International Journal of Computer Theory and Engineering, Vol. 1, No. 4, October2009
1793-8201

UML Design for Performance Evaluation of Object Oriented Programs on Dual Core Processors

Dr. Vipin Saxena and Manish Shrivastava

***Abstract*— In today's scenario, high performance computing is needed to solve the complex scientific problems. In this regards the Multi-core technology is one of the major technologies. Intel's Dual Core processors improve the performance of applications by executing multiple programs at a time. The objective of the present paper is to evaluate the performance of well known Object-oriented programming languages namely Visual C#, Visual C++ and Java on Intel's Dual Core processors. To check the performance of various programs on Dual Core processors, a common program is developed in these three languages. The run time of each program is measured for quantitative comparison of performance of these languages. Before evaluating the performance of these processors, an efficient UML model is designed for the program execution. The UML class and sequence diagrams are designed and comparison is also made between the performance of two selected Dual Core processors namely Dual Core and Core 2 Duo.**

***Index Terms*—Object-oriented programs, Intel Dual Core processor, Intel Core 2 Duo Processor, UML class diagram, UML sequence diagram**

VI. Introduction

In Object-oriented software development, the Unified Modeling Language (UML) is one of the most powerful modeling techniques. It is a set of diagrammatical notations and is currently standardized and supported by the Object Management Group (OMG). The details and good description of the notations are given in Alhir [1], and Booch et al. [2].

The UML can also be used in hardware or system architecture modeling. It also provides extension mechanisms using stereotypes and profiles which can be applied in more domain specific modeling of a system.

The applications of UML design in computer architecture modeling have been described in some research papers. Gomma [3] has developed a UML based Concurrent Object Modeling and Architectural Design Method for designing real-time and distributed applications.

Manuscript received May 12, 2009.

Dr. Vipin Saxena is an Associate professor and Ex-Head of Department of Computer Science, B.B. Ambedkar University (A Central University), Vidya Vihar Rae Bareilly Road, Lucknow U.P. 226025, India, phone: +91-9452372550; fax: +91-522-2440821

Manish Shrivastava (Corresponding Author), is a research student in Department of Computer Science, B.B. Ambedkar University (A Central University), Lucknow, India, phone: +919453847114

The UML based modeling of parallel and distributed systems for performance oriented applications, is described by Pllana, S. and Fahringer, T. [4]. Saxena et al. [5] proposed the UML model for the Multiplex system for the processes which are executing in distributed environment. Pustina Lukas et al. [6] presented a UML based modeling methodology of specifying processor details of ARM. In this paper, UML diagrams are used to model the system architecture and timing behavior. In a recent paper by Saxena and Raj [7], UML modeling has been done for instruction pipeline design and its performance evaluation. In their paper, Fateh Boutekkouk et al. [8] presented a new UML-based methodology for embedded applications design and architectural modeling including the CPU model, the Memory model etc. using stereotypes. An estimation technique of performance is also proposed.

In available literature, some work was found in comparing various programming languages, but they are mostly based on their features, technical similarities, differences, and capabilities. There are very few papers available on quantitative performance comparison of Object-oriented programming languages. Henderson Robert and Zorn Benjamin [9] compared the run-time efficiency and compilation time of language implementations of four modern programming languages that support Object-oriented programming (Oberon-2, Modula-3, Sather and Self), and compared them with C++ also.

Glyph Lefkowitz [10] performed a comparison of execution speed between Java and Python by running some test-cases on Linux plateform. Cowell-Shah [11] discussed a small-scale benchmark test run on nine modern computer languages and their variants. All tests took place on a Pentium 4-based computer (notebook) running Windows XP. Recently Saxena and Arora [12] reported a performance evaluation for Object-oriented software systems using VC++ and C#. The evaluation is done on nodes, equipped with Pentium D and Core 2 Duo processor technologies.

In this paper, the architectural modeling of Intel Core micro-architecture is performed using UML. The UML stereotypes for process and execution cores are defined. UML class and sequence diagrams are designed for modeling of process execution. A common program has been developed in three Object-oriented programming languages namely Visual C++, Visual C# and Java. The programs were executed on Intel Dual Core and Core 2 Duo processor. A comparison of execution time of the program is reported for performance evaluation.

