International Journal of Computer Applications (0975 – 8887)
Volume 139 – No.8, April 2016

'D-Mash' – A Concurrent Distributed Booking System

Shadrack M. Ngumbau
School of Computing and Informatics,
University of Nairobi, Kenya

Andrew M. Kahonge
School of Computing and Informatics,
University of Nairobi, Kenya

ABSTRACT

Today's world represents a community that values and appreciates unity and collaboration amidst distribution and diversity as key tenets to its development. A lot of work has been done on these by trying to guarantee proper communication amongst its distributed entities through the use of technology. However, challenges coming with the technology have been met one of them being to maintain concurrency in a distributed environment. In order to counter this concurrency challenge, a concurrent distributed system prototype implemented in a distributed environment to solve a real-world bus seat booking problem besides helping in assessing the impact of automated distribution and concurrency on Service Quality, is presented. In order to achieve this, the action research design was employed where the prototype was introduced and used in 1 Bus Company in a sample of 18 reputable companies after which 5 officers in a sample of 8 were interviewed. From the data collection and analysis done before and after introducing the prototype, it was found out that the probability of double-booking a seat in a distributed environment was reduced to 0 hence achieving concurrency which improved the service quality.

General Terms

Distributed Computing, Concurrency, Integration

Keywords

Distributed Systems, Action Research, Web Services, Concurrency Control, Middleware, Synchronization

1. INTRODUCTION

In relation to the past, today's era is characterized with communication and collaboration amongst distributed entities with fewer communication hitches. This is because of the ongoing research and implementation of Distributed Systems. However, issues such as real time computing, fault tolerance, synchronization amongst others, remain to be a challenge [1]. To counter synchronization, there is need to incorporate concurrency within distributed systems. It is with spirit that Kenyans have gone to an extent of using mobile phones within distributed environments e.g. the transport sector in order to achieve concurrency.

This is done in the bus industry by booking officers concurrently booking seats for customers through the use of mobile phones. This has worked to some extent although some difficulties and flaws in communication have at times led to a seat being booked twice unexpectedly hence interfering with one's planned journey and in turn leads to payment of damages and degradation of the bus company's service quality. Therefore, it is imperative that a solution be developed to solve the concurrency problem in a distributed environment.

2. PREVIOUS EFFORTS

In line with solving the above mentioned problem, development of interfaces to enable information exchange between different systems has been previously proposed [2]. Further, an integration architecture based on web services can be used to integrate business processes [3].

Also, some companies have gone to an extent of providing a self-service option where customers who only have mobile phones and chip cards registered with mobile service providers can book seats online and pay via electronic money transfer services. From experience and data collected from the sector e.g. from the Bus Company involved in this study, cases of one's journey being rescheduled or tampered with are still present due to the use of manual procedures and mobile phones to achieve concurrency.

Several bus companies use these procedures which have previously helped although with some cases of unintentionally disappointing clients due to double-booking. This leads to synthesis of the suggested methods and other concurrency distributed systems' techniques in this study in order to provide an internal solution that can enable concurrent booking in a distributed environment which in turn leads to an impact on service quality.

3. METHODOLOGY

An action research approach was adopted to help fulfill two main aims in the study [5].These aims entailed real-world problem solving and assessing the impact of the prototype on service quality. The figure below represents the 2 aims.

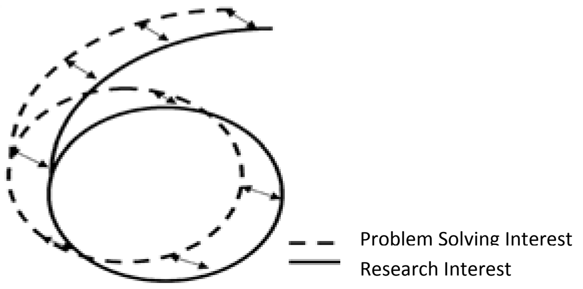

Fig.1: 'Action research viewed as a dual cycle process' (McKay and Marshall, 2001: p.52)

