A Generalized Service Replication Process in Distributed Environments

Hany F. ElYamany[1], Marwa F. Mohamed[1], Katarina Grolinger[2], and Miriam A. Capretz[2]
[1]*Faculty of Computers and Informatics, Suez Canal University, Old Campus, Ismailia, Egypt*
[2]*Department of Computer Engineering, Western University, London, ON, Canada*
{hany_elyamany, marwa_fikry}@ci.suez.edu.eg, {kgroling, mcapretz}@uwo.ca

Keywords: Service replication, service-oriented architecture, cloud, mobile computing, replication process, quality of service

Abstract: Replication is one of the main techniques aiming to improve Web services' (WS) quality of service (QoS) in distributed environments, including clouds and mobile devices. Service replication is a way of improving WS performance and availability by creating several copies or replicas of Web services which work in parallel or sequentially under defined circumstances. In this paper, a generalized replication process for distributed environments is discussed based on established replication studies. The generalized replication process consists of three main steps: sensing the environment characteristics, determining the replication strategy, and implementing the selected replication strategy. To demonstrate application of the generalized replication process, a case study in the telecommunication domain is presented. The adequacy of the selected replication strategy is demonstrated by comparing it to another replication strategy as well as to a non-replicated service. The authors believe that a generalized replication process will help service providers to enhance QoS and accordingly attract more customers.

1 INTRODUCTION

Nowadays, the Web is structured as a mesh of heterogeneous distributed environments including service-oriented architecture (SOA) (Erl, 2008), cloud computing (Erl *et al*., 2013), and mobile computing (Fling, 2009). Web services have a key role in managing and encapsulating business processes inside such environments. Web services are described, discovered, published, and executed using standard protocols such as WSDL for service description, SOAP for message exchange, and UDDI for service registry and discovery (W3C, 2004; Papazoglou, 2008).

Quality of service (QoS) (Al-Masri *et al*., 2007; W3C, 2003) is a significant factor in describing and establishing the service level agreement (SLA) among service providers and consumers. In particular, the SLA is an official contract between the provider and the consumer which specifies non-functional requirements, specifically focussing on performance and availability (Michlmayr *et al*., 2009; Papazoglou *et al*., 2005). Web service replication is a way of improving WS performance and availability by creating several copies or replicas of Web services which may work either in parallel or sequentially under defined circumstances and regulations (Salas *et al*., 2006; May *et al*., 2009).

This paper introduces a generalized replication process in distributed environments with the objective of helping service providers select and implement an appropriate service replication strategy and consequently increase the quality of service provided. The discussed replication process consists of three main steps: sensing the environment characteristics, determining the replication strategy, and implementing the selected replication strategy. A mobile authentication case study is presented to demonstrate the use of the approach.

The rest of this paper is organized as follows: Section 2 introduces the replication literature review. Section 3 introduces a generalized replication process for distributed environments. Section 4 presents the evaluation case study. Finally, Section 5 concludes the paper.

H. F.. ElYamany, M. F. Mohamed, K. Grolinger, M. A. M. Capretz, A Generalized Service Replication Process in Distributed Environments, Proc. of the 5th International Conference on Cloud Computing and Services Science, Lisbon, Portugal, 20-22 May, 2015.

2 REPLICATION LITERATURE REVIEW

Two main replication types may be distinguished depending on whether the number and location of replicas change during runtime: static replication and dynamic replication. In *static replication*, the predefined replica communication group does not change during runtime; when a single replica becomes unresponsive, this replica is still considered a member of the communication group. In other words, the number and position of replicas are fixed over time (Guerraoui *et al*., 1997). *Static replication* is planned at design time (Słota *et al.*, 2005) according to predefined parameters, and implementation is carried out regardless of any changes that may occur during runtime.

