A Review on Distributed Systems and Programming Language Issues

Dr. B.Raveendranadh Singh,
Professor in CSE & Principal,
Visvesvaraya College of Engineering and Technology.

Abstract:

Distributed computing are the computing that leverages parallel computing, high speed computing and other technologies like virtualization, grid commuting and cloud computing. Distributed computing has plethora of applications in the real world. With the advent of technologies like RMI, EJB, JMS, CORBA, DCOM, WCF and web services distributed computing has been realized. These technologies help to reuse existing applications and also integrate heterogeneous applications seamlessly. This helps business integration among similar businesses in supply chain managements. Moreover the distributed computing bestows advantages like fault tolerance, location transparency, 100% availability, scalability and load balancing. Nevertheless, the distributed computing has some issues such as communication, parallelism, and synchronization. This paper throws light into these issues besides covering the applications of distributed computing.

Index Terms :

Distributed computing, distributed computing technologies, distributed computing platforms, applications of distributed computing, issues in distributed computing

1. Introduction:

Distributed computing is the result of evolution of computing since from the inception of it. Distributed computing refers to the system in which multiple computers work together and appear like a single system from the standpoint of end user. Distributed computing can leverage the power of supercomputing with additional advantages such as location transparency, fault tolerance, round the clock availability of service besides making it highly scalable [2]. Right from the standalone computing, the computational method evolved into centralized computing, client server computing and then distributed computing. The distributed computing paradigm has reached further heights in terms of technologies like grid computing [19], cloud computing and virtualization. With respect to the programming languages there are many distributed computing technologies with underlying languages support. For instance Java/J2EE platform supports Remote Method Invocation (RMI) [3], Enterprise Java Beans (EJB) [3], and Java Messaging Service (JMS) [3] as distributed technologies. Only Java language can use these technologies generally unless native code libraries are used. Microsoft .NET platform also provides distributed technologies like Distributed Component Object Model (DCOM) [3] and Windows Communication Foundation (WCF) [3] while Common Object Request Broker Architecture (CORBA) [4] can be used by any programming language. These distributed technologies are used to realize the dreams of enterprises to have cross platform integration with the emergence of supply chain managements and integration of businesses in terms of e-Commerce applications like B2B, B2C and C2C [5].

As these technologies are widely used in a Service Oriented Architecture (SOA) [3], these distributed programming platforms have some inherent issues. The issues include communication, synchronization, and parallelism [6] besides having the distributed computing support partially in terms of platform support. Nevertheless, the distributed computing has plethora of advantages such as cost effectiveness, high speed computing, reliability, scalability (growth in incremental fashion), data sharing, device sharing, machine to machine communication without human intervention and flexibility [2].

2. Distributed Systems:

This section throws light into distributed systems and their applications and underlying programming requirements. Distributed computing is essentially a service side phenomenon which is the computing that takes place over multiple networked services to for a highly reliable and high speed server. Multiple systems will act as a single system with advantages described in the previous section [2]. Applications of distributed computing are common in modern enterprise applications. In such environment shared objects are used their needs synchronization mechanism. Distributed synchronization is required in order to have fault tolerance and load balancing in server side applications. NoSQL is one of the issues in distributed computing where scalability is the reason for the complex environment [6]. Qing and Jianhe [7] studied parallel computing in Local Area Network (LAN). In other words distributed parallel computing is applied an application of distributed computing over LAN. Thus it could solve scientific computing problems with ease.

Zhang and Zheng [8] explored network parallel computing which is very much similar to distributed computing. Liskin et al. [9] focused on integrated feedback systems for the underlying distributed projects. They tested distributed systems towards software quality improvement. Gruver [10] presented distributed intelligent systems that are in the real world. They include distributed energy systems, RFID tracking, automated decision support, distributed scheduling, and manufacturing automation. Liu [11] focused on distributed parameter systems.

Inter process communication is essential in distributed computing paradigms, especially when there is communication among heterogeneous systems. Manteuffel [12] described the parallel computing and underlying issues using TransC language. Wang et al. [13] studied distributed systems. They made software reliability testing on distributed systems. Peng, Yuanyuan and Wu [14] presented mechanisms for synchronization in distributed inter process communication systems. They used V as communication programming interface. Siegel [15] studied robust resource management in distributed systems. Especially they focused on resource management heuristics that are used to optimize resource allocation in distributed environments. Peng and Wang [16] focused on parabolic distributed parameter systems that leverage deterministic learning. Islan and Liu [17] presented the distributed approach in hybrid sliding mode control systems in order to reduce parametric uncertainty. HuaZhou [18] explored distributed systems in terms of visual metamodeling for making domain specific models in distributed environments.