VII. BACKGOUND

A. Process Definition

A process is a block of instructions of a program, which are executed by a processor. For defining the process, we need to first define a processing unit. Using UML, a processing unit can be modeled using a stereotype. Stereotypes are used to define some specialized modeling elements based on core UML base classes. Fig. 1a shows the UML stereotype definition of a processing unit and Fig. 1b shows the class diagram for representing a process. Fig. 1c shows the single and multiple instances of process.

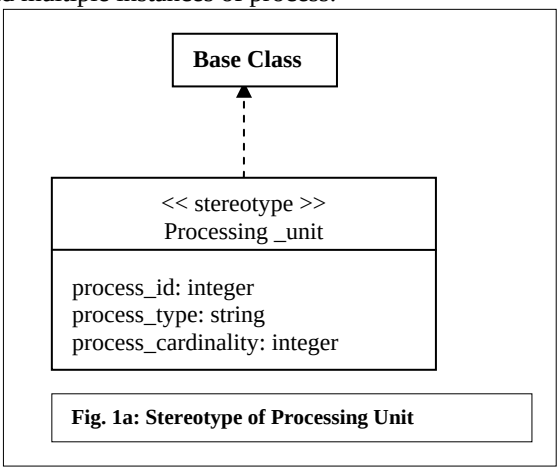

Fig. 1a: Stereotype of Processing Unit

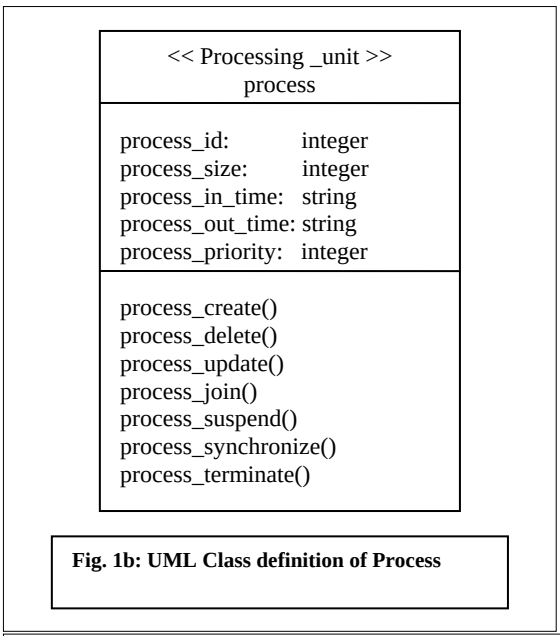

Fig. 1b: UML Class definition of Process

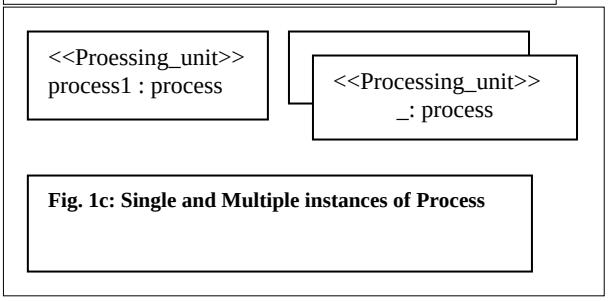

Fig. 1c: Single and Multiple instances of Process

B. Intel Core micro-architecture

Intel's Dual Core processors are based on Intel Core micro-architecture. The Dual Core layout uses CMP (i.e. core multi processor) technology, where two or more CPUs (known as Cores) are fabricated together on one chip along with dual L2 caches. With Dual Core architecture, processors move blocks of many hundreds instructions into cache before executing them in blocks of four or more at a time. The main purpose is to execute even the most complex instructions in one clock tick.

The Intel's Core micro-architecture technology provides more efficient decoding stages, execution units, caches, and buses for increasing the processing capacity, reducing latency and thus achieving high performance. The architectural details of Dual Core are described in [13] and [14].

The architectural modeling of Intel Core micro-architecture is performed using UML. The UML stereotypes for the execution cores are defined. Fig. 2a shows the UML stereotype definition of Execution core. Fig. 2b shows the class diagram for representing a core and the Fig. 2c shows the single and multiple instances of core.

