International Journal of Advance Research in Science and Engineering
Vol. No.6, Issue No. 04, April 2017
www.ijarse.com
IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

CHALLENGES IN TESTING OF DISTRIBUTED SYSTEMS

Mamta

Deptt of comp. Sc., Research Scholar

ABSTRACT

A distributed system is a collection of independent computers which are linked by a network. The testing of distributed system includes unit testing, integration testing, and system testing. There are many tools and techniques are used to carry out the testing. Testing is an important measure to improve software quality. Functional tests are executed to verify the systems behaviour against given requirements. This work deals with general difficulties and aims when testing complex distributed systems, especially when heterogeneous interfaces are used. There is remote test is proposed, a framework for the test of distributed systems and their interfaces. I will survey the recent progress in this area and I will discuss the current state-of-the- art in propositional reasoning focusing on a series of challenge problems concerning propositional encodings, compilation techniques, approximate reasoning, robustness, and scalability.

II. INTRODUCTION

Distributed system is a collection of autonomous computers which are linked by a network or with using software to produce an integrated computing facility [1][2][3].

- **According to Coulouris**

A distributed system consists of hardware and software components located in a network of computers that communicate and coordinate their actions only by passing messages[4].

- **According to Tanenbaum & van Steen**

A distributed system is a collection of independent computers that appears to its users as a single coherent system[5].

- **According to Lamport**

A distributed system is a system that prevents you from doing any work when a computer you have never heard about, fails[6]. Testing is an important measure to improve software quality[7]. Functional tests are executed to verify the systems behavior against given requirements [8]. Since testing is an expensive and also time consuming step, it is essential to drive everything that supports the test process [9]. Interaction property is a natural feature of many distributed systems which finish their work on network [10][11][12].

II. RELATED WORK TESTING DISTRIBUTED SYSTEMS

The testing distributed systems by using a test framework, has been done by [13] [14]. This approach uses Java Reflection and Aspect Oriented Programming in a central development environment to automatically provide a distributed test-bed. It is, however, limited to Java programs and seems to work on a detailed programming

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

level. Another distributed test framework is proposed by [15] [16]. The presented scenario-based framework generates test scripts from a test scenario database and uses a complex 3-Tier architecture that is used to let a master unit communicate to the tested system. It also supports several features like static testing. The focus is however set on the test case generation. In contrast to the previous work, [17] presents a distributed test system completely without a master, where therefore distribution of test sequences among the testers is an important issue. In [18], it is further shown that controllability and observability problems occur in distributed test architectures with no centralized control in contrast to such systems with centralized control. Additionally [19], compares the central test architecture to the distributed test architecture and shows the prototype of a test system. Finally [20], discusses testing distributed real-time systems by analyzing execution orderings and therefore applying testing techniques for sequential software. This analysis however presumes miscellaneous fixed conditions and detailed knowledge, e.g., used scheduling or a synchronized time base. Additionally to the aforementioned testing of systems as a whole, the testing of single system components is discussed by [21], where it is pointed out that it is very important to test components that are being reused in different environments. Continuing this approach [22] finds the techniques for testing system components not well developed. In general, there is no ready-to-use software framework available yet for testing distributed systems. Hand written test cases however have to integrate the various interfaces, must coordinate the system under test (SUT) and also collect the distributed test results. The described problems and the amount of research in this topic leads to the conclusion that there is a need for tools and methods to easily and thoroughly test distributed systems. In terms of complexity this means especially to be able to automate the test and to test their individual components without interference with the system as a whole.

III. CHARACTERISTICS OF DISTRIBUTED SYSTEMS [4][23]

- Resource sharing – the possibility of using available resources any where
- Openness – an open distributed system can be extended and
- Improved incrementally – requires publication of component interfaces and standards protocols for accessing interfaces
- Scalability – the ability to serve more users, provide acceptable response times with increased amount of data
- Fault Tolerance – maintain availability even when individual components fail allow
- Heterogeneity – network and hardware, operating system, programming languages, implementations by different developers

IV. RESOURCE SHARING

Resource managers control access, offer a scheme for naming, and controls concurrency. A resource manager is a software module that manages a resource of a particular type. A resource sharing model describes how resources are made available resources can be used service provider and user interact with each other.