An amalgamation of concurrency and distribution techniques led to development of a prototype whose applicability was tested in a distributed bus booking context. Action research (AR) was considered as an appropriate approach to help in determining the suitability of concurrency and distribution techniques hence practice informing research and research informing practice by having the two complementing each other synergistically [4]. AR combines theory and practice through change and reflection in an immediate problematic situation within a mutually acceptable ethical framework [4]. It's through this combination that a distributed system prototype was developed, introduced, and reflection on its strengths and weaknesses done. The prototype was developed using the architecture represented in the figure below.

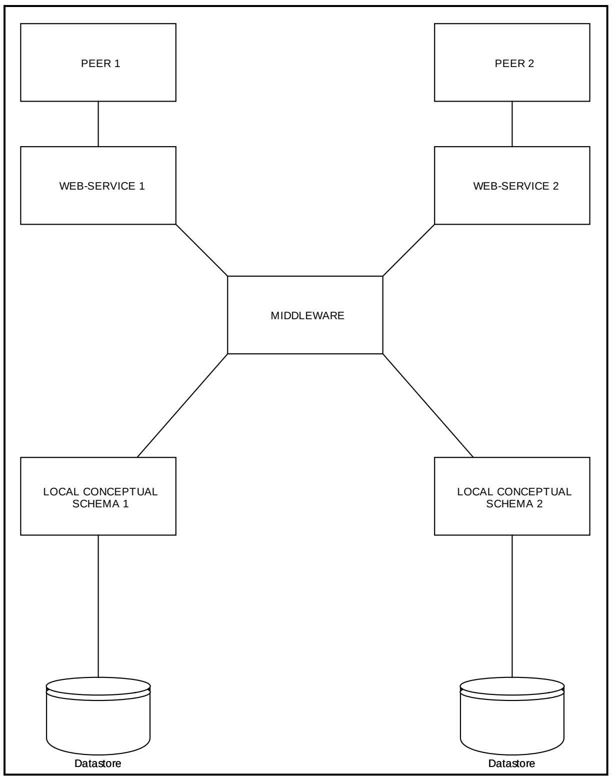

Fig 2: System's Architecture

Above is the distributed database management system's **architecture** with a **distributed database** (a collection of multiple, logically interrelated databases distributed over a computer network) whose data has been physically distributed and stored in computers communicating over a network.

The architecture has interconnected components that work together to achieve **heterogeneity and distribution**, **data fragmentation and replication**, and **distributed concurrency**.

In distributed systems, hardware and software are heterogeneous. The database is also distributed over the distributed nodes connected using a computer network. Every terminal represents a site/branch/booking station with its own autonomy and heterogeneity in the sense of its implementation being of different technologies compared to its peers'.

The physical data organization on each Data store may be different in that it could be using PostgreSQL's data organization structures while the other could be using MySQL's. Whilst the data in a distributed database is spread across the nodes, some of it needs to be fragmented and some needs to be replicated such that replicas of it exist in the distributed sites. Fragmentation entails having data localized such that data about a particular entity is only stored at its local site rather than having it stored in the other distributed sites unnecessarily. The distributed database is hence fragmented, replicated/duplicated, and with transparency in that a user can pose a query and interact with it faster via a distributed system's interface without wanting to know the location, fragmentation, and replication of the data but instead lets the system resolve that.

The system architecture's interconnected components include the middleware, web services, and conceptual schemas.

3.1 Web Services

Web services, which refer to traditional services offered online via the internet, adhere to standards enabling them to be discovered and used by clients conforming to the same standards [6]. Their main goal is to enable exchange of information amongst distributed applications [7].

They're based on XML (Extended Mark-up Language) which is based on SOAP (Simple Object Access Protocol) to enable them connect and communicate with heterogeneous programs launched in different computers [8]. A web service is hence a software application with a stable API (application programming interface) that's made available to consumers [9].