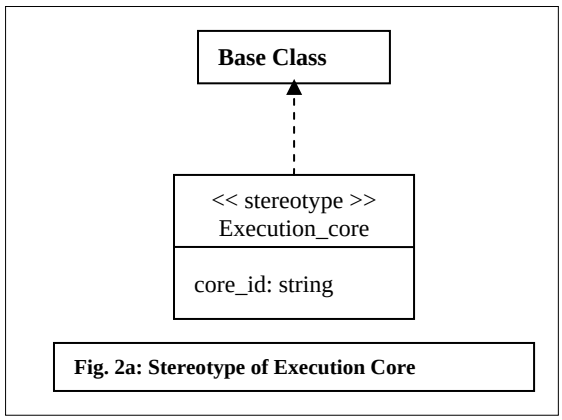

Fig. 2a: Stereotype of Execution Core

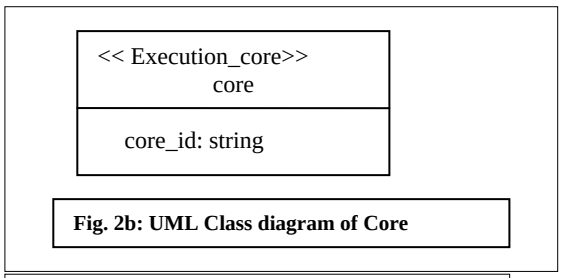

Fig. 2b: UML Class diagram of Core

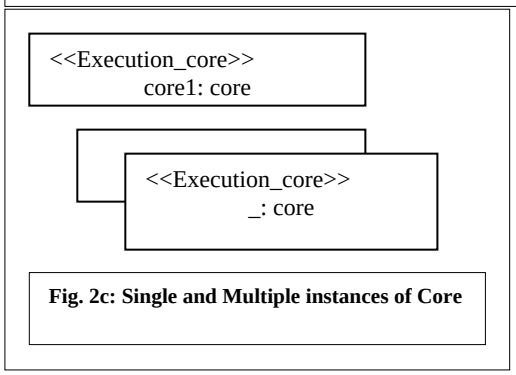

Fig. 2c: Single and Multiple instances of Core

C. Object-oriented Programming

Object-oriented design and programming has become the most prominent technique in today's software development. There are many significant improvements in modeling and building complex systems using Object-oriented approach. It provides many benefits such as encapsulation, polymorphism, inheritance, reusability and extensibility. There are many

Object-oriented languages for commercial software development. Among these languages, the three languages namely Visual C++, Visual C# and Java are most popular and powerful in today's programming environment. All these programming languages support all the features of an Object-oriented language. Visual C++ and Visual C# are developed by Microsoft and are available in Visual Studio. Java was developed by Sun Microsystems and can be executed at any platform using Java virtual machine.

VIII. UML Architectural Modeling

A. UML Class Diagram for Processor Architecture

The Fig. 3 shows the complete architectural model of Dual Core processor architecture. The class Process is directly interacting with the class Process_Execution_Controller (PEC), which is fully responsible for the execution of the assigned task. The PEC is controlling the processes by message exchanging between the classes Processor and Memory. The Processor class contains two cores, i.e. Core1 and Core2 and each core contains many components responsible for process execution as shown in the figure. The class diagram of the entire memory unit is also shown in the figure. Here class L2_Cache is shared between two cores and caches instructions through the class I_ Cache whereas the class D_Cache is responsible for caching the data, which is a sub class of L1_Cache.

B. UML Sequence Diagram for Process Execution

The UML sequence diagram for process execution inside a core is shown in Fig. 4. Here the messages are exchanged among various class objects like Process, Process_Ececution_Controller, L2_Cache and L1_Cache are shown. Instructions are fetched from L2_Cache, decoded into the executable micro operations. The data are loaded from L1_Cache. After execution, the results of these micro operations are passed to the Retirement_Unit. It takes the results, reordered them and rebuilds the final results.

IX. Experimental Study

A fundamental performance metric of any computer system is the time required to execute a given application program. During the performance testing of a developed program, programmers measure program execution time. The programmers may measure the execution time of an entire program or only parts of a program.

