Volume 6, Issue 4, April 2016 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Complexity Analysis Involving Heterogeneous System

Ashok Kumar, Komal Mehta
BITS, Bhiwani, Haryana,
India

Abstract*– *Complexity analysis is one of the most complicated topics in mathematics. It involves an unusual concept and some tricky algebra. This report is a humble trail to demystify the idea in detail. Heterogonous systems are becoming bigger and more complex. While the complexity of large-scale heterogeneous systems has been acknowledged to be an important challenge, there has not been much work in defining or measuring system complexity. Thus, today, it is difficult to compare the complexities of different systems, or to state that one system is easier to program, to manage or to use than another. Here we try to understand the factors that cause heterogeneous systems to appear very complex to people. We define different aspects of system complexity and propose metrics for measuring these aspects. We also show how these aspects affect the system. Based on the aspects and metrics of complexity, we propose general guidelines that can help to measure the complexity of systems. There are many types of analysis and various methods available to analyze the algorithms like appropriate analysis, posterior analysis, micro analysis, macro analysis, amortized analysis, big O notation, theta notation, potential method, accounting method. All analysis and methods are situation specific. Our report does a comparative study of various method and techniques of algorithm analysis giving their specific advantages and disadvantages.

Keywords*– *Encryption, DES/3DES, Big O Notation, Routing Algorithms etc.

I. INTRODUCTION

The size and complexity of heterogeneous systems have been increasing inexorably in the recent past. Large-scale distributed systems such as internet systems, ubiquitous computing environments, grid systems, storage systems, enterprise systems and sensor networks often contain immense numbers of heterogeneous and mobile nodes. These systems are highly dynamic and fault-prone as well. As a result, developers find it difficult to program new applications and services for these systems; administrators find it difficult to manage and configure these complex, device-rich systems; and end-users find it difficult to use these systems to perform tasks. The computational aspects of complexity have been studied extensively. The concepts of time and space complexity of different kinds of algorithms are well understood. However, the human aspect of complexity is still poorly understood. This human aspect of complexity is a significant challenge today, requiring urgent solutions. The complexity of heterogeneous systems for people has been widely identified to be an important problem [1,2]. Many enterprises, governments and other organizations have large, complex computing systems that are difficult to manage, maintain and change. Application and service developers often face steep learning curves before they can start programming large distributed systems. Organizations also tend to spend huge amounts in administration and maintenance costs, which often exceed the cost of buying the system in the first place. End-users, too, are often overwhelmed with the complexity of a single computer, let alone multiple, heterogeneous devices. In spite of the fact that system complexity has been widely talked about, the term ―complexity‖ is often used loosely in connection with computer systems. There are no standard definitions of complexity or ways of measuring the complexity of large systems. As with so many complex things, computer system complexity means different things to different people. In this report, we identify the different algorithms that take part in the complexity of heterogeneous system: Fault tolerance algorithm, routing algorithms, backup algorithm, error detection and correction algorithms etc. We describe the rationale behind these different aspects and show how these aspects manifest themselves in heterogeneous systems. This report shows how these aspects of complexity impact the overall system. The computer systems research community 2 has been actively looking at different approaches to reduce the complexity of systems. These approaches often take the form of middleware or programming frameworks to simplify the task of developers [3], various system management tools for administrators; and intuitive user interfaces for end-users. However, today, there is no way of formally, and quantitatively, saying that a certain solution does reduce complexity, or that one solution is better than another. We hope that our proposed aspects and metrics of complexity will allow people to compare solutions in a more scientific way, as well as guide future solutions to tackling the problem of complexity. So far, approaches to tackling system complexity have been rather ad-hoc in manner. The main contribution of this report is in addressing the problem of heterogeneoussystem complexity in a more formal and scientific way.At last we will find the complexity of Internet banking system. Internet Banking developed due to increasing demand of online banking transactions. The biggest advantages of Internet Banking consist of complex banking solutions, 24 hours availability, quick and secure access to the back-end application through Internet. Customer requests for quick access and no matter the location at their bank accounts, or at financial/banking transactions, all of these have determined the banking institutions worldwide to adopt the internet as the optimal solution for the presented demands. Through the internet network banks are able to connect front-end applications with back-end. Based on such advantages, internet banking applications were created. The evolution of bank presence on web from simple, static applications to complex, dynamic application, to complex, dynamic application with numerous transactions, is presented below (figure 1.0). The advantages of internet banking applications consist of: quickness; secured access to sensitive data as accounts, personal data of customers, transactions; account management; operating sale purchase transactions in real-time and at long distance; suppressing the stress of staying in bank for a transaction; low costs for the maintenance of this kind of applications.