3.2 Middleware

From the system architecture figure above, the middleware links the web services and local schemas. A distributed system is often organized by means of a layer of software known as middleware between applications/clients and servers/operating systems [6]. In this architecture, it is an important element for the distributed and networked application because middleware solutions provide support for functional and non-functional requirements [10].

3.3 Conceptual Schemas

The local conceptual schemas represent each local data store/site. They are logical structures of the sites and through which queries are serviced. A combination of all local conceptual schemas creates a global conceptual schema. The local conceptual schemas are important for they also help in handling data fragmentation and replication. They provide location, replication, and network transparencies.

In order for the three to work together, **integration** was done amongst them. It is through integration that different organizations with different business processes and workflow are able to exchange information through web methods architecture. The integration adopted here entails use of the service-oriented architecture whose one of its important goals, as a solution to data type heterogeneity, is to improve interaction between organizations [12]. The service-oriented architecture enables easy customization and upgrading of existing applications which has led to rapid solutions to integration problems faced by organizations [13].

4. GENERAL PROCEDURE

The booking officer receives a client and asks for an identification e.g. the names and phone number. The booking officer proceeds to ask the customer of the date and time that he/she intends to travel after which a bus' chart / manifest for that type of booking is retrieved to check availability of a seat that hasn't been booked yet.

The middleware receives a chart retrieval request via a branch's web service. A global query is generated during this retrieval which is then translated into a group of local queries directed to the respective data stores in the branches. This is done through the help of the middleware.

Each branch receives a local query regarding available bus seats on a particular date and time. Each branch retrieves through SQL SELECT statements and responds accordingly via a web service connecting the middleware to the branches. The middleware combines the responses and responds to the branch that had requested the chart. This chart retrieval implementation was also tested through a combination of servlets and web service which was found to be fast enough as compared to the use of web services only.

The booking officer presents the chart to the customer via a screen so as for the customer to determine a seat of his / her choice that's available for booking. At this point, a level of location transparency is ensured for the specific location of the seat in the branches' data stores is not known to the customer. The booking officer proceeds to book the seat in the process of which the distributed system ensures concurrency through the use of buffers and assignment of locks to processes regardless of the time the transaction was initiated. A booking process that has acquired and using a lock is given priority. It's through this that a booking officer booking the same seat from a different location is given priority upon his/her triggering of a process that acquires a lock before the others.

As detailed in the branch's DFD (Data Flow Diagram) shown below, the booking officer records the details, books the seat, and proceeds to issue a ticket upon payment confirmation.

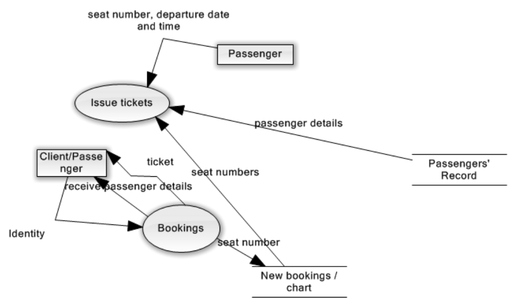

Fig 3: Branch's Data Flow Diagram

In addition, the customer can confirm his / her booking details upon losing a ticket by providing the seat number, departure date, and departure time.

5. RESULTS

Data collection methods i.e. interviewing, data archival, and participatory observation were used leading to quantitative and qualitative results. A quasi-experiment was done in collaboration with Imani Coach Bus Company during which the system prototype was deployed and data was collected before and after introduction of the prototype. The pre and post-test results are hereby presented.

5.1 Pre-test results

Interviewing and data archival were used to conduct the pre-test phase where it was found out that 2 to 3 customers complained of double-booking per month and that double-booking would happen **frequently** especially when there is demand of the service. It was also found out that an **average** of 44 seats was booked daily. This was calculated from a sample of 2543 seats that were booked in 58 days. Amongst the 58days, a 10-day sample's pre-test descriptives is illustrated using a column bar graph as shown below. It illustrates the seats that were booked within the 10 days and the probability for double-booking.

