Detection of Transmission Data Error in Distributed Computer System Through UML

Vipin Saxena[1] Deepak Arora[2]

Abstract

In a distributed computer system, the communication is done by using point-to-point or broadcasting technique among the processes running concurrently. Any process running in a distributed system can access the any resource or communicate with another process that can reside locally or on a remote site. There can be error occurred in data or message bit streams or it may be altered during the transmission from one machine to other machine over the network.

Cyclic Redundancy Code (CRC) is a well-established method for ensuring the data integrity during data transmission. In the present paper, authors have proposed a UML model for detection of transmission data error in distributed computer system. The purpose to design this model is to minimize the data transmission error over the network. UML class, sequence and activity diagrams are presented for the computation of transmission error. The performance of this model is also reported in this paper through a case study.

[1]F. Reader, Department of Computer Science, B. B. Ambedkar University (A Central University), Vidya Vihar, Raebareli Road, Lucknow – 226025, INDIA. Email: vsax1@rediffmail.com

[2]S. Department of Computer Science, B. B. Ambedkar University (A Central University), Vidya Vihar, Raebareli Road, Lucknow - 226025, INDIA Email: deepakarorainbox@gmail.com

Keywords : UML Class Diagram, Sequence Diagram, Data Error, Cyclic Redundancy Code, Polynomial

1. Introduction

UML is a powerful modeling language used to represent visually the software and hardware design problems. The diagrams used in the, Unified Modeling Language are proposed by Booch et al. [2-3]. The object-oriented concepts related to the unified modeling language are explained by the OMG Group [6-7]. Recently the data transferred techniques are shifted from centralized computer system to distributed computing system. The Distributed Computing System allows more than one process, run concurrently. For process synchronization purpose any process can send synchronization messages to other process in the distributed system. The error can occurred in the message due to noisy channels, less reliable storage media or any other cause. One can define the error detection as the ability to detect the presence of errors and error correction as add-on ability to reconstruct the original, error-free data.

A limited research papers are available for the UML modeling related to the distributed computing system. In this direction lot of research work done by Pllana, S., T. Fahringer [9-10] and proposed the performance of the parallel and distributed applications through the unified modeling language. Recently Saxena & Arora [11-12] proposed performance oriented UML models for distributed computer architecture research problems. In [11] they described the object oriented distributed In the

architecture system for distributed computing, while in [12] they proposed a new mutual exclusion algorithm for distributed system using bi-directional ring technology. Cyclic Redundancy code [4, 8, 13] is a commonly known method for detecting the error.

In the present paper authors have done the UML modeling [2, 3, 5] of cyclic redundancy check, through which one can better understand the hidden design aspect of this method, while implementing it to a software component for any industry application development.

The performance of the model has also been evaluated through a performance evaluation tool and found satisfactory. A case study based on the model designed, is also discussed in this paper.

2. Distributed Computer Syatem

One can consider any huge scientific problem as a collection of tasks that has to be performed for determining the solution of that problem. Now these tasks can be assigned to more than one autonomous computer or node, which is responsible for the execution of the assigned task at their own end. In this way one can achieve a form of distributive parallelism. Later on after completion of the assigned tasks, one can combine the results from different nodes, and get the complete result of the scientific problem. Nodes involve in the true distributed computing doesn't share memory or clock, connected by a communication network [14]. For execution purpose, nodes have to communicate each other by using message-passing mechanism. It results in less throughput and turnaround time, regarding overall execution of any program or problem but simultaneously reliability, availability and security are the major strength of distributed computing. A distributed computer system is shown below in figure 1.

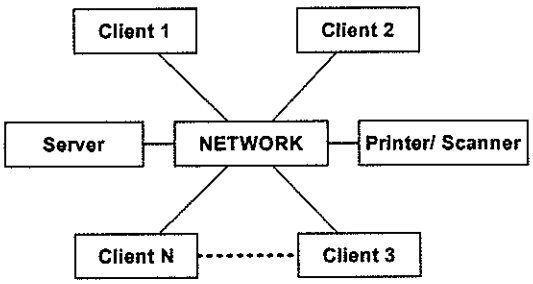


Figure 1 : Distributed Computer System

In the figure1, for execution client 1 can also use the resources, available with client 2. These resources can be shared in a synchronized manner through message passing technique [12]. Therefore it is must that transmission between two should be error free. For this CRC method is a most popular approach. In a cyclic redundancy code, a bit stream of 0 and 1 represents polynomial coefficients. The coefficient list of any polynomial having n term can be represented by n bit frame starting from x^0 to x^{n-1} [13]. In this case the degree of polynomial will become n-1. For example one can represent the polynomial with coefficient $x^6+x^2+x^0$ in bit representation as 1000101. In the polynomial arithmetic like addition and borrow, carries are not taken into consideration. In this way these operation become identical to Exclusive OR (XOR) operation. The division (modulo-2 division) operation remains same as in the binary, except that after division, subtraction will be performed as XOR operation.