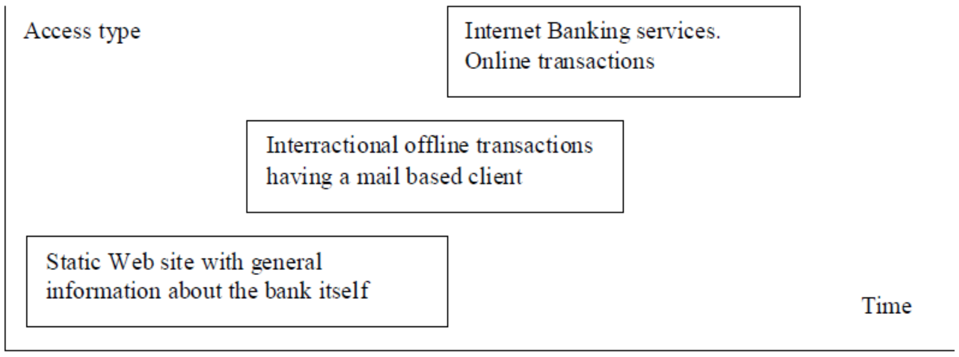

Access type
Internet Banking services. Online transactions
Interractional offline transactions having a mail based client
Static Web site with general information about the bank itself
Time

Figure1.0 Banking Growth

II. PROBLEM STATEMENT

Problem Statement

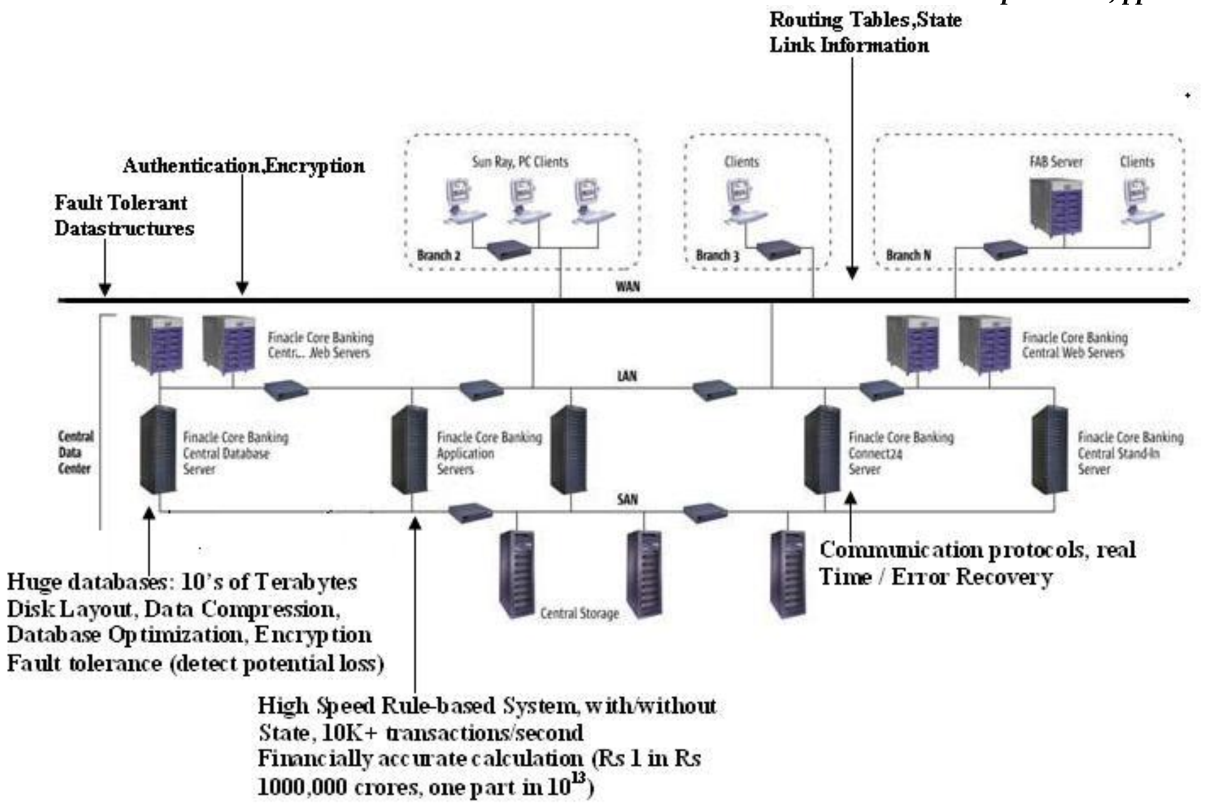
The term complexity has been widely used in different contexts by different people. In general, though, system complexity can be described as a measure of how understandable a system is and how difficult it is to perform tasks in the system. A system with high complexity requires great mental or cognitive effort to comprehend and use, while a system with low complexity is easily understood and used. In this section, we attempt to capture some of the aspects of systems that make them difficult to understand. Heterogeneous systems are becoming complex. To determine the complexity of such system is a challenging task.