Dynamic replication supports a changing number of replicas, changes in physical locations, and selection of running replicas during runtime (Keidl et al., 2003; Mohamed et al., 2013). It is performed in two different styles: *Dynamic replica selection* (Thakur *et al*., 2012) and *Dynamic replica placement* (Mohamed et al., 2013; Dustdar, 2007).
The replication process can be implemented using different techniques depending on the components involved and their characteristics. Salas *et al*. (2006) classified the replication process into three techniques according to the interactions among replicas and requests: *active*, *passive* and *semi-active techniques*. Like Salas *et al*. (2006), May *et al*. (2009) also recognized three categories, but the categories are different: *parallel, serial* and *composite techniques.* Zheng *et al*. (2008) expanded the replication techniques from the work of Salas *et al*. (2006) by combining active, passive, and time-replication techniques. Time replication means that a particular service is invoked a finite number of times before its status is declared as failed. Liu *et al*. (2011) took a very different approach and generated a diverse group of replication techniques using a directed acyclic graph (DAG), which are characterized as active, passive, hybrid, active-passive, and passive-active. In their study, they produced a graph model to represent a replication scheme defined as a directed acyclic graph DAG, G ≡ (V, E), where the vertex set V refers to a set of WS replicas and the directed edge set E refers to the replica invocation. In this approach, the directed edges capture a replication schema.

Several researchers have proposed different replica selection strategies and algorithms (Sayal *et al.*, 1998; da Silva *et al*., 2004; Björkqvist *et al*., 2012). Sayal *et al*. (1998) described six replica selection algorithms: Fixed, Ping, Hops, Parallel, Probabilistic, and Refresh. Da Silva *et al*. (2004) presented five server selection policies: Random Selection, Parallel Invocation, HTTPing (or Probe), Best Last, and Best Median. Finally, Björkqvist *et al.* (2012) defined two replica selection algorithms: Distributed Shortest Queue Selection (D-SQ) and Distributed Round Robin Selection (D-RR).

Although extensive efforts have been made in both academia and industry in the area of service replication, we are not aware of any studies that discuss a generalized replication process. The approach introduced in this paper builds on diverse service replication research to create a generalized replication process with the objective of helping service providers increase QoS.

3. GENERALIZED REPLICATION PROCESS

Distributed environments such as service-oriented architecture (SOA), cloud computing, and mobile computing typically use replication technology to improve operational characteristics, including availability and performance. Unfortunately, replication encounters challenges in these environments.

3.1 Generalized replication process: overview

To help practitioners determine the most suitable replication approach for a specific scenario, a generic replication process is needed. This section introduces a generalized replication process for distributed environments based on the replication approaches presented in the previous sections. Specifically, the findings from the reviewed studies are integrated to form a generic replication process.

Figure 1 illustrates the use-case scenario: a client demands a service through a service provider, where the target service may be located and published in a cloud, SOA, or mobile environment. If the service provider (typically the business service provider or service owner) on behalf of the client(s) or consumer(s) detects a delay in answering incoming requests due to technical problems such as resource failures, service(s) overload, or network issues, the service provider should find a solution to speed up the answering process using service replication. How this replication process should be implemented will vary with respect to several metrics, including the characteristics of the host environment.

Figure 1: Replication environment interactions

For example, in the cloud, the service provider could deploy a replica in any location where there are no technical challenges, whereas in a mobile environment, the nearest device or station may need to be selected to host the required replica.

Four major actors must be considered when designing a generic replication process for different distributed environments: consumers, service providers, replication actors, and the environment. As shown in Fig. 2, the replication process is divided into four layers representing these actors and their behaviour:

- *The consumer layer* represents the users who are seeking the published services. A service-level agreement (SLA) is enforced when the consumer binds with the target service. The service provider must work within the signed SLA to achieve the QoS terms listed in the SLA and attain high consumer satisfaction and loyalty.
- *The service provider layer* consists of the services or business owners. It continuously interacts with the replication and environment layers to host, publish, and enhance the business services that it owns or works on. Service providers monitor and update their services to keep current consumers and attract new consumers.
- *The replication layer* maintains the QoS parameters as defined in the SLA by providing additional replicas as needed.
- *The environment layer* could be a cloud, SOA, or mobile environment; in this layer, services or businesses are located and accessed. Moreover, this layer might be called the infrastructure layer because it contains all required physical and logical resources to host the services created by the various providers and targeted by consumers.