In this method, sender and receiver have to agree for the usage of similar generator polynomial, in which the lower and the higher bit must be 1. For computation of CRC for any frame, it is also necessary that frame of k bits should be greater in size than generator polynomial (divisor) agreed by sender and receiver. The key idea is to create the Transmitted frame, is divisible by generator polynomial, by appending a CRC at the end of the frame.

When this Transmitted frame is reached to the receiver, it divides the frame by the generator polynomial. If this division yields the remainder as 0, then there is no error occurred during transmission of this frame otherwise there is an error. A CRC code having n check bits can detect burst error of length not more than n. Also a valid CRC [4], have exactly one bit less than the divisor and after appending to the end of frame, the resultant frame should be divisible by the divisor. In this paper this concept for error free transmission of data is presented through the Unified Modeling Language. The following algorithm [4, 13] is used for detection of data error:

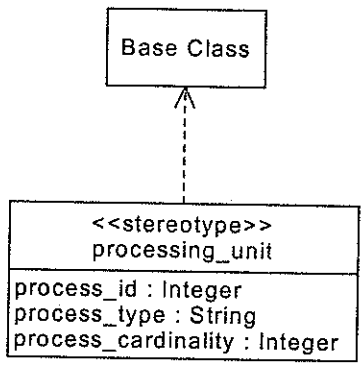
1. A string of n 0's is appended to the end of the frame, which is supposed to be transmitted. Here n should be equal to the degree of predetermined divisor.
2. Now divide the resultant frame by a predetermined divisor, using a process called binary division.
3. The remainder got, after the division is CRC.
4. Now replace all the 0's appended at the end of frame, by this generated CRC. This will create the Transmitted frame for transmission.
5. After receiving the frame, receiver will divide it again by the same predetermined divisor. After division if receiver gets no remainder then received frame contains no error otherwise frame is erroneous.

3. UML Desifn for Detection Of Transmission Error

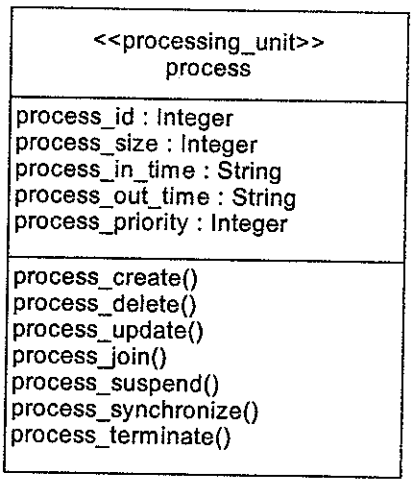
A. Process Definition

In this paper authors have assumed that processes are running concurrently in a distributed computing environment. One can define the process, which is going to be executed in the distributed environment. The process can be classified as a macro, subprogram, subroutine or block of code etc., which has an identification number called as process_id, processing unit as Meta Class, shown in figure 2(a).

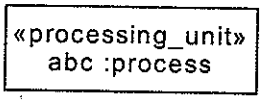
The UML class diagram of process consisting of the number of attributes and methods is shown in figure 2(b). The instance and multiple instances of the process through the objects are also represented as in figure 2(c) & 2(d), respectively [10, 11].



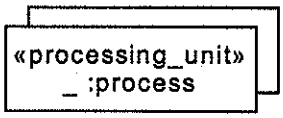
(a) UML Class Diagram of Stereotyped Processing Unit



(b) UML Class Diagram of Process



(c) Instance of UML Class Process

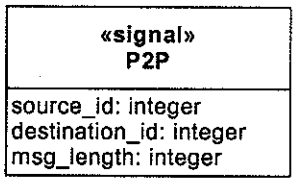


(d) Multiple Instances of UML Class Process

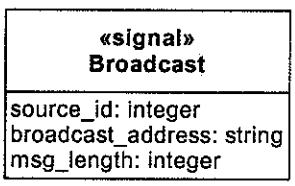
Figure 2 : UML Class and Instances Representation of Process

B. Communication in Distributed Computing System

The communication among the processes in the distributed environment can be handled by the synchronization technique if the common resources are available at the client computer systems as shown in figure 1. The communication can be of the two types; one approach is point_to_point (P2P) protocol method and other technique is broadcast protocol method [9]. UML class diagrams P2P and broadcast signals are given in figure 3.