Justification

There are number of algorithm that contributes in the system's complexity like fault tolerance, authentication and encryption, routing table**, c**ommunication protocol with error recovery mechanism, backup algorithm, recovery algorithm etc. Some algorithms may run in parallel, some may in sequence, how these algorithms affect the overall system complexity. There are various other factors that will play significant roles in the overall complexity. Communication is one of them. Depending upon the distance of locations where servers may be placed due to security or feasibility of the application. Backup is another issue where system needs some time to reconsolidate with the back end links or system. Depending upon the types of backup (hot or cold backup) different timings may be taken up by the system. In case of database application there may be delays due to commit and other updates on various types of storage media, which needs to be taken into account. In this report we will calculate the complexity of online banking system.

III. SYSTEM MODEL

Heterogeneous system contains many different kinds of hardware and software working together in cooperative fashion to solve problems. There may be many different representations of data in the system. This might include different representations for integers, byte streams, floating point numbers, and character sets. Most of the data can be marshalled from one system to another without losing significance. Attempts to provide a universal canonical form of information is lagging. There may be many different instructions sets. An application compiled for one instruction set cannot be easily run on a computer with another instruction set unless an instruction set interpreter is provided. There is no universal binary making process migration difficult. Recent developments in the web and Java may provide a universal interpreted language on most computers. Though a computer language is not an instruction set, this is a good compromise. Some components in the distributed system may have different capabilities than other components. Among these might include faster clock cycles, larger memory capacity, bigger disk farms, printers and other peripherals, and different services. Seldom are any two computers exactly alike. Heterogeneous System May be Based On Software as well as hardware. eg we have a system in which our data is on site A and on site B ,Site A having the SQL Server and Site B having Oracle. In this scenario if we have same hardware but are platforms are different. In another case we may have different Hardware specifications that will also cover under the type of Heterogeneous system. Let's take an example of banking system it has both different hardware as well as software. Now example illustrates the concept to measure the complexity of a heterogeneous system. This banking application utilizes all kinds of algorithms from symbolic through real time through fault tolerant. Figure 5.1 illustrates a simplified form the architectural building blocks which comprise this banking system, and algorithm classes written to execute on it. The web servers are in turn connected to a set of application servers for implementing banking rules and policies. The application server's access mirrored and/or replicated data storage.


Routing Tables,State Link Information
Authentication,Encryption
Fault Tolerant Datastructures
Sun Ray, PC Clients
Branch 2
Clients
Branch 3
FAB Server
Clients
Branch N
WAN
Finacle Core Banking Centr... Web Servers
Finacle Core Banking Central Web Servers
LAN
Central Data Center
Finacle Core Banking Central Database Server
Finacle Core Banking Application Servers
Finacle Core Banking Connect24 Server
Finacle Core Banking Central Stand-In Server
SAN
Central Storage
Communication protocols, real Time / Error Recovery
Huge databases: 10's of Terabytes Disk Layout, Data Compression, Database Optimization, Encryption Fault tolerance (detect potential loss)
High Speed Rule-based System, with/without State, 10K+ transactions/second Financially accurate calculation (Rs 1 in Rs 1000,000 crores, one part in 10^13)

IV. PROPOSED IMPLEMENTATION

Encryption is the process of converting a plaintext message into cipher text which can be decoded back into the original message. An encryption algorithm along with a key is used in the encryption and decryption of data. There are several types of data encryptions which form the basis of network security.