Table 1.1: Pre-test descriptives

Descriptive	Statistic
Mean	40.60
Median	41.00
Variance	52.49
Standard deviation	7.24
Minimum	22.00
Maximum	48.00
Range	26.00
Interquartile Range	6.50
Skewness	-2.07
Kurtosis	5.55

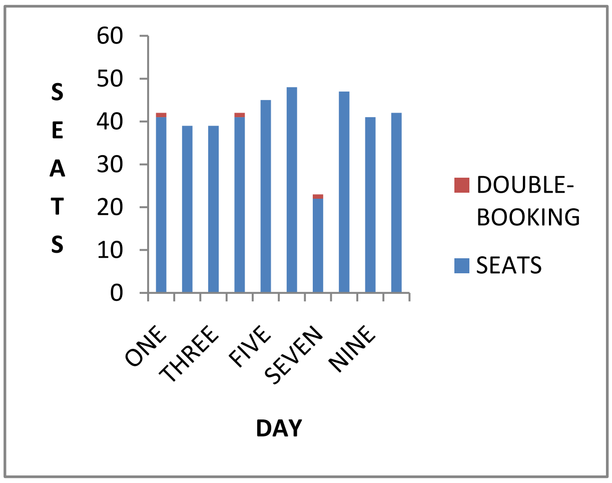

Fig 5: Pre-test distribution

Further, the probability of 2 to 3 out of 1257 seats (per month) being double booked, due to a concurrency challenge is illustrated in the following figure. It illustrates probabilities of 0.001591 (2/1257) and 0.002387 (3/1257) respectively.

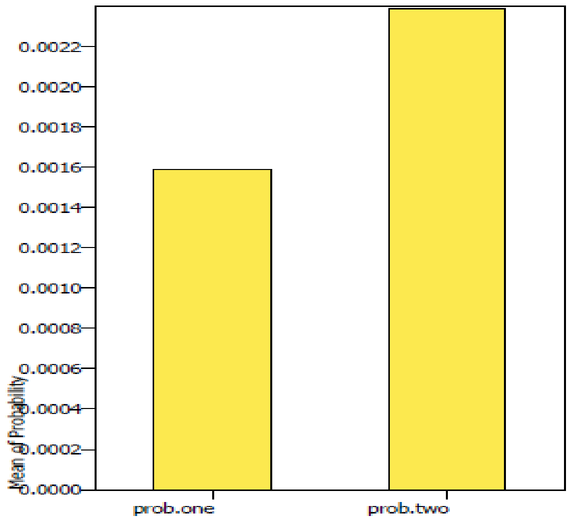

Fig 6: Pre-test probabilities

Below are some pre-test interview questions and answers:

Q: “Does the double-booking scenario happen intentionally? Please explain.”

A: “It happens unintentionally. Truly, it isn’t a good thing.”.

Q: “How easy or hard is it to deal with this situation? Please explain. Have you tried to avoid it before? What do you do to try avoiding it and does it still occur even when trying to avoid it?

A: “It isn’t easy to deal with the situation for the passenger is usually disappointed upon us abruptly interfering with his journey yet he/she had prepared himself/herself and booked the seat early enough. We try to avoid it by trying to be keen. However, it’s usually difficult to maintain that level of keenness hence we find it happening even as we try to avoid it.”

As part of participatory observation, it was found out that 1 client had no seat for it appeared that 10 seats were available yet the booking officer had been assured of 11 seats being available through a phone call. However, the booking officer anticipated double-booking from experience and helped the customer through reservation of an extra seat.

5.2 Post-test results

These showed that **0** customers complained of double-booking within the testing period and that **0** seats were double-booked using the system.

It was also found out from the records that an average of 45 seats was successfully booked from a sample of 449 seats that were booked in 10days. The sample’s descriptives are illustrated using a column bar graph as shown below. It illustrates the seats that were booked within the 10 days and the **0** probability for double-booking.