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

V. SCALABILITY

A system is scalable if it remains effective when there is a significant increase in the amount of resources (data) and number of users Internet: no of users and services has grown enormously. Scalability denotes the ability of a system to handle an increasing future load requirements of scalability often leads to a distributed system architecture (several computers). Scalability problems (1) Often caused by centralized solutions Scalability problems (2) Characteristics of decentralized algorithms: No machine has complete information about the system state. Machines make decisions based only on local information. Failure of one machine does not ruin the algorithm. There is no implicit assumption that a global clock exists. Scaling techniques

(1) Distribution splitting a resource (such as data) into smaller parts, and spreading the parts across the system. Replication replicate resources (services, data) across the system increases availability, helps to balance load caching (special form of replication). Hiding communication latencies avoid waiting for responses to remote service requests (use asynchronous communication or design to reduce the amount of remote requests)

(2) Reducing amount of remote requests: The difference between letting (a) a server or (b) a client check forms as they are being filled.

VI. FAILURE HANDLING

Hardware, software and network fail!! DS must maintain availability even in cases where hardware/software/network have low reliability Failures in distributed systems are partial makes error handling particularly difficult Many techniques for handling failures

1. Detecting failures (checksum a.o.)
2. Masking failures (retransmission in protocols)
3. Tolerating failures (as in web-browsers)
4. Recovery from failures Redundancy (replicate servers in failure-independent ways)

Example: Google File-System Early days… Challenges: …today - Scalability - Fault-tolerance - Auto recovery

VI. DISTRIBUTION TRANSPARENCY

An important goal of a distributed system is to hide the fact that its processes and resources are physically distributed across multiple computers A distributed system that is able to present itself to its users and applications as if it were only a single computer system is said to be transparent. Transparency in a distributed system Different forms of transparency in a distributed system (ISO, 1995)

VII. CHALLENGES OF DISTRIBUTED SYSTEMS

- Concurrency – components execute in concurrent processes that read and update shared resources. It requires coordination.

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

- Global clock – There is no global clock. It makes coordination difficult (ordering of events).
- Independent failure of components – "partial failure" & incomplete information
- Unreliable communication – Loss of connection and messages.
- Message bit errors
- Unsecure communication – Possibility of unauthorised recording and modification of messages
- Expensive communication – Communication between computers usually has less bandwidth, longer latency, and costs more, than between independent processes on the same computer
- Basic design issues- It includes naming, communications, software structure, workload allocation, consistency maintenance etc. The Naming includes communication identifier, name service, contextual resolution of name, name mapping, pure names vs names with meaning. The reasons for communication are transfer of data, synchronization, methods of communications, message passing - send and receive primitives, synchronous or asynchronous, blocking or non-blocking, mechanisms of message passing - channels, sockets, ports, client-server communication model, group multicast communication model [25].

VIII. CONCLUSION

Interaction property is a natural feature of a distributed system. Based on interaction property, the test work can focus on the interesting part and ignore the other part. After the selected interaction property is given, the scale of the problem us reduced. In the paper, test based on interaction property is considered. The definitions related to all test work are given. The input and the output in interaction property are selected not only randomly but also purposely. Meanwhile, in order to check whether the whole interesting work is finished correctly, several interactions are considered from test generation including test verdict, and test implementation. An algorithm is proposed to generate executable test sequence and its complexity is completely analyzed. The advantages are that the test work pertinent is enhanced so that the scale of the problem is reduced and the deployment of the test work is considered simultaneously. The coverage of a test sequence is discussed and the verdict method is given. The research work in the future is the algorithm optimization because we wish we can find an algorithm which may simultaneously cover as many as interesting transitions and has the minimum number of executable test sequences. Moreover if the selected PCOs (Point of Control and Observation) cannot be deployed in expected position, where to deploy them is also needed to consider.

REFERENCES

[1.] Spillner, T. Linz, and H. Schaefer, Software Testing Foundations: A Study Guide for the Certified Tester Exam, 2nd Edition. Rocky Nook, 2007.

[2.] S. Ghosh and A. Mathur, "Issues in testing distributed component-based systems," in First ICSE workshop on testing distributed component-based systems , 1999.