Encryption Algorithm (DES/3DES,IDEA,SEAL)

DES/3DES

The Data Encryption Standard (DES) was developed and endorsed by the U.S. government in 1977 as an official standard and forms the basis not only for the Automatic Teller Machines (ATM) PIN authentication but a variant is also utilized in UNIX password encryption. DES is a block cipher with 64-bit block size that uses 56- bit keys. Due to recent advances in computer technology, some experts no longer consider DES secure against all attacks; since then Triple-DES (3DES) has emerged as a stronger method. Using standard DES encryption, Triple-DES encrypts data three times and uses a different key for at least one of the three passes giving it a cumulative key size of 112-168 bits [10].

IDEA

International Data Encryption Algorithm (IDEA) is an algorithm that was developed by Dr. X. Lai and Prof. J. Massey in Switzerland in the early 1990s to replace the DES standard. It uses the same key for encryption and decryption, like DES operating on 8 bytes at a time. Unlike DES though it uses a 128 bit key. This key length makes it impossible to break by simply trying every key, and no other means of attack is known. It is a fast algorithm, and has also been implemented in hardware chipsets, making it even faster.

SEAL

Rogaway and Coppersmith designed the Software-optimized Encryption Algorithm (SEAL) in 1993. It is a Stream-Cipher, i.e., data to be encrypted is continuously encrypted. Stream Ciphers are much faster than block ciphers (Blowfish, IDEA, DES) but have a longer initialization phase during which a large set of tables is done using the Secure Hash Algorithm. SEAL uses a 160 bit key for encryption and is considered very safe [11]

V. RESULT

In online banking system we have following modules:-

1) Fault tolerance and Recovery
2) Authentication and encryption
3) Routing
4) Backup
5) Communication

Fault Tolerance and Recovery Algorithm Complexity: -

Check pointing with rollback recovery is a well-known method for achieving fault-tolerance in distributed systems. The check pointing algorithms can handle multiple concurrent initiations by different processes. While taking checkpoints, processes do not have to take into consideration any application message dependency. The synchronization is achieved by passing control messages among the processes. Application messages are acknowledged. Each process maintains a list of unacknowledged messages. Here a logical checkpoint is being used, which is a standard checkpoint (i.e., snapshot of the process) plus a list of messages that have been sent by this process but are unacknowledged at the time of taking the checkpoint. The worst case message complexity of the check pointing algorithm is *O(kn)* when *k* initiators initiate oncurrently. The time complexity is *O(n)*. For the recovery algorithm, time and message complexities are both *O(n)*[18].

Authentication Algorithm Complexity:-

In Alice and Bob Protocol Cost of each number is O(log n) bits, n numbers are sent so, Total cost is O(n*log n) bits[15].

Routing Complexity: -

In a communication network routing is one of the latency sources. Routing algorithm has task to determine the suitable port for delivering message address to one network node. Correct delivery of message from source node to destination node is considered as optimal routing. The aim is to have the fastest and optimal routing available in the network. The principle of routing with routing tables is straightforward. Ever node keeps a table with entries for each node. Using these entries, it could be determined via which outgoing port a message had to be sent. Interval routing is a space efficient routing strategy used in computer networks. The basis of interval routing idea is to label all of 40 the nodes with integer from one set (for example {1, 2, 3...*n*}) and labelling of all arcs with interval (in range of number of nodes). These intervals means that a message with interval which includes anode *u* is forwarded [16].

Table 1 Complexity of message passing in wide area networks**.**

	Routing Tables	Interval routing
Space complexity	O(n log Δ)	O(k Δ log n)
Time complexity	O(1)	O(log(k Δ))

Note: - Δ is maximum degree of the graph, n is number of nodes in the graph, k is interval

Backup Algorithm Complexity:-

The backup tree algorithm to compute a set of n-1 backup multicast delivery trees from the default multicast tree for application level multicast. For each backup multicast tree exactly one link of the default multicast tree is replaced by a backup link from the set of available links. The backup tree algorithm calculates the n-1 trees with a complexity of O (m log n)[17].