Table 1.2: Post-test descriptives

Descriptive	Statistic
Mean	45.30
Median	48.00
Variance	23.79
Standard deviation	4.88
Minimum	33.00
Maximum	48.00
Range	15.00
Interquartile Range	5.25
Skewness	-2.14
Kurtosis	4.67

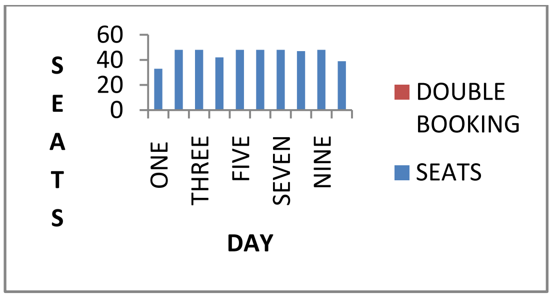

Fig 8: Post-test distribution

Further, the probability of **0** out of 315 seats (per week) has been illustrated besides the earlier demonstrated pre-test probability of 0.001591 (2/1257) in the following figure.

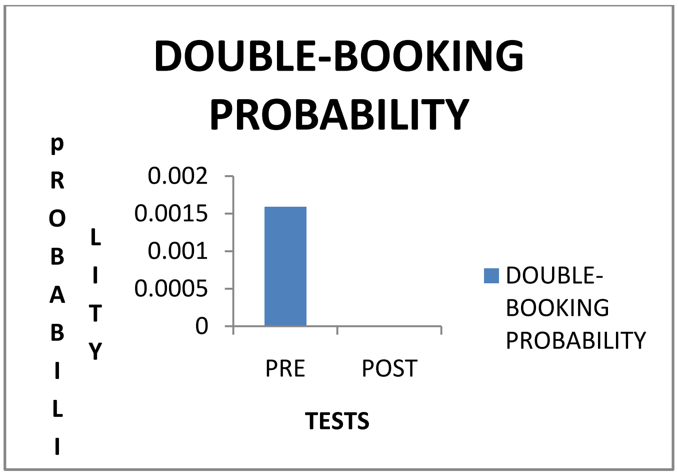

Fig 9: Pre and post-test double-booking probabilities (comparison)

Below are some post-test interview questions and answers:

Q: “Would you recommend such a system to any other organization and why?”

A: “Yes, I would recommend it for use elsewhere. This is because we have tested it and found it to have been good.

Q: “How possible is it now for you to concurrently book a seat without any case of double-booking?”

A: “Yes, it’s possible. A seat will only be available for booking to the first person.”

6. CONCLUSION

With the help of the results above and conceptual framework below, it came out that a probability of double-booking existed hence pre-test results generally suggested presence of a double-booking problem which occurred unintentionally and was accompanied by unpleasing consequences.

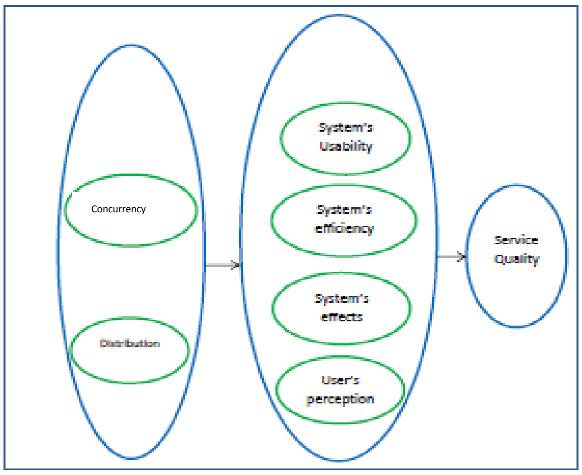

Fig 11: Conceptual Framework

That was because of the booking officers trying to concurrently book a seat from different locations without the help of distributed system software. The post-test results generally suggested that the prototype solved the double-booking problem.