The experimental results are obtained by executing a common code written in each programming language. A sample code for displaying a message repetitively inside a loop is taken to evaluate the performance. These sample codes were executed on two systems having different processor architectures. The Visual C++ and Visual C# programs are developed as windows applications and executed under Visual studio 2008 on Microsoft.Net framework v3.5. The Java program was developed for console application and executed using JDK1.5.0_18. We measured the execution time spent in a critical loop of the program. The architectural detail of the systems is given in table 1 below. All the experimental results are averaged from 5 different runs. Table 2 shows the execution time computed in milliseconds on Dual Core processor and the table 3 shows the execution time computed in milliseconds on Core 2 Duo, for which the experimental study is performed. Table 4 shows the comparison between average execution times.

Fig. 5a and 5b clearly display above results in the form of graph as a performance comparison of Dual Core and Core 2 Duo processor in term of execution time of programming codes having 1000 and 10000 lines.

X. Results and Discussions

Based on the experimental results, it was found that Visual C++ is more efficient Object-oriented programming language in comparison to Visual C# and Java. It is clear from the above tables that the execution time is lesser in case of Visual C++ in comparison to Visual C# and Java. It is also observed that the execution time on Core 2 Duo processor based system is less than the Dual Core processor based system as per the specifications mentioned above.

XI. Conclusion

It is concluded that UML is a powerful modeling language for formal specifications of hardware systems and various research problems. In present paper, the performance of two processors namely Intel Dual Core and Core 2 Duo is observed by taking different lines of codes, which are developed in three Object-oriented programming languages namely Visual C++, Visual C# and Java. Results showed that the Intel Core 2 Duo had the best performance for a variety of lines of codes as compared to Intel Dual Core as per the given specifications. It is also found that Visual C++ takes less execution time as compared to Visual C# and Java over similar processor architectures. It is also observed that the performance of Core 2 Duo processor is better than the Dual Core and therefore, recommended for long computations.