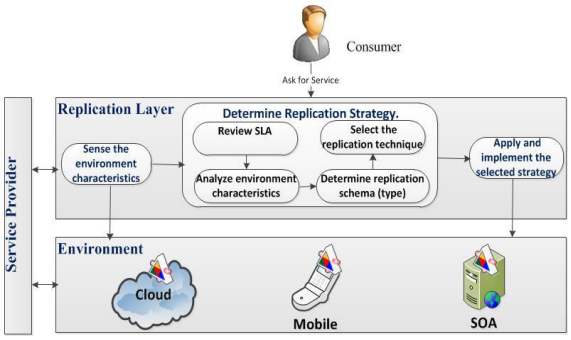

Figure 2: Generalized replication process

3.2 Generalized replication process: steps

Figure 3 represents the basic interconnections between the defined replication process activities in Fig. 2 and the main actors or layers: the consumer, service provider, replication, and environment layers. To achieve a robust replication process, the following steps should be followed, as shown in Fig. 3:

1) Sense the environment characteristics: this activity occurs between the replication and environment layers to obtain the current status of the consumed services in terms of QoS characteristics.

In addition, the capabilities of the available resources in terms of functional characteristics are investigated. For example, this activity may search for a trusted server in a cloud to host a new replica or assign the nearest node to run the new replica in a mobile environment. Examples of the detected characteristics of the various environments are given in Table 1. The information obtained is saved in a database located inside the environment layer to be used in step 2.

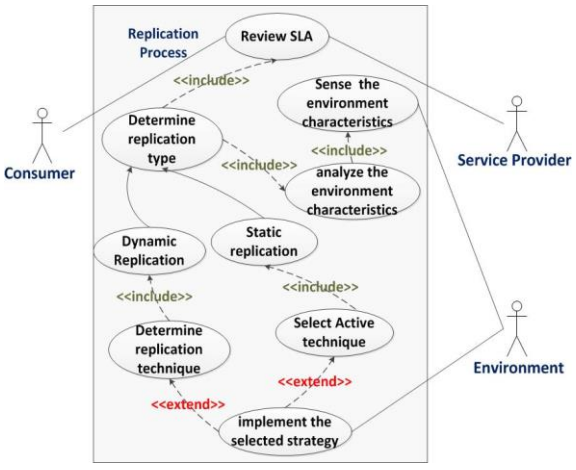

Figure 3: Replication process use-case model

Table 1: Detected environmental characteristics

Environment	Functional characteristics	Non-functional characteristics
SOA	Number of hops Number of servers	Server performance Server availability Type of WS: composite or basic.
Cloud	Number of replicas Trusted servers	Service performance Service availability Ensure service consistency. Type of service: deterministic or not
Mobile	Mobile battery power Number of replicas Nearest node	Service availability Network traffic

2) Determine the replication strategy: This step is composed of the four steps as shown in Fig. 2, which can be described as follows:

a. Review SLA: this activity checks the signed SLA contract periodically or when a change in quality is detected by environmental sensing with respect to the defined non-functional terms specified in the SLA. The SLA contract could be maintained inside the service provider layer or in the environment layer based on the agreement between providers, consumers, and environment owners. First, the SLA items are reviewed against current environment characteristics to determine the replication target (e.g., availability or performance) on which the replication layer must focus when a violation occurs. Then it notifies the provider and the consumer of the result. In addition, it retrieves the environment and functional capabilities from the first step to determine the candidate host for the new replica.

Table 2: Expected strategy requirements

Replica Selection Algorithm	Algorithm description	← Replication Type	← Actions
Ping/Probe	The client periodically sends a ping request to all available replicas and then forwards request(s) to the replica with minimal ping round-trip time.	Dynamic	Service performance
Hops	The client sends requests to the nearest replica according to the number of hops between the replica and the client.	Static/Dynamic	Number of hops
Parallel	The client sends requests to all available replicas. The one which works on the incoming request first responds to request and communicate with the consumer directly.	Static/Dynamic	In dynamic case, service availability
Probabilistic	Replica selection is based on a probability that has been calculated and assigned to each replica. Probability is calculated based on SLA.	Static/Dynamic	SLA review
Refresh/Best Last	The client sends requests to the server with the minimal request latency. Latency samples are refreshed periodically.	Dynamic	Service availability and performance
Best Median	Replica selection depends on the lowest median response time among the set of successful invocations recorded for each replica used.	Dynamic	Service availability and performance
Shortest Queue	Service selection depends on the smallest number of queued invocations as determined by locally maintained statistics related to service activities.	Dynamic	Service availability and performance
Round Robin	A list of active replicas is maintained and updated periodically by adding /removing the newly activated /deactivated replicas. Upon service replica selection, the requests are rotated around a list of active replicas.	Static/Dynamic	In dynamic case, service availability
Random	Random replica selection	Static/Dynamic	In dynamic case, service availability