(a) UML Class Diagram of P2P Signal



(b) UML Class Diagram of Broadcast Signal

Figure 3 : UML Class Diagram for P2P & Broadcast Signals

C. UML Class Model for Detection of Transmission Data Error

The class diagram for the CRC computation is shown below in figure 4. One can see that CRC_Compute class has four components, Binary_Divider, Binary_Subtractor, Divisor (generator polynomial) and Dividend (data frame). CRC_Checker and CRC_Generator are the inherited classes, responsible for CRC checking at the receiver end and CRC generation at sender end respectively. CRC_Generator class will also interact with the Binary_Adder class for preparing the final Transmitted frame for being sent from the sender end. CRC_Generator class will generate the CRC code and submit it to the Binary_Adder class, which is further responsible for replacing the corresponding least significant bits (LSB) in the frame with CRC generated, and to generate the final Transmitted frame. The Divisor class is interacting with the polynomial class, which contains bits stream corresponding to the polynomial coefficients, agreed on, by sender and the receiver. Dividend class, will also interacting with the Bit_Shifter class for left or right bit shift operation. The message class will accumulate all the send/receive operations of synchronization and data messages and will interact directly with the communication network and Binary_Adder class.

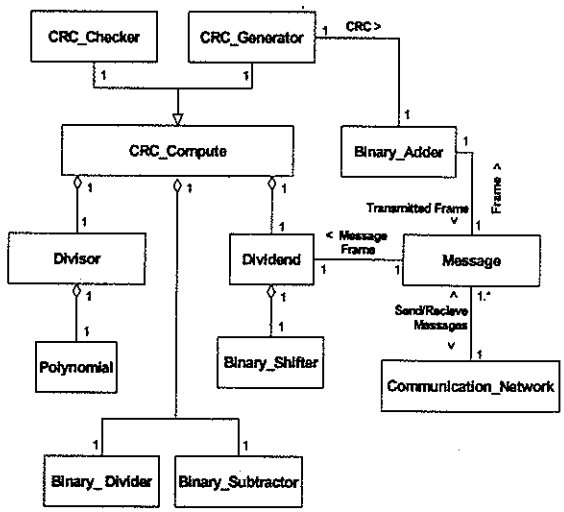


Figure 4 : UML Class Diagram for CRC Computation

D. UML Sequence Diagram for Detection of Transmission Data Error

The sequence diagram for CRC generation is shown above in figure 5. One can easily notify the sequence of messages gets passed, while generating the CRC code and final Transmitted frame. Firstly CRC_Generator will communicate with the Divisor for getting binary representation of polynomial coefficient. Now CRC_Generator will communicate with the Dividend. Now Dividend will interact with the Polynomial for getting its degree and message. Now Dividend will append 0's (Equals to the polynomial's degree) at the least significant bit (LSB) end of message frame. Now CRC_Generator will start the division. To perform the division operation, Polynomial as Divisor and message frame appended with 0's as Dividend, can be done with the help of Binary_Divider, Binary_Subtractor and Binary_Shifter. Complete Dividend can be considered as different segments. Here segments are equivalent to the remainder obtained after one division step, shifted right by one bit. When the division completes, the remainder is CRC code that should be replaced in the place of 0's in the message frame, which is Transmitted frame. Now this Transmitted Frame is ready to send.

The sequence diagram for CRC checking is shown above in figure 6. In this diagram one can see that when Transmitted frame reaches at the receiver end, the CRC_Checker will start the checking of the arrived frame whether it contains error or not. After getting the divisor and dividend frame CRC_Checker will start division process. After division has been completed, the remainder bits are submitted back to the CRC_Checker by the Binary_Divider. Here CRC_Checker will check the remainder bits whether all the remainder bits are zero or not. If zero then declare the frame is error free otherwise there must be an error in frame.

