Fault Tolerance in Cloud Computing: A Survey

Abdeldjalil Ledmi
Departement of Computer Sciences.
Abbas Laghrour University of Khenchela
ICOSI Lab,
Khenchela, Algeria
abdeldjalil.ledmi@univ-khenchela.dz

Makhlouf Ledmi
Departement of Computer Sciences.
Abbas Laghrour University of Khenchela
ICOSI Lab,
Khenchela, Algeria
ledmi.makhlouf@univ-khenchela.dz

Mohammed El Habib Souidi
Departement of Computer Sciences.
Abbas Laghrour University of Khenchela
ICOSI Lab,
Khenchela, Algeria
souidi.mohammed@univ-khenchela.dz

2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI) | 978-1-6654-4171-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICRAMI52622.2021.9585903

***Abstract*—Using cloud computing services has many advantages, such as improving efficiency, reducing costs, compatibility with multiple formats, unlimited storage capacity, and easy access to services anytime and anywhere. It should be mentioned that the fault tolerance is the main restriction of all varieties of cloud computing services. Cloud service providers need to effectively handle performance-related reliability, availability, and throughput issues to maximize the potential of using cloud computing services.**

This paper provides a comprehensive overview of the issues related to fault tolerance in cloud computing. It focuses on important advanced technologies, and methods. Its purpose is to provide insight into traditional fault-tolerant approaches and the challenges that still need to be overcome. This investigation enumerates several promising methods that can be used for efficient solutions.

***Index Terms*—Cloud Computing, Fault Tolerance, Cloud Services.**

I. Introduction

According to the U.S. National Institute of Standards and Technology (NIST): *"Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (for example servers, networks, storage, services, and applications) that can be quickly provisioned and released with least management effort or service provider interaction"* [1].

Cloud computing provides end-users with various resources on-demand in the form of services. It allows companies and users to use applications without having to install them on physical machines and allows access to necessary resources over the Internet. It provides high performance, pay-as-you-go, connectivity, interactivity, reliability, efficiency, scalability, and other features [2].

As a fast-developing technology, cloud computing is increasingly used to host many services and business applications. However, the widespread use of cloud-based services to host business or enterprise applications can lead to service availability and reliability issues for service providers and users [3]. These problems are inherent in cloud computing due to its highly distributed characteristics, heterogeneity of resources, and large-scale. Therefore, multiple types of failures can occur in the cloud environment, leading to failures and performance degradation. The main types of failures are the following [4]:

- **Network fault**: cloud computing resources are accessed through the Internet, the main cause of cloud computing failure is a network failure. These failures may be caused by network partition, packet loss or damage, congestion, target node or link failure, etc.
- **Physical fault**: These failures mainly occur in hardware resources, such as CPU failures, memory failures, storage failures, power failures, etc.
- **Process fault**: may be caused by insufficient resources, software errors, etc.
- **Service expiration fault**: If the service time of the resource expires while the leased application is using it, it will cause the service to fail.

Fault tolerance refers to a system's capacity to continue performing its expected function despite defects. In other words, fault tolerance has to do with dependability, successful functioning, and the absence of breakdowns [5]. A fault tolerance based system should be able to investigate defects in specific software or hardware components, power outages, and other types of unanticipated adversities while still meeting the requirements.

The remainder of this survey paper is structured as follows: Section 2 provides an overview of the cloud computing environment. Section 3 discusses the traditional fault tolerance approaches used in distributed systems. Section 4 examines the current fault tolerance approaches in the cloud computing environment. Finally, section 5 presents the concluding remarks.

II. Basic Concept

Efforts in numerous computer research fields, such as distributed computing, grid computing, and virtualization technologies, have resulted in cloud computing. This section describes cloud computing in three dimensions: (A) Cloud infrastructure (B) Cloud deployment models (C) Cloud service delivery models as shown in Figure 1 .

A. Cloud Computing Infrastructure

cloud computing infrastructure consists, broadly [6], of the following:

- Servers – The physical machines that host one or more virtual machines.