Zhu and Fan [20] managed to present a system that deals with digital cities. These researchers focused on digital cities and processing them in distributed computing paradigm. Xueping and Zhiquan [21] studied distributed parameter systems by using a method known as "model description". In [22] graphical specification language was studied for distributed systems to know the behavioral relations. Lee et al. [23] extended UML for modeling distributed control systems with case study examples. Leuven et al. [24] studied the problems of fault tolerance, security and quality of service in distributed systems. White [25] explored decision support system in distributed environments by gathering requirements.

2.1 Classes of Distributed Systems:

Distributed systems are used for many different types of applications. Programming applications on distributed systems fall into four general categories: decreasing turnaround time for single system, increasing reliability and availability, the use of parts of the system to provide special functionality, and the inherent distribution of the application [1]. Distributed computing can be used especially for leveraging computing resources and optimal usage of resources. It is also used to exploit the modern parallel processing.

2.2 Parallel, High-Performance Applications:

Parallelism is a reason for an application to run on a distributed system to achieve speedup. Some programs will run faster when running different parts of the program on different systems at the same time. Parallel applications can be run on shared-memory multiprocessors; however, shared-memory processors do not scale to large number of processors. This explains the interest in implementing parallel programs on distributed systems. Distributed systems are known for parallelism and the support of applications that can make use of computing power. The distributed systems make use of the power of normal CPUs from multiple systems and use them as a super computing. It does mean that distributed system can bring about the power of a super computer with advantages of supercomputer.

2.3 Fault-Tolerant Applications:

Distributed systems are potentially more reliable because of their capability to fail partially. Due to this property, failure of one processor does not affect the functioning of other processor, since the processors are autonomous. By replicating functions or data of the application on several processors, reliability can be increased. Especially in server side functionality distributed computing is essential. For instance multiple servers can be connected at server side with replication so as to balance load among the server. This will ensure that the distributed computing brings about fault tolerant applications. When a server is down, the requests will be processed by other servers. Thus it is fault tolerant.

2.4 Requirement for Distributed Systems Support:

Distributed systems need the support of hardware and software. With regard to hardware, they need multiple processing to achieve high speed computing. With regard to software, they need technologies like RMI, EJB, JMS, CORBA, DCOM, and WCF for integrating multiple systems in distributed environments. The systems that are integrated might be homogenous or heterogeneous.

2.5 Language for Distributed Systems:

Object oriented languages like C++, Java and C# support distributed programming. These languages with technologies in the underlying platforms can help building scalable distributed applications. The issues to be addressed in the design of language for distributed systems are parallelism, inter-process communication and synchronization, and partial failure.

3. Language Support for Distributed Systems:

This section focuses on the language support for distributed systems that will have mechanism like parallelism, communication and synchronization. It also throws light on partial failure and how the distributed systems are inherently fault tolerant.

3.1 Parallelism:

The multiple processors in a distributed system enable more than one part of the program to run simultaneously.

This is called parallelism. On the other hand, sometimes a program is represented as collection of processes running parallel, irrespective of the actual processes running simultaneously on different processors. It is called pseudo-parallelism. The distinction between parallelism and pseudo-parallelism are usually hidden form the programmer. In some languages, the distinction between the two is not hidden from the programmer and it is possible to assign different parts of program to different processing units. Parallel programming is the issue when it is not supported by good technology. There are many advantages of it when it is used with care.

3.2 Inter-process Communication and Synchronization:

Inter-process communication and synchronization deals with how the parts of program running parallel on different processors interact with each other, i.e., communicate and synchronize. One of issues in synchronization is non-determinism, where a process waits for information from more than one processes. Inter-process communication can be divided into two – shared data passing and message passing. These are used based on the requirement.

Inter process communication is essential in order to have robust coordination among the processes and achieve more than one job simultaneously. However, in reality the processors run tasks one after another unless multiple processors are used. In modern computers where multiple processors are used, it is possible to have inter-process communication.

The factors that affect sending of a message – who is sender, what is sent, who is the recipient, it is guaranteed to be delivered, it is guaranteed to be received, will there be a reply, what to do when some thing geos wrong. Synchronization is essential when there are replicas maintained by server side mechanisms. Replicas help in load balancing at server side processing. However, synchronization has to be achieved correctly else it becomes a problem. The distributed computing platforms are providing robust mechanism for synchronization.

3.3 Partial Failure:

Distributed systems languages must address a potential of partial failure of the system. Even if some of the processors in the system fail, the distributed system should be able to function with the remaining processors provided all the information on the failed processors is also stored on at least one of healthy processors.

A system is fault tolerant if it can continue its operations even during partial failures. When a server machine fails in distributed computing, other server machines will complete the job. This is achieved due to the clustering concept used in the server. The server machines form as cluster and they contain replica of users' data. This will ensure the scalability in processing increased number of requests. This will help to add more servers to make the system remain reliable, scalable with high availability.