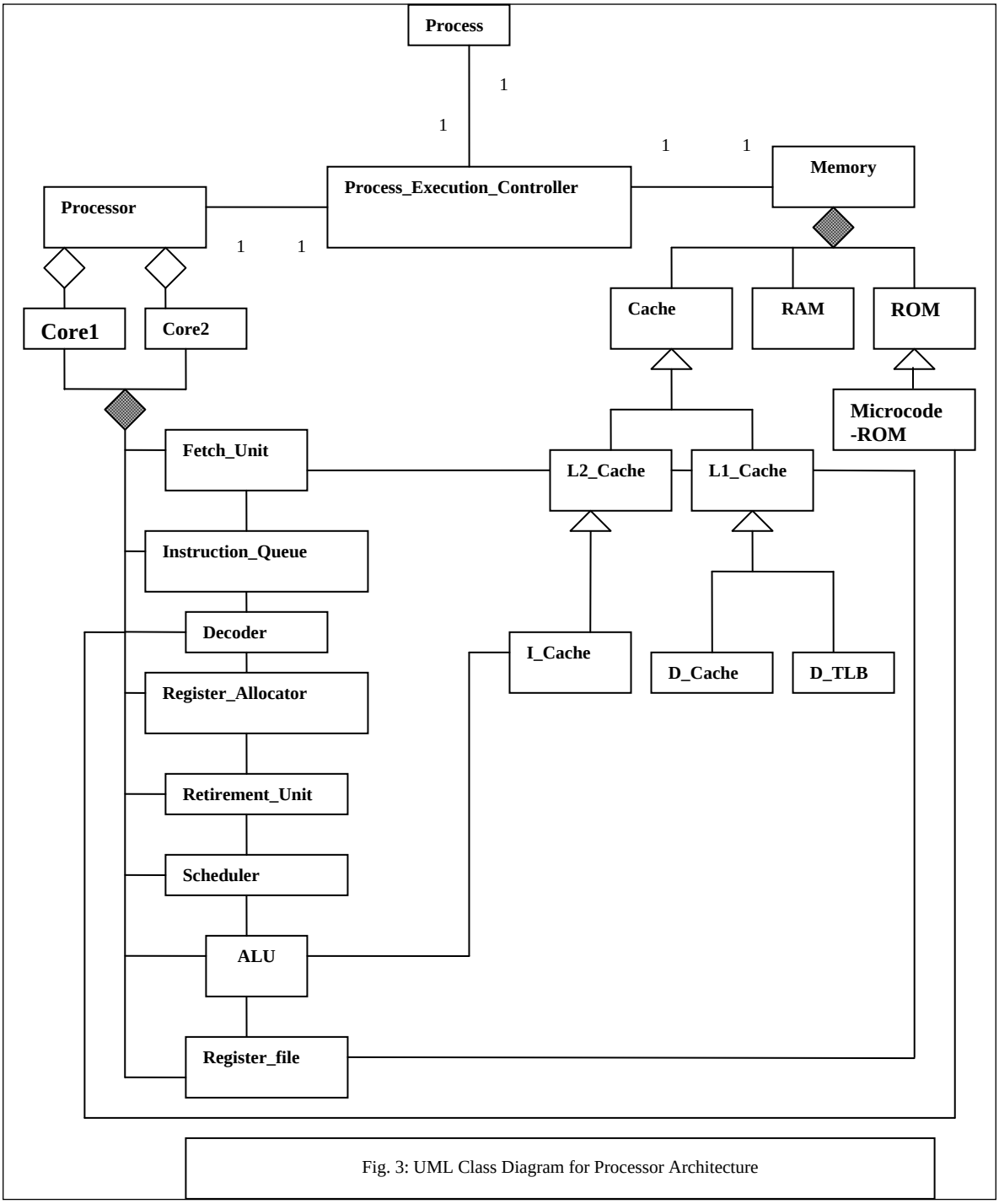

Fig. 3: UML Class Diagram for Processor Architecture

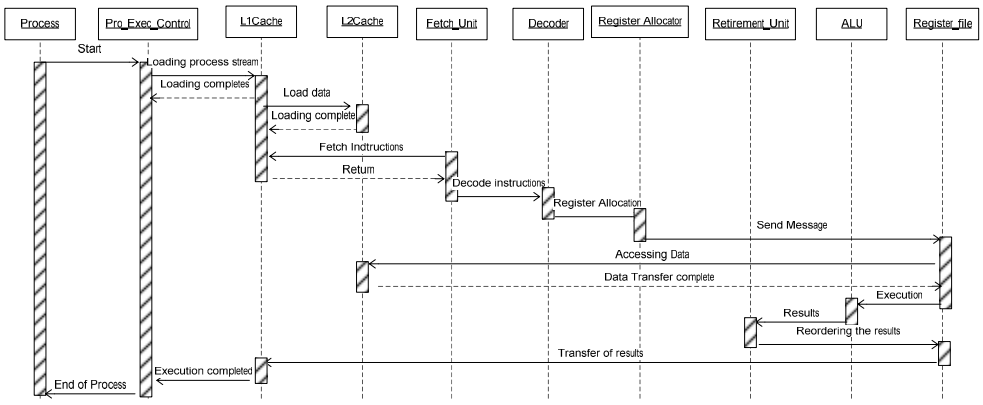

Fig. 4: UML Sequence Diagram for Process Executio

TABLE I: ARCHITECTURAL DETAILS OF PENTIUM DUAL CORE AND CORE 2 DUO MACHINES

Specifications	Intel® Pentium® Dual Core CPU	Intel® Core™2 Duo Core CPU
Number of cores	02	02
Family	Intel Pentium Dual Core for Moile	Intel® Core™2 Duo Mobile Processor
Model number	T3200	T5670
Clock speed	2.00GHZ	1.86 GHZ
Bus speed	667 MHZ	800 MHZ
Level 1 cache size	2 x 32 KB instruction caches 2 x 32 KB data caches	2 x 32 KB instruction caches 2 x 32 KB write-back data caches
Level 2 cache size	shared 1 MB	shared 2 MB
Instruction sets	MMX instruction set, SSE, SSE2, SSE3, EM64T	MMX instruction set, SSE, SSE2, SSE3, EM64T, Supplemental SSE3
Memory size	1.86 GB	3.00 GB
Operating System	Windows XP Professional, Ver. 2002, Service pack2	Windows Vista Ultimate Service pack1
Make	Lenovo	Dell