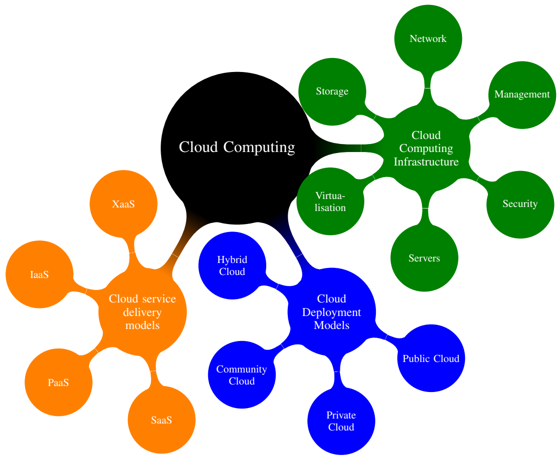


Figure 1. Cloud Computing Architecture

- Virtualization: technology that abstracts physical components such as storage, and networks and makes them available as software resources.
- Storage: in the form of SAN (Storage Area Networks), disk drives, etc. Along with archiving and backup.
- Network – To provide interconnections between physical servers.
- Management – Administration, management and monitoring of cloud infrastructure.
- Components that guarantee integrity, availability, and privacy.

B. *Cloud Deployment Models*

The cloud deployment model is based on the reasons for using cloud services and the environment in which they will be used. The costs spent, as well as the energy consumption of the resources, are determined by the deployment type chosen. [7]. Public cloud, private cloud, community cloud, and hybrid cloud are the most common deployment models in cloud infrastructures.

- Public cloud: The public cloud allows the general public to use business providers' systems and services. Because multi-tenancy is often utilized, it enables flexibility, scalability, and location independence at a very cheap cost [6]–[9].
- Private cloud: A private cloud is one that is utilized within a single organization, meaning that the cloud's resources and services may only be accessed or used by that company. This approach ensures that apps and data are kept secure and private [7]–[9].
- Community Cloud: This model is used by multiple companies/organizations at the same time and helps specific communities/societies that include public participation (for example, security requirements, compliance considerations, etc.). The model can be operated, owned, and managed by one or more organizations within the community or a third party. [9].
- Hybrid cloud refers to a combination of public and private clouds. Critical events (e.g., events that require secure operations) are performed using private cloud services, whereas non-critical events are done using public clouds in this sort of cloud deployment [7], [9].

When companies want to use collaboration services like chat and video conferencing but don't have the IT resources or infrastructure locally, public clouds are the best option. A private deployment strategy, on the other hand, should be utilized if stringent security and privacy are top priorities. A hybrid deployment strategy, on the other hand, should be chosen by a company with a vast IT infrastructure that is also increasing its capabilities.

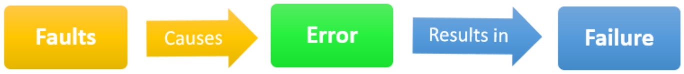
C. *Cloud Service Delivery Models*

The basic service models are described as follow:

- Software-as-a-Service (SaaS): In this model, cloud service providers provide software applications to consumers/end users in the form of services [2], [8]. The application delivered as customer service does not need to be installed and run on the user's computer. Web conferencing services, email apps, social networking platforms, and other similar services are examples. Amazon AWS, Google Compute Engine, Microsoft Azure, CloudStack, OpenStack, OpenNebula, CloudForge, Citrix, and others are among the SaaS providers.
- Platform-as-a-Service (Paas): This approach provides a cloud-based platform for developing, running, testing, and managing applications [2], [7], [10]. Users can rent an environment from Cloud Services Providers and use it for custom application development. Acquia Cloud, Amazon AWS, Apprenda, AppScale, Bluemix, Cloud 66, Cloudways, and others are among the PaaS providers.
- Infrastructure-as-a-Service (IaaS): The IaaS model provides the ability to access some major resources, namely physical machines, storage, networks, servers, virtual machines in the cloud, etc. [2], [4], [10]. IaaS providers propose services such as the dynamic configuration of virtual machines as well as on-demand storage. Salesforce, Microsoft, Amazon Web Services, Slack, Zendesk, GitHub, Oracle, Cisco, and others are among the SaaS suppliers.
- Anything-as-a-Service (XaaS): XaaS is a service model that can provide anything or everything. To manage a vast number of resources to fulfill individual, fine-grained, and particular demands, cloud systems can employ Security-as-a-Service, Identity-as-a-Service, and DaaS (Database-as-a-Service) [6].

III. Fault Tolerance Approaches

Fault tolerance is very important to the system, even in the presence of a component failure one or more [11], [12], the system can provide the necessary services. Failures in the system are the result of errors caused by faults (Figure III). The following is a list of them:



A. Faults

Because of an abnormal condition or bug in one or more elements of the system, the system is unable to fulfill its necessary/required functions [4], [13]. Various types of failures can appear in the system, and their classification is shown in Figure 2.