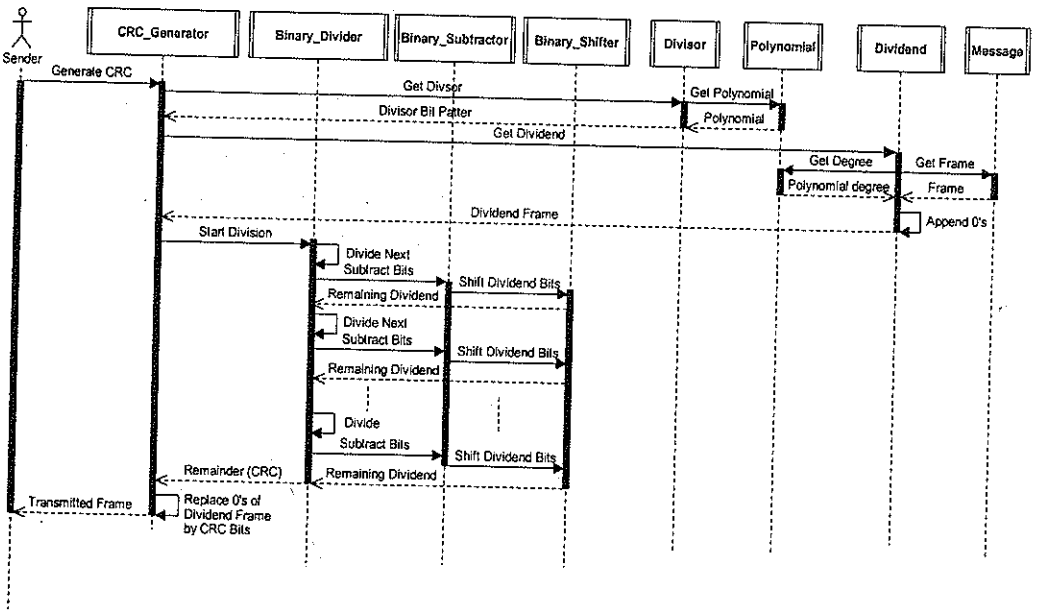


Figure 5: UML Sequence Diagram for CRC Generation at Sender End

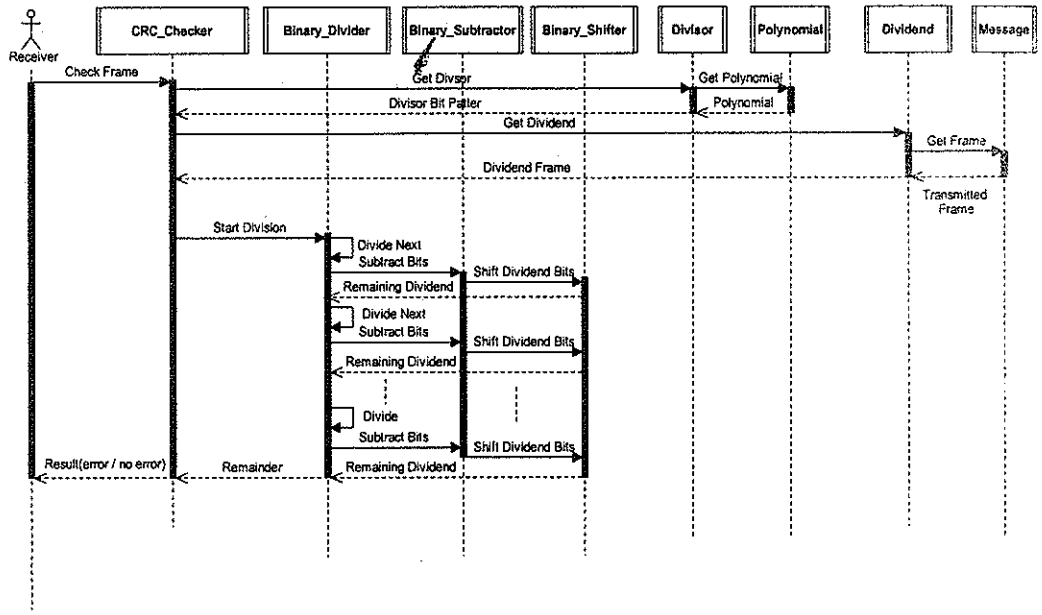


Figure 6: UML Sequence Diagram for Checking Transmitted Frame at Receiver End

E. UML Activity Diagram

A UML activity diagram is shown above in figure 7. One can see that what are the important activities and decision taken by the system while generating and checking the CRC code and Transmitted frame at the sender and receiver end, respectively. After loading the polynomial, divisor generation will start.

After loading the message frame, if it is sender end then CRC code generation will start otherwise in the case of receiver end checking of Transmitted frame would be started, whether it is error free or not. The activity append zero frame will further includes activities calculate degree of polynomial and count number of zeros to be appended. Similarly divide frame activity will further include bits subtraction and bits shifting like activities.