Note: - m is number of messages and n are number of nodes (systems)

Communication Complexity:-

It depends upon the algorithm used in system for Distance Vector routing scheme complexity will be *O(m)*.In case of Link state Algorithm complexity will be *O(n).*

Module wise Complexity Analysis Communication Fault Tolerance Recovery Authentication Routing Backup

*O(m) O(kn) O(n) O(n*log n) O(1) O(m log n)*

Table 6.2: Complexity of all Modules The above shown results are just an approximation, complexity may vary according to the algorithm .There are number of things that can affect the system, front end and network speed plays a vital role in this. In this type of system any algorithm may change the complexity of the system. There are number of algorithms available for particular task. In above case if we choose Open Ear Decompositions for fault tolerance routing scheme then complexity will be *O(m)* for that particular module [21]. If we change routing scheme in the system then also the complexity will be 41 change like we may switch to distance vector routing then the communication complexity will be *O(n)* instead of *O(m).*

Communication Fault Tolerance Recovery Authentication Routing Backup

*O(n) O(m) O(n) O(n*log n) O(1) O(m log n)*

Table 6.3. System Complexity As the user login first the authentication is being done then routing protocol comes in to existence ,then depends upon the mode of transaction, the user may only view the account statement in this case the complexity will be {*O(n*log n) + O(1)+O(m)&&*
O(kn)} ,authentication , routing will run in sequence viewing account statement fault tolerance will run in parallel. If user wants to transfer the money from one account to another then complexity will be different approximate {*O(n*log n)+O(n)+O(1)+O(n)+[O(m)&&O(mlogn)]}.*

VI. CONCLUSION

Hence, heterogeneous systems should provide mechanisms for uncovering the different layers of abstraction and deducing the cause of faults. The complexity of debugging a system is an important measure of the ease of using and

maintaining the system. System complexity forms just one of the factors that influence the productivity of developers, administrators and end-users. Other factors that affect complexity include communication changing requirements, technology churn, etc. While we are still far from being able to quantify these other factors, they would all form part of any evaluation of the productivity of different kinds of users. Heterogonous systems are becoming bigger and more complex. While the complexity of large-scale heterogeneous systems has been acknowledged to be an important challenge, there has not been much work in defining or measuring system complexity. Thus, today, it is difficult to compare the complexities of different systems, or to state that one system is easier to program, to manage or to use than another.. In this report, we try to understand the factors that cause heterogeneous systems to appear very complex to people. We define different aspects of system complexity and propose metrics for measuring these aspects. In conclusion, we have tried to formalize the notion of complexity of heterogeneous systems. We have proposed a number of 43 metrics to measure different aspects of system complexity. In this report, we have study about various methods of analysis and notations like appropriate analysis, posterior analysis, micro analysis, macro analysis, amortized analysis, big *O* notation, theta notation, potential method, accounting method. All analysis and methods are situation specific. Our report does a comparative study of various method and techniques of algorithm analysis giving their specific advantages and disadvantages. First of all we have analyzed the problem in detail, then study the algorithm in context to the online banking system. Generally people ask the question: How to calculate the complexity of heterogeneous system? How different algorithms affect the final complexity of system? How they take part in system? These questions don't have an easy or unambiguous answer, however. The complexity depend quite heavily on the environment what the system is?
Calculating the complexity of this type of system is just an approximation because complexity may vary on the mode of transaction. One algorithm may affect the system. In this system there are various factors that will play significant roles in the overall complexity like one minor change may increase or decrease the complexity of the system e.g. if we change authentication algorithm then the overall system complexity will affected. There are many algorithms that are running simultaneously as we discussed earlier like fault tolerance, authentication, encryption, routing tables, communication protocols and for error recovery. Every algorithm contributes in the system, so every algorithm have time and space complexity, if the algorithms are running parallel then the complexity will be highest out of them. E.g. we have complexities n, n+1, n2 then the complexity will be n2. If we have N algorithms and they are running sequentially then the complexity will be sum of all these.eg. We have N algorithms running simultaneously then the total complexity will be $Tc = 1+2+\ldots\ldots+N$. There are various other factors that will play significant roles in the overall complexity. Communication is one of them. Depending upon the distance of locations where servers may be placed due to security or feasibility of the application. Backup is another issue where system needs some time to reconsolidate with the back end links or system. Depending upon the types of backup (hot or cold backup) different timings may be taken up by the system. In case of database application there may be delays due to commit and other updates on various types of storage media, which needs to be taken into account.