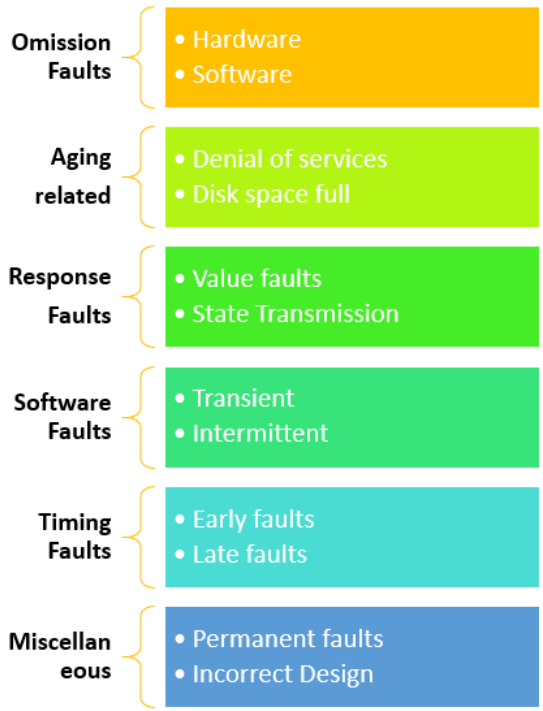


Figure 2. Classification of Faults

B. Error

Due to the existence of faults, errors or incorrect states can occur in system components. Incorrect operation of system components may cause some or all system failures [14], [15]. Distributed systems can contain many types of errors, as shown in Figure 3.

C. Failure

It refers to the inappropriate behavior of the system that can be observed by the user (human or other computer systems). When the system's output or outcome is wrong, it is called a failure [14], [15]. Faults can be classified as shown in Figure 4 .

D. Fault Tolerance Approaches

In cloud platforms, fault tolerance is important because it can ensure application performance, reliability, and availability. To achieve the robustness of cloud computing, it is necessary to effectively access and handles the failures [3], [4], [14], [16]. The following are some fault tolerance techniques identified in the literature:

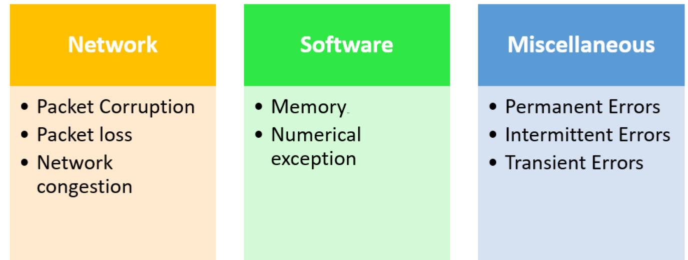


Figure 3. Error Classification

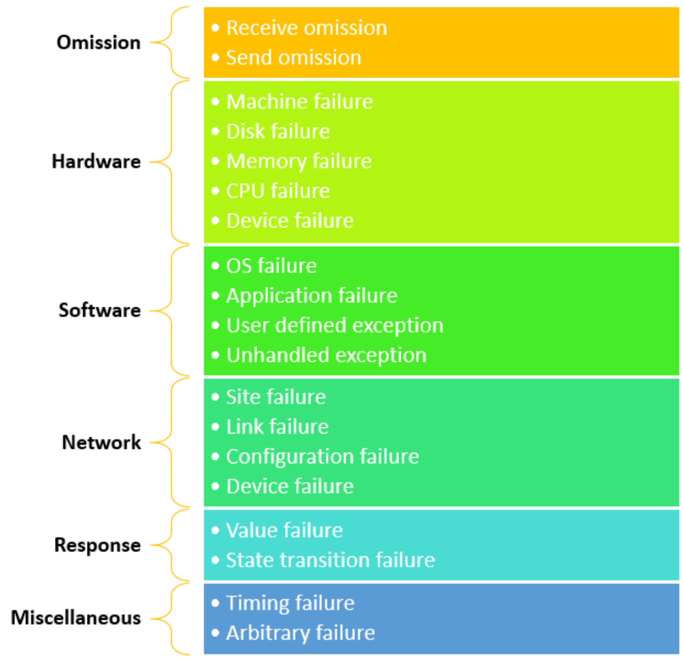


Figure 4. Failure Classification

1) Reactive fault tolerance: This approach is primarily used to reduce the impact of cloud system failures after the failure/faults occur. It ensures the system's robustness or dependability [4], [11]. For the cloud and other distributed systems, reactive fault tolerance techniques have been investigated. Some techniques are described below:

- **Check-pointing:** It is used to save the system state periodically. If the constitutive task fails, the job will be restarted from the last checked report status instead of starting from the beginning [17], [18].
- **Job Migration:** If a job cannot run on a particular physical machine for some reason and fails, it will be migrated to another machine [19].
- **Replication:** It's used to duplicate a job and store the duplicates in separate locations. In the case of a failure or malfunction, the task can continue to execute until all copies are destroyed [14], [18].
- **S-Guard:** It based on the recovery and rollback process

[20].

- **Retry:** In this technique, the job will repeatedly run until it succeeds. Retrying unsuccessful jobs is done with the same resource [19].
- **Task Resubmission:** In this technique, The unsuccessful task is resubmitted to the same resource and/or a new machine for execution [14], [17].
- **Rescue workflow:** Allows the system to continue to operate after a task/job fails until it cannot continue to operate without correcting the failure [19].

2) Proactive fault-tolerance:: This approach is used predict failures and replace suspect components with some running components. An overview of some proactive Fault tolerance techniques is described below.

- **Self-Healing:** This method uses a divide-and-conquer technique, in which an enormous task is broken down into multiple parts. The main purpose of partitioning is to increase system performance. The failure of an application instance will be handled automatically when several instances of the same application are operating on various VMs (virtual machines). It allows the computer system or device to identify itself and solve the dilemma/problem without relying on the administrator [4], [17], [21].
- **Software Rejuvenation:** In this method, the system will periodically reboot and start each time from a different state [14], [19].
- **Pre-emptive Migration:** In this method, The feedback loop control approach is used because the application is constantly monitored and analyzed [20], [22].
- **Load Balancing:** When a certain/maximum limit is surpassed, this approach is utilized to balance memory and CPU load. The extra CPU load will be distributed to other CPUs that haven't reached their maximum capacity.

IV. Related work

A. Proactive approaches

The following are some proposed models and works based on proactive fault tolerance:

- PeiYun Zhang., et al, 2018. [23], SVM-Grid, an online fault detection technique, was proposed. This method is said to be critical for cloud stability. Several defect detection models, according to the author, are necessary to comprehend the underlying structure of the cloud system. The conventional SVM (Support Vector Machine) is the most often used model, although its accuracy is poor. An online fault detection model based on SVM-Grid has been developed to tackle this problem. SVM-Grid forecasts new cloud issues. The grid technique is used to enhance the model input parameters, resulting in predictions that are modified for better/higher accuracy. In addition, in order to improve the fault prediction performance as well as lowering the time cost, a fine-tuned prediction algorithm and a Fault tolerance update algorithm have been developed for the sample DBs (databases). A simulation experiment was performed using a data set from Google (Google2 application cluster). The proposed method is also compared with existing methods, namely back propagation, LVQ (learning vector quantization), and traditional SVM. Experimental results show that compared with BP, LVQ, and traditional SVM, the newly developed model provides higher accuracy and lower time cost.
- Jialei Liu., et al, 2016. [24] proposed a PCFT (Proactive Co-ordinated Fault Tolerance) method that accepts the virtual machine (VM) harmonized/coordination method to find the deteriorating physical machine in the data center (DC), and then automatically migrates a virtual machine from a physical machine downgraded to some optimal target PM. There is also a proposal for an initial virtual cluster allocation mechanism. The suggested technique is divided into two parts: the first part proposes a framework for predicting the physical machine's depreciation based on CPU temperature. The second stage involves searching for the optimal target physical machine using a meta-heuristic method and a particle swarm optimization technique. The effectiveness of the suggested approach was also calculated using relevant indicators, and it was compared to five other methods: FF(First-fit), best-fit (BF), random first-fit (RFF), modified best fit decreasing (MBFD), and IVCA. The PCFT technique has less transmission overhead and a shorter overall executio time than the other five algorithms, according to the testing data, the PCFT technique has a lower transmission overhead and takes less time to complete.