That was because of introducing automated distribution and concurrency programming techniques. The bar chart plotting the pre and post-test probabilities clearly shows that the probability of double-booking was reduced to zero hence the booking officers ended up successfully concurrently booking a seat from different locations with the use of the system prototype.

In order to complement the above, grounded theory coding was done on the basis of responses from the interviews where several concepts that led to 4 categories were discovered. These led to the theoretical model below that led to a positive confirmation of the above conceptual framework's contribution. The concepts were difficulty, satisfaction, task time, automation, synchrony, contribution, experience, and outcome while the categories were system's usability, system's efficiency, user's perception, and system's effects. These are illustrated in the figure below.

Table 1.3: Theoretical Model

Independent variables	**Concepts**	**Categories/Intervening variables**	**Dependent variables**
Concurrency and Distribution	Difficulty	System's usability	Service Quality
	Satisfaction		
	Task time		
	Automation	System's efficiency	
	Synchrony		
	Contribution	Users' perception	
	Experience		
	Outcome	System's effects	

Results' evaluation concisely supported the possibility of concurrently performing a transaction in a distributed environment using the above presented techniques hence improving on the quality of service rendered to customers. This translates to a positive effect and growth on the business entity's business processes.

With limitations notwithstanding (e.g. absence of 2 separate control groups, the distributions having a significant level of Skewness, other organizations having not been in a position to test the prototype during the research), adoption of the action research (AR) method is hereby recommended as a practical problem-solving and research method for it was of great help towards achieving the research goal in not more than 1 iteration cycle. The iteration cycle followed the steps illustrated in the figure below.

Further, it is through AR that this research work hereby recommends and provides a practical successfully tested solution in addition to demonstration of a framework and prototype that can be adopted in future. Moreover, the evaluation advocates for the use of automated concurrency and distribution techniques in our systems so as to better our services and avoid some unpleasing incidences which hinder a country's holistic growth.

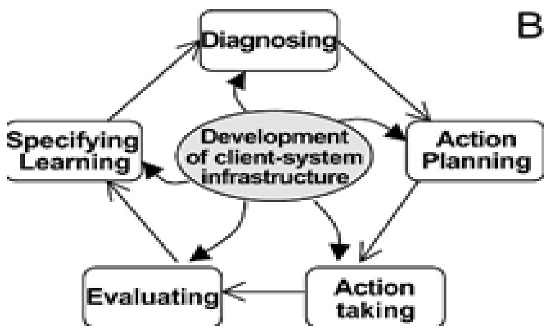

Fig.13: 'Representations of the action research cycle' (McKay and Marshall, 2001: p.49) `

It is hereby exhibited and backed that distributed systems' technologies are important and can be leveraged on to improve on the quality of services provided in daily commitments towards good progress. Through comparison of the service quality before and after introduction of distributed technologies, the results suggest that automated concurrency and distribution have a positive effect on service quality.

6.1 Further work

Further research can be carried out in Research work on distributed systems' ability to automatically update the fragments upon resumption of the network connection between the distributed entities and middleware. This can also be referred to as the ability to being autonomic. This was done manually where the system would be prompted and one would be prevented from booking a seat in a remote fragment if the seat had already been booked before there happened to be a network halt.

Additionally, it can be carried out in Research work on discovery of reliable and affordable network linkage that would support the achievement and success of distributed systems in a real-time environment e.g. in nuclear power generation firms and hospitals. This will help in stabilization of distributed systems as opposed to experiencing network halts which sometimes have an adverse effect on service provision.

7. ACKNOWLEDGEMENT

The authors wish to thank Imani Coach Bus Company booking officers for their collaboration towards solving the problem encountered by the company and for their support towards assessment of the concurrent distributed booking system prototype.

8. REFERENCES

[1] Casavant, T.L. (eds). (1994) *Readings in Distributed Computing System*. 1st edition. United States of America: IEEE Computer Society Press.

