us consider a single node N_1 as shown in **Figure 3**, when a process P1 on node N_1 needs resources which are executing on other nodes then the process P1 and other processes in queue on N_1 will become in the deadlock situation. This situation is represented as reflexive situation and the avoidance method from deadlock is given below:

Avoidance: The reflexive situation of the deadlock can be avoided through the mutual exclusion of the processes in which resources are allocated to the processes as per the incoming request of the resources then at a time only one process can execute inside the critical section and after finishing the execution of process resources are released and next process will enter inside the critical section then in this situation no deadlock will occur. Processes synchronization will occur inside the critical section.

Case 2: Symmetric Situation of Deadlock

The symmetric situation of deadlock arises when two nodes N_1 and N_2 are participating for execution of the processes and connected under step topology as shown in **Figure 3** and they communicate through message passing technique, then following two cases arise:

1) When multiple processes are going to execute on node N_1 as per the scheduling algorithm then if resources are allocated to the younger processes then older processes which are waiting from a long time for resources will die and deadlock will occur.

2) If the process on the either node N_1 or N_2 will wait for a long time for resources then the processes will die (time out) due to wait and die condition after some time and deadlock will occur.

Avoidance: In the first case, the deadlock situation can be handled by making a priority queue of the processes. If the older processes are rolled back by giving priority over younger processes then deadlock can be avoided.

In the second case, if the resources are available, resources which are allocated and resources which will be released by other processes are known to the other processes which are trying to execute on either N_1 or N_2 and they communicate about the resources as said above through message passing technique, then deadlock will not occur.

6. Conclusion and Future Scope of Work

From the above it is concluded that the Unified Modeling Language (UML) is used to construct a model for the deadlock situation occurring in the static step topology. Two cases in the form of reflexive and symmetric deadlock are discussed and corresponding avoidance methods are explained. The present work can be extended for validation of the proposed UML model through Markov chain or Finite State methods.

7. Acknowledgements

Our thanks are due to University Grants Commission (UGC), New Delhi, India for providing financial assistance to carry out this work.

REFERENCES

[1] A. Siberschatz and P. B. Galvin, "Operating Systems Concepts," 5th Edition, John Wiley and Sons, Inc., Hoboken, 2008.

[2] A. S. Tanenbaum, "Distributed Operating Systems," Prentice Hall, Upper Saddle River, 1995.

[3] N. Minar, "Distributed System Topologies Part 1 and Part 2," 2012. http://www.open2p.com/lpt/a/1461

[4] K. Hwang, "Advanced Computer Architecture", McGraw-Hill Series in Computer Engineering, McGraw-Hill Inc., New York City, 1993.

[5] M. Milenkovic, "Operating Systems: Concepts and Design," Tata McGraw-Hill, Noida, 1997.

[6] L. Lamport, "Time, Clocks and Ordering of Events in a Distributed System," *Communications of ACM*, Vol. 21, No. 7, 1978, pp. 558-565. doi:10.1145/359545.359563

[7] G. Ricart and A. Agrawala, "An Optimal Algorithm for Mutual Exclusion in Computer Networks," *Communications of the ACM*, Vol. 24, No. 1, 1991, pp. 9-17.

[8] M. Maekawa, "A N Algorithm for Mutual Exclusion in Decentralized Systems," *ACM Transactions on Computer Systems*, Vol. 3, No. 2, 1985, pp. 145-159. doi:10.1145/214438.214445

[9] D. Agrawal and A. El Abbadi, "An Efficient and Fault Tolerant Solution for Distributed Mutual Exclusion," *ACM Transactions on Computer Systems*, Vol. 9, No. 1, 1991, pp. 1-20. doi:10.1145/103727.103728

[10] I. Suzuki and T. Kasami, "A Distributed Mutual Exclusion Algorithm," *ACM Transactions on Computer Systems*, Vol. 3, No. 4, 1985, pp. 344-349. doi:10.1145/6110.214406

[11] H. Gomma, "Designing Concurrent, Distributed, and Real-Time Applications with UML," *Proceedings of the 23rd International Conference on Software Engineering* (*ICSE*'01), Toronto, 12-19 May 2001, pp. 2-15.

[12] S. Pllana and T. Fahringer, "On Customizing the UML for Modeling Performance Oriented Applications," *The Unified Modeling Language*, Springer-Verlag, Dresden, Germany, 2002.

[13] S. Pllana and T. Fahringer, "UML Based Modeling of Performance Oriented Parallel and Distributed Applications", *Winter Simulation Conference*, Washington, 8-11 December 2002, pp. 497-505. doi:10.1109/WSC.2002.1172922

[14] V. Saxena and D. Arora, "UML Modeling of a Protocol for Establishing Mutual Exclusion in Distributed Computer System," *International Journal of Computer Science and Network Security*, Vol. 8, No. 6, 2008, pp. 227-235.

[15] V. Saxena and T. Zaidi, "Step Topology for Static Interconnection of Computer Systems under Distributed Environment," *World Conference of Information Technology*, Barcelona, 14-17 November 2012, in Press.

[16] V. Saxena and T. Zaidi, "Modeling Aspects for Step and Bus Topologies under Distributed Computing System," *International Journal of Computer Applications*, Vol. 53, No. 6, 2012, pp. 20-24.

[17] V. Saxena and T. Zaidi, "National knowledge Network vs. Information Communication Technology," *Proceedings of National Seminar in Information Technology*, Muz-

zafarfur, 11-12 February 2012, pp. 24-28.

[18] V. Saxena and T. Zaidi, "Modifications in Lamport Algorithm for Distributed Computing System," *International Journal of Computer Applications*, Vol. 53, No. 6, 2012, pp. 28-35.