b. Analyze Environment Characteristics: this activity examines the environment characteristics collected in Step 1 to match them with the characteristics of the corresponding selection algorithm. For example, as shown in Table 2, if the Hops replication selection algorithm is selected within a dynamic replication strategy, the number of hops should be considered and estimated. At the end of this activity, the defined replication actions are saved in a database to be considered in the implementation activity.

c. Determine Replication Type: Depending on the data collected from the previous activities, the detection process determines the type of replication, as shown in Table 3. This activity obtains the required information from the SLA Review and Analyze Environment Characteristics processes to make a decision about the replication type. Section 2 shows two different types of replication that can be used: static and dynamic replications. In static replication, only the active replication technique can be used. But within dynamic replication, all replication techniques can be used.

Table 3: Data needed to determine the replication type

Environment	Target	Characteristic	Replication Type
SOA	Availability Performance	Uses a fixed number of replicas Servers have average load performance	Static
	Availability Performance Responsiveness	The number of replicas is needed during runtime Variable servers	Dynamic
Cloud	Availability Security	Services are deterministic Multicast consumer requests	Static
	Availability Security	Services are deterministic or nondeterministic Multicasting not required	Dynamic
Mobile	Availability	Always runs in a dynamic environment	Dynamic

d. Select Replication Technique: Once the replication type and target have been determined, the process moves on to selecting the replication technique. Table 4 shows how the replication strategy is selected and composed (Steps 2a to 2d) based on the replication target, replication type, and selected replication technique.

Table 4: Composition of the replication strategy

Replication Type →	Replication Target →	Replication Techniques	Replication Strategy
Static	Availability	Active	Parallel
	Performance	Static load balancing	Round Robin Probabilistic
Dynamic	Availability	Active – Passive – Semi-Active	Ping or Probe, Refresh, Best Last, or Best Median.
	Performance	Dynamic load balancing	Distributed Round Robin Selection
	Responsiveness	Dynamic load balancing	Weighted Round Robin

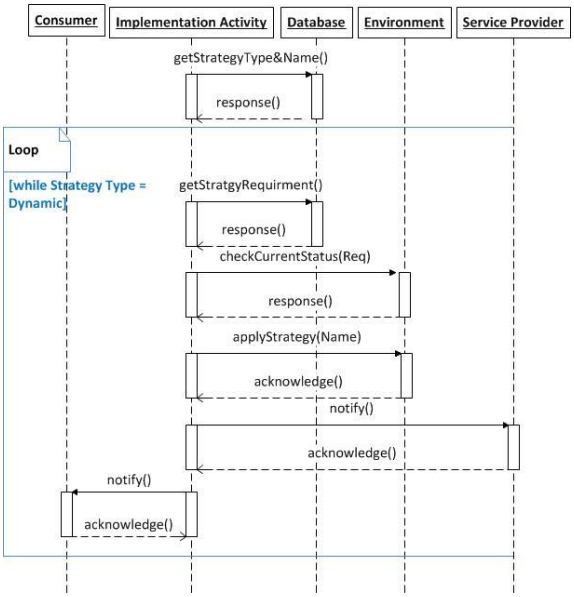

Figure 4: Sequence diagram for the Implement the Selected Replication Strategy activity

3) Implement the Selected Replication Strategy*:* The sequence diagram shown in Fig. 4 illustrates how this process is executed. When a replica is needed, the implementation process obtains the strategy type and name to be used from the database. Basically, it verifies the selected strategy requirements against the current state of the deployed environment to ensure a correct replication process. Once the implementation has been accomplished successfully, a notification is sent to both the provider and consumer that their agreed QoS terms are still being achieved as expected.