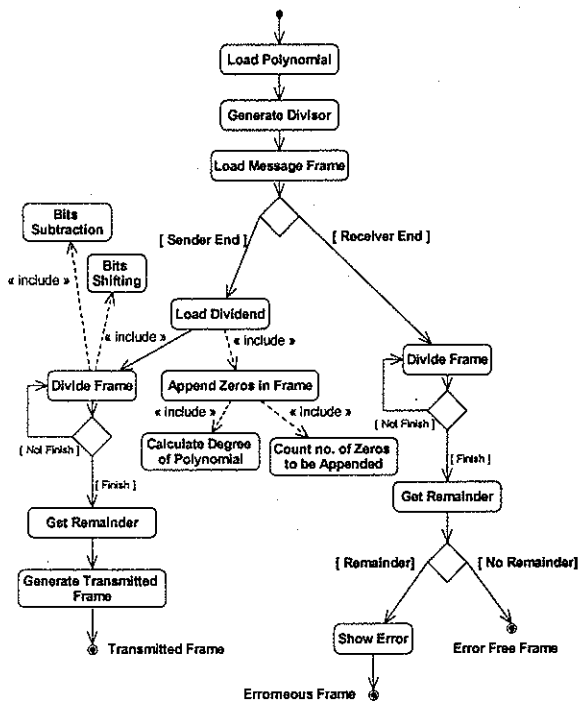


Figure 7: UML Activity Diagram for CRC Computation

4. Computation of Detection of Data Error in Frame - A Case Study

In the distributed computing scenario, the above model is tested, by considering the following set of values. In this one can see the role of individual classes given in the model in the CRC code generation at the sender and checking of the Transmitted frame at the receiver end as well.

Let us consider the following:

Message (8 bit Frame): 10010100

Generator Polynomial agreed on: $x^5 + x^4 + x^2 + 1$ (CRC-5-ITU Standard) [15]

Divisor (Binary representation of generator polynomial): 110101

Dividend (Frame after appending 0's by Binary_Adder): 10010100 00000

The following Table 1 shows the computation of CRC generator & Transmitted frame, generated by CRC_Generator by replacing 0's, by CRC code, in the message frame with the help of Binary_Adder.

After division the CRC 01110 is generated. Now to generate the Transmitted frame, these CRC bits have to be appended at the end of the frame and the resultant frame is produced as 1001010001110. Now this frame is transmitted to the receiver. At the receiver end CRC_Checker, will check this Transmitted frame, by dividing it by the predetermined Divisor. At the receiver end, one must consider the following two cases, if there is no error occurred during transmission of this frame and another for the possible error occurred in the frame. Table 2 and Table 3 given below are showing these scenarios.

Here one can see that, after division if CRC_Checker gets all the remainder bits zero after division of received frame, shows that data is error free.If CRC_Checker finds any remainder after division of received frame, then there must be an error occurred during transmission.

CRC method of error detection is heavily dependent on the polynomial (divisor) chosen by both sender and the receiver. The polynomial should be divisible by x and at the same time it should not be divisible by x+1. This method detects the bust errors of length, equal to the degree of polynomial chosen in odd number of bits frame.

Table 1: Computation Of Crc_Generator At Sender End

Step	Divisor	Dividend	Binary_Divider (Quotient)	Binary_Subtractor (Remainder)	Binary_Shifter (Left Shift Remainder by 1 Bit) ▼ Next Dividend
1	110101	1001010000000	1	0100000000000	100000000000
2	110101	100000000000	1	010101000000	10101000000
3	110101	10101000000	1	01111100000	1111100000
4	110101	1111100000	1	0010110000	010110000
5	110101	010110000	0	010110000	10110000
6	110101	10110000	1	01100100	1100100
7	110101	1100100	1	0001110	001110
8	110101	001110	0	001110	01110 (CRC)

Case 1: Message (Received Frame) : 1001010001110

Table 2 : Crc_Checker At Receiver End (No Error In Data)

Step	Divisor	Dividend	Binary_Divider (Quotient)	Binary_Subtractor (Remainder)	Binary_Shifter (Left Shift Remainder by 1 Bit) ▼ Next Dividend
1	110101	1001010001110	1	0100000001110	100000001110
2	110101	100000001110	1	010101001110	10101001110
3	110101	10101001110	1	01111101110	1111101110
4	110101	1111101110	1	0010111110	010111110
5	110101	010111110	0	010111110	10111110
6	110101	10111110	1	01101010	1101010
7	110101	1101010	1	0000000	000000
8	110101	000000	0	000000	00000 (No Error in Frame)