B. Reactive approaches

The following are some proposed models and works based on reactive fault tolerance:

- In A. Zhou., et al, 2017. [25]The Edge Switch Failure Aware Checkpointing (EDCKP) concept was suggested. This approach seeks to increase cloud computing system service dependability. The fat tree network's topology has been examined, and two methods for detecting edge switch failure have been developed. The checkpoint image storage server is chosen using one technique, while the recovery server is chosen using the other. The suggested model was compared to current models such as NOCKP (No checkpoint-based method) and NDCKP (No checkpoint-based method) (which is a network topology aware distributed delta checkpoint-based technique). The EDCKP approach decreases overall execution time, uses less network resources, and provides greater service dependability, according to simulation results.
- In G. Yao et al, 2017. [26] The ICFWS (Fault Tolerance Workflow Scheduling) algorithm was proposed. The benefits of fault tolerance scheduling based on resubmission and replication techniques are provided by this approach. First, it recursively splits the whole soft deadline of the process into numerous sub-deadlines for all tasks. Then, depending on the results of the deadline division,

it chooses the appropriate fault-tolerant method for each task from resubmission and replication. Then, to change due dates for tasks that were not completed during task execution, an online appointment and modification technique was provided.

- In S. Wang., et al, 2016. [27] The OPVMP (Optimal Redundant Virtual Machine Placement) model was presented. Using the replication-based fault tolerance methodology, this method was created to increase the dependability of server-based cloud applications. The suggested technique consists of three steps: selecting the host server, determining the optimal virtual machine location, and determining the best recovery plan. Heuristic algorithms have been used to correctly select the best placement for host servers and virtual machines. Experiments to verify the advantages of this method have been performed on the CloudSim simulator [28]. The suggested method's findings were compared to five other existing models. The suggested technique consumes fewer network resources than previous methods, according to experimental results.
- Mohd. Amoon, 2016. [29] created an adaptive model/framework to deal with difficulties in cloud environments caused by faults For a highly dependable platform to handle client requests, the adaptive model includes replication and checkpoint fault tolerance mechanisms. The proposed framework includes two algorithms: one for selecting virtual machines to service client requests and another for deciding/selecting fault tolerance strategies (replication or checkpoint). The suggested framework's performance was assessed in terms of performance, overhead, availability, and monetary cost. The CloudSim tool was used to run the simulation. The suggested framework's performance was compared to that of the current OCI (optimal checkpoint algorithm). As a result, as compared to current methods, the adaptive features of the proposed framework improve cloud environment performance.
- J. Zhao et al, 2016. [30] To enable customizable reliability optimization in a cloud context, researchers suggested the JCSR (Joint Checkpoint Scheduling and Routing) approach. Based on the client's individual requirements and an evaluation of the data center's complete resources, the peer-to-peer checkpoint technique was utilized to maximize the client's consistency level/point. Through dual decomposition, a distributed method is constructed to tackle the joint optimization issue. However, a specific solution can boost resource usage and offer data center owners with a new revenue stream. The results demonstrate that, as compared to previous approaches, the dependability has improved substantially. The method was created with the goal of identifying specific users' optimization problems.
- W. Chen et al, 2015. [31] offered a framework for task failure modeling. A hypothetical examination of runtime performance is done in this framework to understand the impact of transitory failures in scientific procedures. Three fault-tolerant clustering strategies are proposed to improve the workflow's runtime performance: selective re-clustering algorithm (a new clustered job is used to keep the clustered job having only failed tasks), dynamic re-clustering algorithm (used to dynamically adjust the granularity or size of the cluster or a number of tasks inside a job), and vertical re-clustering algorithm (used to divide the clustered jobs into improved jobs that reduce the job granularity and then retry them). The simulation was carried out using the WorkflowSim tool.
- C. Chen et al, 2014. [32] presented the KNF (k-out-of-n reliability) framework for energy efficiency and fault tolerance. The proposed framework provides the convenience of data fragmentation when nodes store data. Other nodes can obtain data with constant nominal power consumption and allow mobile nodes to process distributed data. However, the energy consumption required to process data is also reduced. MATLAB was used to test the framework.
- W. Qiu et al, 2013. [33] developed the ROCloud (reliability-based design optimization) model/framework for improving application reliability using fault tolerance approaches. Legacy apps have been migrated to the cloud using the framework. Legacy application analysis, major component ranking, and automated optimum fault-tolerant strategy selection are the three phases of the framework. However, two techniques have been developed to arrange the components of conventional (all application components may be transferred to the cloud) and hybrid (all application components can be migrated to the cloud) systems (only some parts of components can be migrated to the cloud). The simulation was carried out using a tool written in the C++ programming language.