TABLE II: EXECUTION TIME ON INTEL DUAL CORE CPU

	VC++				VC#				JAVA			
Lines of Code	10	10^2	10^3	10^4	10	10^2	10^3	10^4	10	10^2	10^3	10^4
Execution Time in	0	16	187	1140	0	31	218	1156	15	63	281	2766
Milli Seconds	0	15	203	1203	0	15	171	1109	16	46	297	2828
	0	16	218	1938	0	31	171	1937	16	47	344	2859
	0	31	187	1141	0	31	203	1296	15	47	328	2828
	0	32	203	1062	0	15	203	1359	16	46	343	2860

TABLE III :EXECUTION TIME ON INTEL CORE 2 DUO CPU

	VC++				VC#				JAVA			
Lines of Code	10	10^2	10^3	10^4	10	10^2	10^3	10^4	10	10^2	10^3	10^4
Execution Time in	0	15	125	1124	0	15	140	1138	0	47	203	1607
Milli Seconds	0	15	109	1139	0	15	109	1154	0	46	172	1560
	0	15	109	1192	0	15	124	1170	16	47	187	1653
	0	15	109	1107	0	15	109	1076	15	32	203	1638
	0	15	125	1061	0	15	124	1107	16	31	203	1591

TABLEIV.COMPARISON BETWEEN AVERAGE EXECUTION TIME (IN MILLI SECONDS)

	VC++				VC#				JAVA			
Processor	10	10^2	10^3	10^4	10	10^2	10^3	10^4	10	10^2	10^3	10^4
Dual Core T3200	0	22.0	165.8	1296.8	0	22.6	193.2	1371.4	15.6	49.8	318.6	2828.2
Core 2 Duo T5670	0	15.0	115.4	1124.6	0	15.0	121.2	1129	9.4	40.6	193.6	1609.8

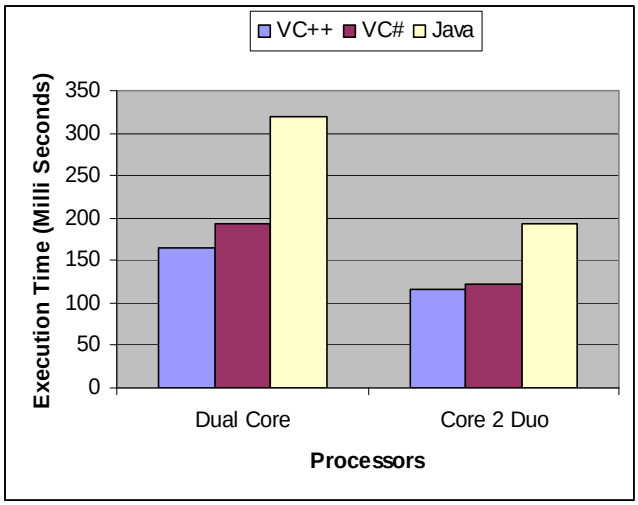

Fig. 5a: Performance comparison for 103 lines of code

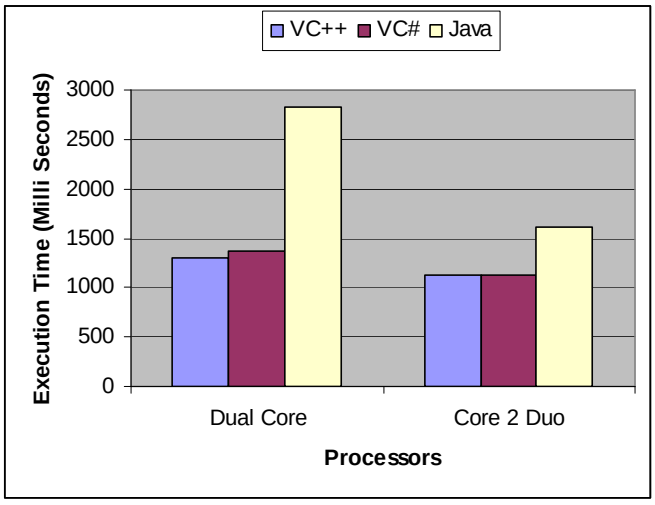

Fig. 5b: Performance comparison for 104 lines of code

ACKNOWLEDGMENT

The authors are very thankful to Prof. B. Hanumaiah, Vice-Chancellor, Babasaheb Bhimrao Ambedkar University (A Central University), Vidya Vihar, Rae Bareilly Road, Lucknow, India, for providing excellent computation facilities in the University campus. Thanks are also due to the

University Grant Commission, India, for providing financial assistance to the Central University for research work.