Case 2 : Message (Received Frame) : 1001110001110

Table 3 : Crc_Checker At Receiver End (Error In Data)

Step	Divisor	Dividend	Binary_Divider (Quotient)	Binary_Subtractor (Remainder)	Binary_Shifter (Left Shift Remainder by 1 Bit) ▼ Next Dividend
1	110101	1001110001110	1	0100100001110	100100001110
2	110101	100100001110	1	000001001110	00001001110
3	110101	00001001110	0	00001001110	0001001110
4	110101	0001001110	0	0001001110	001001110
5	110101	001001110	0	001001110	01001110
6	110101	01001110	0	01001110	1001110
7	110101	1001110	1	0100100	100100
8	110101	100100	1	010001	10001 (Error in Frame)

5. Concluding Remarks

From the above it is concluded that the CRC is very effective error detection method for the data transmission error in any distributed computer system. Also UML is very popular and dominating visual modeling language widely used for modeling the complex software and hardware problems. In this paper authors have presented the UML design of error detection method, through which one can easily understand the complexities behind the design of any error detention method for the distributed computing scenario. Also authors have evaluated the performance of the model, found satisfactory. With the help of this UML modeling one can easily develop the application based on error detection for any distributed computing environment. This work can be further extended for the real time environment using UML.

References

[1] Alhir. S.S, "*UML in a Nutshell*", O'Reilly, 1998

[2] Booch. G, Rumbaugh. J and Jacobson. I, *"The Unified Modeling Language User Guide"*, Addison Wesley, Reading, MA 1999.

[3] Booch. G, *"Object Oriented Analysis & Design With Applications"*, Second Edition, Addison Wesley, 1994.

[4] Forouzan. B. A, *"Data Communications and Networking"*, third edition, Tata McGraw-Hill, 2004

[5] Fowler. M and Scott. K, *"UML Distilled: Applying the Standard Object Modeling Language"*, Addison-Wesley, 1997.

[6] OMG, *"Unified Modeling Language Specification"*, Available online via http://www.omg.org.

[7] OMG, *"XML Metadata Interchange (XMI) Specification"*. Available online via http://www.omg.org.

[8] Peterson.W.W and Brown. D.T, *" Cyclic Codes for Error Detection"*, Proceedings of the IRE, Vol. 49, Issue 1, PP. 228 – 235, Jan. 1961

[9] Pllana. S, Fahringer. T, *"On Customizing the UML for Modeling Performance Oriented Applications In <<UML>>"*, Model Engineering Concepts and Tools, 2002, Springer-Verlag., Dresden, Germany.

[10] Pllana. S, Fahringer. T, *"UML Based Modeling of Performance Oriented Parallel and Distributed Applications"*, Winter Simulation Conference, 2002.

[11] Saxena. V, Arora. D , Ahmad .S, *"Object Oriented Distributed Architecture System through UML"*, conference IEEE, International Conference on Advances in Computer Vision and Information Technology, ACVIT-07, ISBN 978-81-89866-74-7, PP.305-310, 2007,.

[12] Saxena.V, Arora . D, *"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 PP. 227-235, 2008.

[13] Tennbaum . A, *"An Introduction to Computer Network"*, Prentice Hall, 1997.

[14] Tanenbaum. A.S and Steen. M.V, *"Distributed Systems: Principles and Paradigms (2nd Edition)"*, Prentice Hall, 2006.

[15] http://en.wikipedia.org/wiki/Cyclic_redundancy_check

Author's Biography

Dr. Vipin Saxena, He is a Reader & Head, Dept. of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, India. He got his M.Phil. Degree in Computer Application in 1991 & Ph.D. Degree work on Scientific Computing from University of Roorkee (renamed as Indian Institute of Technology, India) in 1997. He has more than 12 years teaching experience and 16 years research experience in the field of Scientific Computing & Software Engineering. Currently he is proposing software designs by the use of Unified Modeling Language for the various research problems related to the Software Domains & Advanced Computer Architecture. He has published more than 55 International and National publications.

Deepak Arora, He is a Research Scholar, Dept. of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, India. He got his Master Degree of Computer Applications in 2003 and M.Phil. Degree in Computer Science in the year 2006. Currently he is actively engaged in the research work on Distributed Computing Systems through the Unified Modeling Language. Ha has produced several outstanding publications on Distributed Computing Systems.