4. MOBILE AUTHENTICATION CASE STUDY

This section describes a scenario from a telecommunication company in which a huge number of cell phones are authenticated when connected to the company network through a particular Web service called *mobile authentication service*. Eventually, this service lacks availability and/or performance during peak times. In this paper, the steps of the introduced replication process are used to overcome the availability and performance challenges for this service. The architecture of the mobile authentication service environment can be divided into three layers: *The mobile layer* represents the users who are seeking mobile authentication service. *The portal layer* contains the replication management middleware which controls the interaction between consumers and the WSs replicas. *The cloud layer* contains all the physical and logical resources required to host the services. In such an environment, the replication management middleware (Portal) will follow the suggested replication steps as described in Section 3:

1) Sense the environment characteristics: As outlined in Table 1, the replication management middleware will collect the information; specifically, for this use case, the collected information is listed in Table 5. This case study assumes that three WS replicas are running on one virtual machine, but are installed on different ports. The WS is deterministic because for a given input, it always produces the same output. The user enters username and password, and then the system responds with "successful login" or not.

Table 5: Mobile authentication case study characteristics

Environmental characteristics	General	Mobile authentication case study
Functional characteristics	Number of replicas	Three replicas used
	Trusted servers	One virtual server with Web services installed on three different ports
Non-functional characteristics	Performance	Required
	Availability	Required
	Ensure service consistency	Not required, because the WSs are retrieving data from a database. The case study makes no changes to data.
	Service type: deterministic or not	Deterministic service.

2) Determine the replication strategy consists of the following steps:

a. Review SLA: The replica management middleware reviews the SLA to determine the replication target. The target of this case study is ensuring Web service availability and performance.

b. Analyze Environment Characteristics: the replication management middleware will analyze the strategic requirements depending on the environmental characteristics and SLA target. The selection process can be managed by consumers or the service provider. In this case study, the service provider manages the selection process, and therefore the Best Last and Best Median methods are ignored because they depend on consumer preferences; the consumer selects the replicas with lowest response time or median lowest response time depending on the invocation history.

Moreover, in this case study, all Web service replicas are placed on one virtual machine, and therefore the Hops and Probabilistic strategies are removed from the selection list. The Random strategy cannot provide the desired service availability because the failed copy may be selected, and therefore it is also removed. Moreover, the aim of the case study is to forward consumer requests to the best available WS; the load balancing carried out by Round Robin is eliminated. Hence, the selection list contains four strategies: Ping, Parallel, Refresh and Shortest Queue, as shown in Table 6.

Table 6: Analysis of case-study characteristics

Replica selection algorithm	Strategic requirements	Selection list
Ping/Probe	Service availability	√
Hops	Number of hops	X
Parallel	Service availability.	√
Probabilistic	SLA review, host performance history.	X
Refresh	Service availability and performance	√
Best last	Service availability and performance	X
Best Median	Service availability and performance	X
Shortest Queue	Service availability and performance	√
Round Robin	Service availability	X
Random	Service availability	X

c. Determine Replication Type: in this step, there are three options for replication type: static, dynamic service placement, and dynamic

service selection. Depending on the cloud row in Table 3, static replication is ignored, and dynamic replication is used. The mobile authentication service deals with a large number of users, so that multicasting of consumers' requests to provide service availability is undesirable because it may cause network failure. Moreover, all replicas are installed on a private cloud, and therefore dynamic service selection is chosen. Dynamic service replacement is preferable when multiple independent resources are available.

d. Select Replication Technique: according to Table 6, there are four choices: Ping, Parallel, Refresh and Shortest Queue. The Parallel strategy is dropped from the selection list because multicasting of consumers' requests is not supported in this case study. The Shortest Queue strategy is out of consideration because it selects the replica with the lowest load to achieve the shortest response time, which is the same target as the Refresh strategy. The Ping selection process is based on the WS host/port with the lowest ping round-trip time, but the Refresh strategy depends on the WS with the lowest response time. Therefore, the preferred option in this case is the Refresh strategy.