The main features of the discussed works have been summarized in Table I .

V. Conclusion

The primary motivation for using fault tolerance strategies in cloud computing is to improve availability, dependability, and failure recovery.

Cloud computing ideas, components, service models, and deployment methodologies have all been covered in this survey study. Various fault tolerance approaches, models, and algorithms to increase cloud service dependability have been listed. We have gathered several recent works based on fault tolerance approaches and summarized all of these works in a table, taking into account the limits of existing fault tolerance techniques and new technology in related fields.

References

[1] P. Mell, T. Grance *et al.*, "The nist definition of cloud computing," 2011.

[2] M. U. Bokhari, Q. M. Shallal, and Y. K. Tamandani, "Cloud computing service models: A comparative study," in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2016, pp. 890–895.

Table I
SUMMARY OF THE PROPOSED WORKS BASED ON PROACTIVE AND REACTIVE FAULT TOLERANCE TECHNIQUES

Reference	Technique	Proactive/Reactive	Reliability	Availability	Simulator
PeiYun Zhang., et al, 2018	Pre-emptive Migration	Proactive	X	X	Matlab
Jialei Liu., et al, 2016	Pre-emptive Migration	Proactive	X	X	Not menthoned
A. Zhou., et al, 2017	Check-pointing	Reactive	Yes	X	Ftcloudsim
G. Yao et al, 2017	Resubmission/Replication	Reactive	X	Yes	WorkflowSim toolkit
S. Wang., et al, 2016	Replication	Reactive	Yes	X	Ftcloudsim
Mohd. Amoon, 2016	Replication/Check-pointing	Reactive	Yes	Yes	CloudSim
J. Zhao et al, 2016	Check-pointing	Reactive	Yes	Yes	Ftcloudsim
W. Chen et al, 2015	Self-Healing/Retry	Proactive Reactive	Yes	X	WorkflowSim
C. Chen et al, 2014	Replication	Reactive	X	X	Matlab
W. Qiu et al, 2013	Job Migration/Retry/Task Resubmission	Reactive	Yes	X	C++ language

[3] M. K. Gokhroo, M. C. Govil, and E. S. Pilli, "Detecting and mitigating faults in cloud computing environment," in *2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT)*. IEEE, 2017, pp. 1–9.

[4] L. P. Saikia and Y. L. Devi, "Fault tolerance techniques and algorithms in cloud computing," *International Journal of Computer Science & Communication Networks*, vol. 4, no. 1, pp. 01–08, 2014.

[5] E. Dubrova *et al.*, "Fault tolerant design: An introduction," *Department of Microelectronics and Information Technology, Royal Institute of Technology, Stockholm, Sweden*, pp. 22–3, 2008.

[6] S. Singh, Y.-S. Jeong, and J. H. Park, "A survey on cloud computing security: Issues, threats, and solutions," *Journal of Network and Computer Applications*, vol. 75, pp. 200–222, 2016.

[7] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in *2009 Fifth International Joint Conference on INC, IMS and IDC*. Ieee, 2009, pp. 44–51.

[8] S. Patidar, D. Rane, and P. Jain, "A survey paper on cloud computing," in *2012 second international conference on advanced computing & communication technologies*. IEEE, 2012, pp. 394–398.

[9] L. Savu, "Cloud computing: Deployment models, delivery models, risks and research challenges," in *2011 International Conference on Computer and Management (CAMAN)*. IEEE, 2011, pp. 1–4.

[10] E. AbdElfattah, M. Elkawkagy, and A. El-Sisi, "A reactive fault tolerance approach for cloud computing," in *2017 13th International Computer Engineering Conference (ICENCO)*. IEEE, 2017, pp. 190–194.

[11] T. J. Charity and G. C. Hua, "Resource reliability using fault tolerance in cloud computing," in *2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*. IEEE, 2016, pp. 65–71.

[12] G. Vallee, K. Charoenpornwattana, C. Engelmann, A. Tikotekar, C. Leangsuksun, T. Naughton, and S. L. Scott, "A framework for proactive fault tolerance," in *2008 Third International Conference on Availability, Reliability and Security*. IEEE, 2008, pp. 659–664.

[13] Y. M. Essa, "A survey of cloud computing fault tolerance: techniques and implementation," *International Journal of Computer Applications*, vol. 138, no. 13, 2016.

