Fault Tolerance in Cloud Manufacturing: An Overview

Auday Al-Dulaimy[1,2](✉), Mohammad Ashjaei[1], Moris Behnam[1], Thomas Nolte[1], and Alessandro V. Papadopoulos[1]

[1] Mälardalen University, Västerås, Sweden
{auday.aldulaimy,mohammad.ashjaei,moris.behnam,thomas.nolte, alessandro.papadopoulos}@mdu.se
[2] Dalarna University, Falun, Sweden
aay@du.se

Abstract. Utilizing edge and cloud computing to empower the profitability of manufacturing is drastically increasing in modern industries. As a result of that, several challenges have raised over the years that essentially require urgent attention. Among these, coping with different faults in edge and cloud computing and recovering from permanent and temporary faults became prominent issues to be solved. In this paper, we focus on the challenges of applying fault tolerance techniques on edge and cloud computing in the context of manufacturing and we investigate the current state of the proposed approaches by categorizing them into several groups. Moreover, we identify critical gaps in the research domain as open research directions.

Keywords: Cloud computing · Edge computing · Cloud manufacturing · CMfg · Manufacturing as a Service · MaaS · Fault tolerance · Reliability

1 Introduction

To drive profitability, manufacturers adopted the concept of Cloud Manufacturing (CMfg) in their business, as it offers manufacturing services with lower cost and better performance. CMfg builds on top of cloud and edge computing, and it uses the infrastructure on cloud data centers (cloud layer) and on the factory-site computing servers (edge layer) to transform traditional manufacturing resources into services.

Cloud computing has huge computing and storage capabilities, however, its classical centralized architecture comes with some limitations [1]. These limitations include (1) A stable connectivity between factory sites and cloud data centers is required to offer convenient services, (2) Cloud computing assumes that

This research was partially sponsored by the Knowledge Foundation (KKS) under the SACSys project, and by the research center XPRES (Excellence in Production Research) - a strategic research area in Sweden.

there is enough bandwidth to transfer data between the manufacturers' devices and the cloud data centers, (3) Massive data transfer causes network bottlenecks leading to latency and performance deterioration issues, and (4) Data transfer results in security concerns. On the other hand, edge computing tries to overcome such limitations by providing decentralized and closer to factory-site computing and storage resources. However, such resources usually have limited capacities [1,8].

In CMfg, the above-mentioned limitations in cloud and/or edge layers may result in system failures. A system failure can be defined as [12]: *"An event in which the system fails to operate according to its specifications. A system failure occurs, when a system deviates from fulfilling its normal system function for which it was aimed at."* Besides the limitations accompanied by cloud and edge layers, the production phases that consist of dynamic and long life-cycle processes add more complexity to the systems [26], and such complexity comes with more difficulties in providing reliable and fault tolerances services in the manufacturing environments. Therefore, there is a need to manage the failures that may occur in the services offered to the manufacturing and industrial sectors. Without proper fault tolerance approaches, multiple manufacturing services will fail to lead to great losses. A better understanding of the topics related to fault tolerance in the context of manufacturing will help improve productivity and increase profitability. Therefore, the aim of this work is to cover the essential directions related to system failures in CMfg.

1.1 Contributions

The main contributions of this paper are summarized as follows:

– investigating the system failures in the context of the CMfg environment (specifically the edge and cloud layers), with a focus on failures at the edge layer, and presenting an overview of the issues related to system failures.
– presenting the research gaps that are associated with the existing fault tolerance approaches in the edge layer. Such gaps need to be investigated when designing new fault tolerance systems, taking the features of the edge devices into consideration, which are also listed in this paper.

1.2 Paper Layout

The rest of this paper is organized as follows: Sect. 2 presents the taxonomy of the paper. It presents a background on reliability and fault tolerance in cloud and edge computing. In addition, it gives a brief literature review of the existing fault tolerance approaches and categories them. Section 3 defines the research gaps in the topic. Finally, Sect. 4 concludes the paper.