[2] GULLEDGE, T. (2006) What is Integration? *Industrial Management and Data Systems.* [Online] 106 (1) pp.5-20 Availablefrom:http://www.avyg86.dsl.pipex.com/ecom/0291060101.pdf [Accessed 07th July 2014].

[3] SURUGIU, I. (2012) Integration of Information Technologies in Enterprise Application Development. *Database Systems Journal.* [Online] 3 (1) pp.21-31 Available from: http://www.dbjournal.ro/archive/7/7_3.pdf [Accessed 07th July 2014].

[4] AVISON, D., LAU, F., MYERS, M., and NIELSEN, P. (1999), Action Research. *Association for Computing*

Machinery. Communications of the ACM. [Online] 42 (1) pp.94-97 Available from: http://www.ic.unicamp.br/~wainer/cursos/2s2006/epistemico/act-acm.pdf [Accessed: 11th June 2015].

[5] McKAY, J. and MARSHALL, P. (2001) The Dual imperatives of action research. *Information Technology & People.* [Online] 14 (1) pp.46-59 Available from: doi:10.1108/09593840110384771 [Accessed: 10th June 2015].

[6] TANENBAUM, A.S. (eds). (2007) *Distributed Systems: Principles and paradigms*. 2nd edition. United States of America: Pearson Prentice Hall

[7] MUMBAIQAR, S., and PADIYA, P. (2013) Web Services based on SOAP and Rest Principles. *International Journal of Scientific and Research Publications.* [Online] 3 (5) pp.1-4 Available from: http://www.ijsrp.org/research-paper-0513/ijsrp-p17115.pdf [Accessed 04th July 2014].

[8] GASHTI, M.Z., (2012) Investigating SOAP and XML Technologies in web service. *International Journal on soft computing.* [Online] 3 (4) Available from: http://airccse.org/journal/ijsc/papers/3412ijsc02.pdf [Accessed 05th July 2014].

[9] ALONSO, G., CASATI, F., KUNO, H. AND MACHIRAJU, V. (2004) Web Services. In: ALONSO, G., CASATI, F., KUNO, H. AND MACHIRAJU, V. (eds.) *Web Services – Concepts, Architectures and Applications.* Springer Verlag. pp. 123-149 [Online] Available from: http://ahvaz.ist.unomaha.edu/azad/temp/softarch/04-alonso-webservices-server-architecture-soa.pdf [Accessed 05th July 2014].

[10] F.M GHALEB, and REDA N.M (2012) Open-Gate: An Efficient Middleware System for Heterogeneous Distributed Databases. *International Journal of Computer Applications.* [Online] 45 (2) pp.44-49 Available from: http://research.ijcaonline.org/volume45/number2/pxc3879009.pdf[Accessed 05th July 2014].

[11] SCHMIDT D.C, (2002) Adaptive and Reflective Middleware for Distributed Real-time and Embedded Systems. In: VINCENTALLI, S.A.and SIFAKIS, J. (eds.) *Embedded Software.* Springer Verlag. pp. 282-293 [Online] Available from: doi:10.1007/3-540-45828-x_21 [Accessed 05th July 2014].

[12] HASHEMI, S. and HASHEMI S.Y. (2012) A Novel Service Oriented Architecture for integration of Information Systems in electronic city. *International journal of scientific and Technology Research.* [Online] 1 (11) pp.6-9 Available from: http://www.ijstr.org/final-print/dec2012/A-Novel-Service-Oriented-Architecture-For-Integration-Of-Information-Systems-In-Electronic-City.pdf[Accessed 07th July 2014].

[13] KESTER, Q. (2013) Using SOA with Web Services for effective Integration of Hospital Information Systems via an Enterprise Service Bus. *International Journal of Research in Engineering & Advanced Technology.* [Online] 1 (2) Available from: http://arxiv.org/ftp/arxiv/papers/1307/1307.7790.pdf [Accessed 07th July 2014].