REFERENCES

[15] Alhir, S.S. *UML in a Nutshell: A Desktop Quick Reference*, O'Reilly & Associates, First Indian Reprint, 1998.
[16] Booch, G., Rumbaugh, J., Jacobson, I. *The Unified Modeling Language User Guide*, Twelfth Indian Reprint, 2004, Pearson Education.
[17] Gomaa, H., "Designing Concurrent, Distributed, and Real-Time Applications with UML". In proceedings of the 23rd International Conference on Software Engineering (ICSE'01), 2001, *IEEE Computer Society*.
[18] Pllana, S. and Fahringer, T., "UML based modeling of Performance Oriented parallel and Distributed Applications", in winter Simulation Conference, 2002.
[19] Saxena, V., Arora D. and Ahmad S., "Object Oriented Distributed Architecture System through UML", in IEEE International Conference on Advanced in Computer Vision and Information Technology, ACVIT-07, Nov. 28-30, 2007, ISBN 978-81-89866-74-7, pp. 305-310.
[20] Pustina, Lukas, Schwarzer, Simon, Martini, Peter, Muurinen, Jari, Salomaki, Ari,, "A Methodology for Performance Predictions of Future ARM Systems Modelled in UML", in SysCon 2008 - IEEE International Systems Conference, Montreal, Canada, April 7–10, 2008.
[21] Saxena, V. and Raj D., "UML Modeling for Instruction pipeline", in World Conference on Science, Engineering and Technology, WCSET 2008, August, 30-September, 1, 2008. Available: www.waset.org/pwaset.
[22] Fateh Boutekkouk, Mohammed Benmohammed, Sebastien Bilavarn and Michel Auguin, "UML for Modelling and Performance Estimation of Embedded Systems", in *Journal of Object Technology*, vol. 8, no. 2, 2009, pp. 95-118. Available: http://www.jot.fm/issues/issue_2009_03/article1/
[23] Henderson, Robert and Zorn Benjamin, "A Comparison of Object-oriented Programming in Four Modern Languages", in Software—Practice and experience, vol. 24(11), pp. 1077–1095, John Wiley & Sons, Ltd. 1994.
[24] Glyph Lefkowitz, "A subjective analysis of two high-level, object-oriented languages Comparing Python to Java", 2000. Available: http://twistedmatrix.com/~glyph/rant/python-vs-java.html
[25] Cowell-Shah, Christopher W., "Nine Language Performance Round-up: Benchmarking Math & File I/O", 2004. Available: http://www.osnews.com/story/5602
[26] Saxena, Vipin and Arora, Deepak , "Performance Evaluation for Object Oriented Software Systems" , SIGSOFT Software Engineering Notes, March 2009, Volume 34, Number 2.
[27] Simcha Gochman, Avi Mendelson, Alon Navh and Efraim Rotem, "Introducttion to Intel Core TM DUO Processor Architecture" , Intel technology Journal, Vol. 10, Issue 2, May, 15, 2006.
[28] Ofri Wechsler, "Inside Intel® Core™ Microarchitecture: Setting New Standards for Energy-Efficient Performance", Technology@Intel Magazine, March 2006.

Dr. Vipin Saxena: He is a Reader, Founder and Ex-Head, Dept. of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, India. He got his M.Phil. Degree in Computer Application in 1992 & Ph.D. Degree work on Scientific Computing from University of Roorkee (renamed as Indian Institute of Technology, India) in 1997. He has more than 13 years and 08 months of teaching experience and 17 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 and Hardware Domains. He has published more than 75 International and National publications. Phone: +91-9452372550

Manish Shrivastava: He is a Research Scholar, Dept. of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, India. He got his M.Phil. Degree in Computer Applications in 1992. He has more than 12 years of teaching experience. Currently he is actively engaged in the research work on the Unified Modeling Language. He has produced several outstanding research publications. Phone: +91-9453847114;