3) Implementation: A simulation of this environment was constructed and run using the specifications shown in Table 7.

Table 7: Specifications of simulation environments

Cloud	Google App Engine (Platform as a Service)
Portal	Apache/2.2.11 (Win32) PHP/5.3.0 Processors: Intel(R) core i3 Memory 4 GB
Mobile	Smart Phone

The implementation scenario can be described as follows:

- The replication management middleware (RMM) ensures that the cloud environment has three Web service replicas. If not, the middleware transfers the required replicas to the cloud environment.
- RMM notifies the users and the telecom company admin(s) to access the service. Users access authentication services. Each user types his/her username and password and presses Enter.
- RMM passes consumer requests to the best available mobile authentication service using the Refresh algorithm.
- The mobile authentication service processes the consumers' requests and sends a response back to replication management middleware. Then RMM forwards the results to the consumer.

The experiments were conducted using ApacheBench Version 2.0.40-dev by passing different consumer loads (1, 3, 5, 8) over 100 times (100, 300, 500, 800 requests). Then throughput and response time were recorded for three cases: the case without replication and the Ping and Refresh strategies. As shown in Figs. 5 and 6, the Refresh replication strategy provides better throughput rates than non-replication. The Refresh strategy passes consumer requests to the best available service, so that the WS used can be changed during runtime; this leads to a balance in incoming requests between replicas, but not in an equally likely manner.

In the case of the Ping strategy, before every request, a ping was sent to all ports, and the port with minimal round-trip time was selected. The selection depends on the round-trip time of the ping message, not on the service response time, and therefore it is no better than the Refresh strategy. However, it is better than the case without replication for high consumer loads (5, 8) because the port can be changed during runtime, so that load balancing occurs, but not in an equally likely manner.

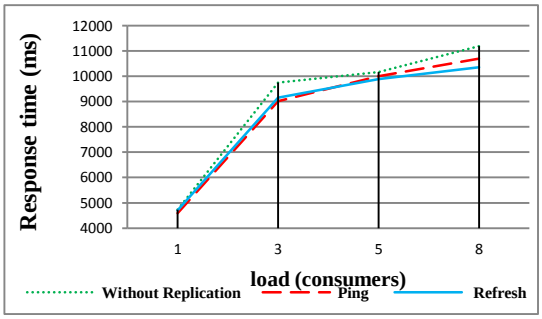

Figure 5: Response time when running 1700 requests

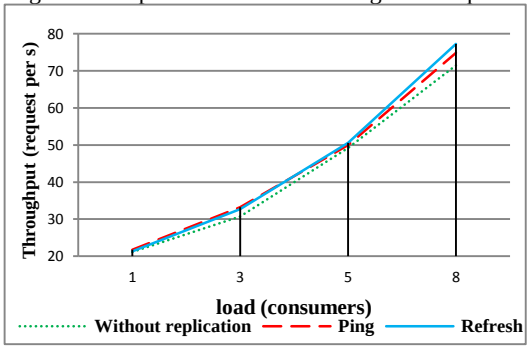

Figure 6: Throughput when running 1700 requests

5. CONCLUSIONS

In this paper, a generalized service replication process was introduced to manage and control replication inside distributed environments. The process consists of three steps: sensing the environment characteristics, planning a complete replication strategy, and implementing the selected replication strategy. The application of the generalized process is demonstrated in a case study involving a telecommunication scenario. The selected replication algorithm, the Refresh algorithm, was compared to the Ping algorithm and non-replicated service. Results show that the Refresh algorithm outperformed both Ping and non-replication in terms of throughput and response time.

Future work will include deploying the generalized replication process in a real-world environment and expanding the validation. In addition, it is planned to extend the review of the replication process to cover embedded systems and other distributed environments such as the Internet of Things (IoT) and cyber physical systems.

REFERENCES

Al-Masri, E. & Mahmoud, Q. H., 2007. QoS-based discovery and ranking of Web services. In *Computer Communications and Networks, 2007 (ICCCN 2007), Proceedings of 16th International Conference*, pp. 529-534. IEEE.