REFERENCES

[1] Asprey, W. et al., ―Conquer System Complexity: Build Systems with Billions of Parts,‖ in CRA Conference on Grand Research Challenges in Computer Science and Engineering, Warrenton, VA (2002), pp. 29-33.

[2] Kephart, J.; Chess, D., ―The Vision of Autonomic Computing,‖ IEEE Computer, Vol. 36, No. 1 (2003), pp. 41-50.

[3] Campbell, A.T.; Coulson, G.; Kounavis, M.E. ―Managing Complexity: Middleware Explained‖, IEEE Computer Society's IT Professional Magazine, Vol 1, No 5, September/October 1999

[4] Design and Analysis of Algorithms By R.Panneerselvam Prentice-Hall India. ISBN: 978-81203-3278-2.

[5] Nicholas J. Higham, Handbook of writing for the mathematical sciences, SIAM. ISBN 0898714206, p. 25

[6] Donald Knuth. Big Omicron and big Omega and big Theta, ACM SIGACT News, Volume 8, Issue 2, 1976.

[7] Allan Borodin and Ran El-Yaniv (1998). Online Computation and Competitive Analysis. Cambridge University Press, 20,141.

[8] ANSI X9. 17-1985, American National Standard for Financial Institution Key Management (wholesale), Department of the Treasury Directives Manual, Electronic Funds and Securities Transfer Policy, Chapter TD 81, Section 80.

[9] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms, Second Edition. MIT Press and McGraw- Hill, 2001. ISBN 0-262-03293-7. Chapter 17: Amortized Analysis, pp.405– 430.

[10] Federal Register vol 70, number 96, Announcing Approval of the Withdrawal of Federal Information Processing Standard (FIPS) 46–3, Data Encryption Standard (DES); FIPS 74, Guidelines for Implementing and Using the NBS Data Encryption Standard; and FIPS 81, DES Modes of Operation.

[11] Balenson, D. Automated distribution of cryptographic keys using the financial institution key management standard, Communications Magazine, IEEE Volume 23, Issue 9, Sep 1985 Page(s): 41 – 46. 45

[12] FIPS PUB 1-2, Code for Information Interchange, Its Representations, Subsets, and Extensions.

[13] FIPS PUB 113 - Computer Data Authentication - the Federal Information Processing Standard publication that defines the Data Authentication Algorithm.

[14] ANSI X9.9-1982, American National Standard for Financial Institution Message Authentication.

[15] Based on lecture by Dr. Ely Porat, Complexity course, Computer Science Department, Bar-Ilan University, January 2008

[16] Complexity Analysis of Routing Algorithms in computer network, M. Bieliková (Ed.), IIT.SRC 2005, April 27, 2005, pp. 69-76.

[17] Torsten Braun,Vijay Arya and Thierry turletti ― A backup Tree Algorithm for Multicast Overlay Network‖ in Springer Berlin/Heidelberg Volume 3462/2005 ISBN 978-3-540-25809-4.

[18] Pratha Sarathi Mandal and Krishnendu Mukhopadyaya ―Concurrent checkpoint initiation and recovery algorithms on asynchronous ring networks‖ in Journal of Parallel and Distributed Computing, Volume 64 , Issue 5 (May 2004)Year of Publication: 2004 ISSN:0743-7315.

[19] Plank, J., Chen, Y., Li, K., Beck, M., Kingsley, G., Memory Exclusion: Optimizing the Performance of Checkpointing Systems, Technical Report UTCS- 96-335, University of Tennessee, August, 1996

[20] Russinovich, M., Segall, Z., Application-Transparent Checkpointing in Mach 3.0/UX, Proceedings of the 28th Annual Hawaii International Conference on System Sciences, 1995

[21] Algorithms by Kenneth A. Berman and Jerome L. Paul Algorithms, 2002 by Course Technology. ISBN-13:978-81-315-0512-2.

[22] A. Ranganathan, ―An Task-Execution Framework for Autonomic Ubiquitous Computing‖ Phd Thesis, University of Illinois at Urbana-Champaign, 2005.

[23] nternetworking Technology Handbook Cisco Publication Chapter 5