[14] Z. Amin, H. Singh, and N. Sethi, "Review on fault tolerance techniques in cloud computing," *International Journal of Computer Applications*, vol. 116, no. 18, 2015.

[15] G. Singh and S. Kinger, "A survey on fault tolerance techniques and methods in cloud computing," *International Journal of Engineering Research and Technology*, vol. 2, no. 6, 2013.

[16] M. N. Cheraghlou, A. Khadem-Zadeh, and M. Haghparast, "A survey of fault tolerance architecture in cloud computing," *Journal of Network and Computer Applications*, vol. 61, pp. 81–92, 2016.

[17] S. M. Ataallah, S. M. Nassar, and E. E. Hemayed, "Fault tolerance in cloud computing-survey," in *2015 11th International computer engineering conference (ICENCO)*. IEEE, 2015, pp. 241–245.

[18] I. Mahallat, "Fault-tolerance techniques in cloud storage: a survey," *Int J Database Theory Appl*, vol. 8, no. 4, pp. 183–190, 2015.

[19] S. Prathiba and S. Sowvarnica, "Survey of failures and fault tolerance in cloud," in *2017 2nd International Conference on Computing and Communications Technologies (ICCCT)*. IEEE, 2017, pp. 169–172.

[20] A. Bala and I. Chana, "Fault tolerance-challenges, techniques and implementation in cloud computing," *International Journal of Computer Science Issues (IJCSI)*, vol. 9, no. 1, p. 288, 2012.

[21] J. Park, G. Yoo, and E. Lee, "Proactive self-healing system based on multi-agent technologies," in *Third ACIS Int'l Conference on Software Engineering Research, Management and Applications (SERA'05)*. IEEE, 2005, pp. 256–263.

[22] C. Engelmann, G. R. Vallee, T. Naughton, and S. L. Scott, "Proactive fault tolerance using preemptive migration," in *2009 17th Euromicro International Conference on Parallel, Distributed and Network-based Processing*. IEEE, 2009, pp. 252–257.

[23] P. Zhang, S. Shu, and M. Zhou, "An online fault detection model and strategies based on svm-grid in clouds," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 2, pp. 445–456, 2018.

[24] J. Liu, S. Wang, A. Zhou, S. A. Kumar, F. Yang, and R. Buyya, "Using proactive fault-tolerance approach to enhance cloud service reliability," *IEEE Transactions on Cloud Computing*, vol. 6, no. 4, pp. 1191–1202, 2016.

[25] A. Zhou, Q. Sun, and J. Li, "Enhancing reliability via checkpointing in cloud computing systems," *China Communications*, vol. 14, no. 7, pp. 1–10, 2017.

[26] G. Yao, Y. Ding, and K. Hao, "Using imbalance characteristic for fault-tolerant workflow scheduling in cloud systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 12, pp. 3671–3683, 2017.

[27] A. Zhou, S. Wang, B. Cheng, Z. Zheng, F. Yang, R. N. Chang, M. R. Lyu, and R. Buyya, "Cloud service reliability enhancement via virtual machine placement optimization," *IEEE Transactions on Services Computing*, vol. 10, no. 6, pp. 902–913, 2016.

[28] R. N. Calheiros, R. Ranjan, C. A. De Rose, and R. Buyya, "Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services," *arXiv preprint arXiv:0903.2525*, 2009.

[29] M. Amoon, "Adaptive framework for reliable cloud computing environment," *IEEE Access*, vol. 4, pp. 9469–9478, 2016.

[30] J. Zhao, Y. Xiang, T. Lan, H. H. Huang, and S. Subramaniam, "Elastic reliability optimization through peer-to-peer checkpointing in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 2, pp. 491–502, 2016.

[31] W. Chen, R. F. da Silva, E. Deelman, and T. Fahringer, "Dynamic and fault-tolerant clustering for scientific workflows," *IEEE Transactions on Cloud Computing*, vol. 4, no. 1, pp. 49–62, 2015.

[32] C.-A. Chen, M. Won, R. Stoleru, and G. G. Xie, "Energy-efficient fault-tolerant data storage and processing in mobile cloud," *IEEE Transactions on cloud computing*, vol. 3, no. 1, pp. 28–41, 2014.

[33] W. Qiu, Z. Zheng, X. Wang, X. Yang, and M. R. Lyu, "Reliability-based design optimization for cloud migration," *IEEE Transactions on Services Computing*, vol. 7, no. 2, pp. 223–236, 2013.