4. Conclusion:

This paper studies distributed computing, its applications and issues in distributed computing. Especially it focuses on issues like parallelism, communication and synchronization. These are actually advantages that help in realization of distributed computing unless they are done properly.

This paper throws light on these issues and the wide range of application possibilities of distributed systems. In essence distributed systems in the real world are capable of integrating businesses and sharing of systems that provide many advantages like fault tolerance, location transparency, 100% availability, scalability and load balancing. This paper also explores distributed technologies, development platforms and programming languages support in some detail.

5. References:

[1]. Henri E. Bal, Jennifer G. Steiner, Andrew S. Tanenbaum. "Programming Languages for Distributed Computing Systems", In ACM Computing Surveys, Vol.21, No. 3, September 1989.

[2]. Guohong Cao. (n.d). CSE 513: Distributed Systems. psu.edu. p1-16.

[3] Parasoft. (2007). Parasoft SOATest. Parasoft. p1-2.
[4] M. L. Liu. (n.d). The Common Object Request Broker Architecture (CORBA). Rutgers.edu. p1-54.
[5] Sxccal. (n.d). Introduction to e-commerce. Sxccal. edu. p1-28.

[6] Joseph M. Hellerstein. (2013). BOOM: Experiences in Language and Tool Design for Distributed Systems (Keynote). IEEE. p1.

[7] Guan Qing and Guan Jianhe. (2012). The Realization of Parallel Computing in LAN. IEEE. p950-953.

[8] kefei Zhang and Xiuhong Zheng. (2012). A Network Parallel Computing System Based on Ethernet. IEEE. p65-68.

[9] Olga Liskin, Christoph Herrmann, Eric Knauss, Thomas Kurpick, Bernhard Rumpe and Kurt Schneider. (2012). Supporting Acceptance Testing in Distributed Software Projects with Integrated Feedback Systems: Experiences and Requirements. IEEE. p84-93.

[10] William A. Gruver. (2012). Distributed Intelligent Systems: A Paradigm Shift. IEEE. p1.

[11] Feng Liu, Guodong Shi and Guodong Shi. (2010). Well-Posedness of Two Classes of Singular Distributed Parameter Systems in Hilbert Space. IEEE.p299-304.
[12] Henning Manteuffel, Cem Savas¸ Bas¸soy, Friedrich Mayer-Lindenberg. (2010). The TransC Process Model and Interprocess Communication. IEEE. p87-93.

[13] lun Wang, Weiru Chen and Hongji Yang. (2010). Architecture Description Language Based on Software Reliability Evaluation for Distributed Computing System. IEEE. p370-374.

[14] Xiao Peng and Li Yuanyuan, Deng Wu. (2009). A Model of Distributed Interprocess Communication System. IEEE. p276-279.

[15] Howard Jay Siegel. (2009). Stochastically Robust Resource Management in Heterogeneous Parallel Computing Systems. IEEE. p1-2.

[16] Tao Peng and Cong Wang. (2009). Identification of A Class of Parabolic Distributed Parameter Systems via Deterministic Learning. IEEE. p35-40.

[17] Islam.S and Liu.P.X (2010). A Distributed Hybrid Sliding Mode Control System for a Class of Nonlinear Mechanical Systems. IEEE. p356-361.

[18] HuaZhou XingPing Sun+ ZhiHong Liang HongWei Kang QingDuan Hongji Yang. (2008). XMML: A Visual Metamodeling Language for Domain-Specific Modeling and Its Application in Distributed Systems. IEEE. p133-139.

[19] Weiwei Lin, Changgeng Guo, Deyu Qi ,Yuehong Chen and Zhang Zhili. (2006). Implementations of Grid-Based Distributed Parallel Computing. IEEE. p1-6.

[20] Dingju Zhu and Jianping Fan. (2008). Application of Parallel Computing in Digital City. IEEE. p845-848.

[21] Dong Xueping and Wang Zhiquan. (2006). Stability analysis of a class of nonlinear distributed parameter switched systems. IEEE. p1055-1059.

[22] Jorge Cortes Galicia and Felipe Rolando Menchaca Garcia. (2006). Graphical Specification Language for Distributed Systems. IEEE. p1-6.

[23] Chongwon Lee, Jongdae Han, Jaekeun Shim, Chunwoo Lee, Taeksu Kim, Yoohoon Kang, Byungjeong Lee and Chisu Wu. (2006). Extending UML for Development of Distributed Control Systems with Heterogeneous Languages. IEEE. p1-8.

[24] Rodica Tirtea, LeuvenGeert.K.U, Deconinck and Ronnie Belmans. (2006). Fault Tolerance Adaptation Requirements vs. Quality-of-Service, Real- Time and Security in Dynamic Distributed Systems. IEEE. p296-303.

[25] Stephanie M. White. (2006). Requirements for Distributed Mission-Critical Decision Support Systems. IEEE. p1-7.