Björkqvist, M., Chen, L. Y., & Binder, W., 2012. Dynamic replication in service-oriented systems. *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGRID 2012)*, pp. 531-538. IEEE Computer Society.

da Silva, J. A. F. & das Chagas Mendonça, N., 2004. Dynamic invocation of replicated Web services. *WebMedia and LA-Web, 2004. Proceedings*, pp. 22-29. IEEE.

Dustdar, S. & Juszczyk, L., 2007. Dynamic replication and synchronization of Web services for high availability in mobile ad-hoc networks. *Service Oriented Computing and Applications*, vol. 1, no. 1, pp. 19-33.

Erl, T., 2008. *SOA: Principles of Service Design*, vol. 1. Upper Saddle River: Prentice-Hall.

Erl, T., Puttini, R., & Mahmood, Z., 2013. *Cloud Computing: Concepts, Technology & Architecture*. Pearson Education.

Fling, B., 2009. *Mobile Design and Development: Practical Concepts and Techniques for Creating Mobile Sites and Web Apps*. O'Reilly Media.

Guerraoui, R. & Schiper, A., 1997. Software-based replication for fault tolerance. *Computer*, vol. 30, no. 4, pp. 68-74.

Keidl, M., Seltzsam, S., & Kemper, A., 2003. Reliable Web service execution and deployment in dynamic environments. In *Technologies for E-Services*, pp. 104-118. Springer, Berlin, Heidelberg.

Liu, A., Li, Q., & Huang, L., 2011. Quality-driven Web services replication using directed acyclic graph coding. In *Web Information System Engineering (WISE 2011)*, pp. 322-329. Springer, Berlin, Heidelberg.

May, N. R., Schmidt, H. W., & Thomas, I. E., 2009. Service redundancy strategies in service-oriented architectures. *Proceedings, Software Engineering and Advanced Applications, 2009 (SEAA'09) 35th Euromicro Conference*, pp. 383-387. IEEE.

Michlmayr, A., Rosenberg, F., Leitner, P., & Dustdar, S., 2009. Comprehensive QOS monitoring of Web services and event-based SLA violation detection. *Proceedings, 4th International Workshop on Middleware for Service Oriented Computing*, pp. 1-6. ACM.

Mohamed, M. F., ElYamany, H. F., & Nassar, H. M., 2013. A study of an adaptive replication framework for orchestrated composite Web services. *SpringerPlus*, vol. 2, no. 1, pp. 1-18.

Papazoglou, M. P. & Van den Heuvel, W. J., 2005. Web services management: A survey. *Internet Computing, IEEE*, vol. 9, no. 6, pp. 58-64.

Papazoglou, M., 2008. *Web Services: Principles and Technology*. Pearson Education.

Salas, J., Perez-Sorrosal, F., Patiño-Martínez, M., & Jiménez-Peris, R., 2006. WS-replication: a framework for highly available Web services. *Proceedings of the 15th International Conference on World Wide Web*, pp. 357-366. ACM.

Słota, R., Nikolow, D., Skitał, Ł., & Kitowski, J., 2005. Implementation of replication methods in the grid environment. In *Advances in Grid Computing (AGC 2005)*, pp. 474-484. Springer, Berlin, Heidelberg.

Sayal, M., Breitbart, Y., Scheuermann, P., & Vingralek, R., 1998. Selection algorithms for replicated Web servers. *ACM SIGMETRICS Performance Evaluation Review*, vol. 26, no. 3, pp. 44-50.

Thakur, M. R. & Sanyal, S., 2012. A PAXOS-Based State Machine Replication System for Anomaly Detection. *arXiv Preprint, arXiv:1206.2307*.

W3C *Working Group Note: Web Services Architecture*, 2004. Available from: http://www.w3.org/TR/ws-arch/ [14 March 2015].

W3C *Working Group Note: QoS for Web Services: Requirements and Possible Approaches*, 2003. Available from http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/ [14 March 2015].

Zheng, Z., & Lyu, M. R., 2008. A distributed replication strategy evaluation and selection framework for fault tolerant Web services. *Proceedings, Web Services 2008 (ICWS'08) IEEE International Conference*, pp. 145-152. IEEE.