2 Taxonomy

The paper presents examples of cloud manufacturing in practice, followed by a detailed discussion on the reliability and fault tolerance issues in the context of a cloud manufacturing environment. The detailed discussion covers the following perspectives:

– Cloud manufacturing in practice
– Features of the edge devices
– Reliability in edge-cloud environments
– Reasons for system failures in edge-cloud environments
– Fault tolerance solutions in edge-cloud environments

The work in this paper is focusing on the system failures in the *edge layer*. Several existing works (*e.g.*, [18,24], and [22]) survey fault tolerance and its related topics in cloud computing environments, not specifically looking at the edge. Finally, complementary to the work presented in this paper, the work in [3] compares and shows the differences/similarities between the edge and cloud layers.

2.1 Cloud Manufacturing in Practice

Recently, manufacturing started broaden its overall objective from production-oriented to include more cases of service-oriented manufacturing, and as a consequence the cloud service providers began to offer what is called Manufacturing as a Service (MaaS). With MaaS the manufacturers can outsource the services and new industrial technologies related to all process stages to a trusted party, while concentrating on the innovation and core mission. Among the main pioneers in providing MaaS, we can mention the following:

Google Cloud for Manufacturing. Google cloud aims at helping manufacturers to transform into a digital environment by providing innovative solutions that reshape the production and factory-floor operations [10].

Amazon Cloud for Manufacturing. Amazon provides advanced digital transformation solutions to manufacturers. Such solutions utilize machine learning and data analysis to optimize production and improve operational efficiencies [4].

Microsoft Cloud for Manufacturing. Microsoft offers manufacturing services that drive productivity and improve security. The core processes and requirements of the industry are encapsulated and provided as capabilities from Microsoft aiming at providing secure connection and resilient business processes. Such capabilities enhance the time-to-value metrics for manufacturers in a scalable fashion. Microsoft cloud and edge resources are integrated with smart components for providing different manufacturing scenarios such that the beneficiaries select the highest-value scenario [16].

However, to provide stable manufacturing services, the providers need to offer and maintain fault tolerant systems at the edge and cloud layers.

2.2 Features of the Edge Devices

The infrastructure at the edge computing layer has specific characteristics. In particular, the edge devices are featured by the following:

– Constrained devices, *i.e.*, they have limited compute power and fixed storage capacity.
– Geo-distributed devices.
– Heterogeneous devices.
– Connectivity cannot be guaranteed with the cloud layer.
– Edge devices run containers more efficiently compared with Virtual Machines (VMs).

The devices at the edge of the network need to perform communication and computation tasks in order to provide real-time responses for a large number of end devices at the Manufacturing layer. They are connected horizontally with each other, and vertically across layers (either upwards with the cloud layer or downwards with the manufacturing layer) [7].

The aforementioned characteristics and design goals must be considered when proposing fault tolerance approaches in the edge layer environment.

2.3 Reliability in Edge-Cloud Environment

Reliability in service-oriented edge-cloud computing, which is adopted and used by manufacturers, is how consistently the services are provided without interruption and/or failure. A failure is a state when any system fails to operate according to its design goals, or when the system can not work according to a specific predefined Quality-of-Service (QoS). Fault tolerance is a way to prevent or deal with failures, such that the system continues operating and providing services regardless of the failure type. In CMfg, fault tolerance approaches are essential to meet the manufacturers' requirements, and to understand the infrastructure needed to provide persistent manufacturing services.

Designing a robust fault tolerant system in CMfg requires a deep understanding of the reasons and types of failures, and how the systems should respond to such failures. This will be discussed in the following sections.

2.4 Reasons for System Failures

Understanding the causes of the occurrence of the failure is fundamental in proposing the appropriate solution to avoid or deal with failures. The reasons for failures in a manufacturing environment, which comprises the edge and cloud layers, can be categorized into five main categories, as shown in Fig. 1.