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

[3.] O. Gantz, V. Knollmann, K. P. Jaschke, and K. Lemmer,"Visualisierung im bahnlabor railsite," Z. univerzita v Ziline, Ed. EDIS, May 2006, vol. 3, pp. 162–168. [Online]. Available: http://elib.dlr.de/43823/

[4.] M. Hammerl, M. D. Filippis, I. Steinh auser, C. Torens, O. Gantz, M. M. zu H orste, and K. Lemmer, "From a testing laboratory for railway technical components to a human factors simulation environment," in 3RD INTERNATIONAL RAIL HUMAN FACTORS CONFERENCE , Februar 2009. [Online]. Available: http://elib.dlr.de/58410/

[5.] Torens, I. Steinh auser, M. Busse, and L. Ebrecht, "Virtuelle realit at und simulation im bahnlabor railsite - modulare architektur erm¨oglicht flexiblen einsatz," ,GI- Workshop Virtuelle und Erweiterte Realitat, ser. Workshop der GI-Fachgruppe VR/AR, A. Gerndt and M. E. Latoschik, Eds. Shaker Verlag, November 2009, pp. 61–72. [Online]. Available: http://elib.dlr.de/60990/

[6.] The Single UNIX Specification, Version 2: sys/shm.h – shared memory facility, The Open Group Std., 1997.

[7.] IEEE, Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks , IEEE Computer Society Std., December 2008.

[8.] Common Object Request Broker Architecture (CORBA) Specification, Version 3.1, Object Management Group Std., January 2008.

[9.] I.120 : Integrated services digital network (ISDN), INTER-NATIONAL TELECOMMUNICATION UNION (ITU) Std.,March 1993.

[10.] CAPI: COMMON-ISDN-API , CAPI Association e.V. Std.,June 2001.

[11.] TIA-232-F Interface between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Bi-nary Data Interchange , Telecommunications Industry Asso-ciation Std., October 1997.

[12.] TIA-485-A Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems,Telecommunications Industry Association Std., March 1998.

[13.] J. L. Buie, "Coupling transistor logic and other circuits," U.S. Patent 3,283,170, November, 1966.

[14.] Hughes, P. Greenwood, and G. Coulson, "A framework for testing distributed systems," in P2P '04: Proceedings of the Fourth International Conference on Peer-to-Peer Computing .Washington, DC, USA: IEEE Computer Society, 2004, pp. 262–263.

[15.] W. T. Tsai, Y. Na, R. Paul, F. Lu, and A. Saimi, "Adaptive scenario-based object-oriented test frameworks for testing embedded systems," in Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th An-nual International , 2002.

[16.] W. T. Tsai, L. Yu, and A. Saimi, "Scenario-based object-oriented test frameworks for testing distributed systems," in FTDCS '03: Proceedings of the The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems . Wash-ington, DC, USA: IEEE Computer Society, 2003, p. 288. Khoumsi, "Testing distributed real time systems using a distributed test architecture,"Computers and Communica-tions, IEEE Symposium on , vol. 0, p. 0648, 2001.

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

[17.] R. M. Hierons and H. Ural, "The effect of the distributed test architecture on the power of testing," The Computer Journal vol. 51, no. 4, pp. 497–510, 2008. [Online]. Available:

[18.] http://comjnl.oxfordjournals.org/cgi/content/abstract/51/4/497 W. Ulrich, P. Zimmerer, and G. Chrobok-Diening, "Test architectures for testing distributed systems," 1999, presented at Quality Week 1999.[Online]. Available: http://www.stickyminds.com/sitewide.asp?ObjectId=1462&Function=edetail&ObjectType=ART

[19.] H. Thane, "Monitoring, testing and debugging of distributed real-time systems," Ph.D. dissertation, Mechatronics Labo-ratory, Department of Machine Design Royal Institute of Technology (KTH) S-100 44 Stockholm, Sweden., May 2000.

[20.] J. Weyuker, "Testing component-based software: A cau-tionary tale,"IEEE Softwarevol. 15, pp. 54–59, 1998.

[21.] Y. Wu, D. Pan, and M.-H. Chen, "Techniques for testing component-based software,"Engineering of Complex Computer Systems, IEEE International Conference on, vol. 0, p.0222, 2001.

[22.] G. F. Coulouris, J. Dollimore, and T. Kindberg, Distributed systems concepts and design. Addison-Wesley, 1994.

[23.] V. Massol and T. Husted, JUnit in Action . Greenwich, CT, USA: Manning Publications Co., 2003.

[24.] J. M. Zelle, Python Programming: An Introduction to Computer Science. Franklin B, 2003.

[25.] T. Mackinnon, S. Freeman, and P. Craig, "Endo-testing: Unit testing with mock objects," Extreme programming examined,pp. 287–301.