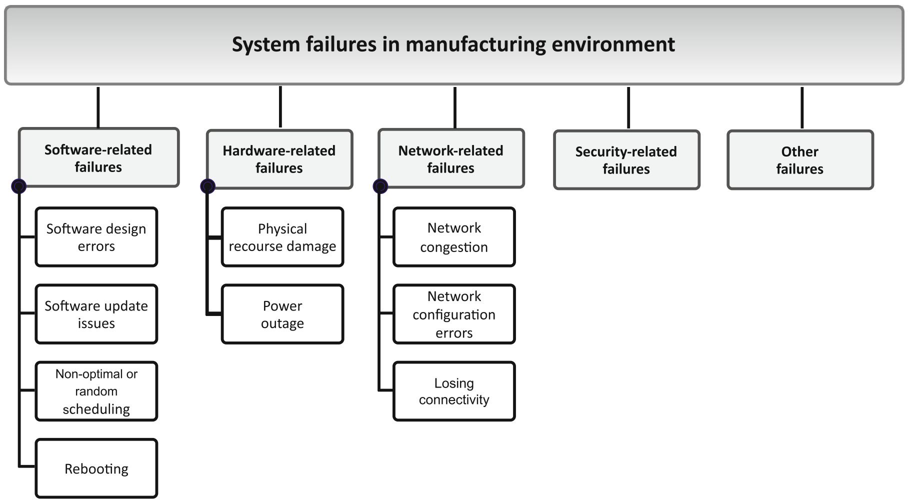
System failures in manufacturing environment
Software-related failures
Software design errors
Software update issues
Non-optimal or random scheduling
Rebooting
Hardware-related failures
Physical recourse damage
Power outage
Network-related failures
Network congestion
Network configuration errors
Losing connectivity
Security-related failures
Other failures

Fig. 1. The reasons of system failures in manufacturing environments.

Software-Related Failures: This category includes all failures resulting from the running software systems and applications. Recently, the software is getting complex and accompanied by diverse functional and non-functional requirements. Moreover, the software is more sophisticated as they are designed to work in an edge-cloud environment. Therefore, the software became a significant reason for system breakdown which causes loss in business and revenue. Failures may occur due to the following situations in this category:

- Software design errors: Designing software in the right way is essential in avoiding many errors which may lead to system faults [28]. The common fault examples include implementing incorrect infinite loops, numerical overflow/underflow, and no protection against deadlocks.
- Software update issues: As a result of security issues, or to enrich the applications with additional features, the software is regularly updated with new batches. This can make the software volatile to faults.
- Non-optimal or Random scheduling: Usually, the designed software has to be executed together with other applications with either data or time dependencies. This normally requires proper scheduling for the software to avoid any blocking or dependency issues among the software applications. A naive scheduling method can increase the chance of faults during the run-time of the software.
- Rebooting: The software can be rebooted, either planned or unplanned, during the execution of the system, which can potentially stop serving the running applications, and bring new faults that were not covered before.

Hardware-Related Failures: This category includes all failures resulting from hardware resource failures or replacements. In addition, power failures can fit in this category. The main cases within this category are as follows:

- Physical recourse damage on a server, a CPU, memory, a disk, and network links.
- Planned or unplanned power outage.

Network-Related Failures: In edge-cloud computing, the services are provided by communication networks that connect the edge and cloud layers, and also connect the nodes within the same layer [2]. Any outage of the system network leads to a service outage. For IIoT applications, especially real-time applications that require meeting deadlines, network performance is essential in providing stable services. Any network delay may lead to a service failure. The network service failures could be divided into the following:

- Network congestion: It is a state when a link or any network device in the system is forwarding a huge amount of data which can result in over-consumption of the communication bandwidth. This can violate the QoS requirements but is also considered a fault in the network.
- Network configuration errors: It covers the processes of assigning network settings, policies, controls, and data flows [30]. In the edge-cloud environment, the design and infrastructure of the networking are virtualized and then implemented by underlying software across physical network devices. The proper network configuration is essential in supporting the network flow and stability, otherwise, it may lead to network failures.
- Losing connectivity: Availability of the network is a metric that is also dictated by a Service Level Agreement (SLA) and violating that can affect the overall system performance.

Security-Related Failures: Several security issues lead to failures, such as viruses and malicious. In an edge-cloud environment, the system must be able to defend against malicious attacks and provide a trusted storage and computing base, otherwise, the security attacks may lead to a system failure.

Other Faults: In this category, we can include many examples that are not categorized in the previous groups, such as human errors or natural disasters (*e.g.*, earthquakes).

2.5 Dealing with Failures

The approaches to deal with failures in the manufacturing environment can be classified into four different categories, as depicted in Fig. 2. In the following, these categories are elaborated with more details.

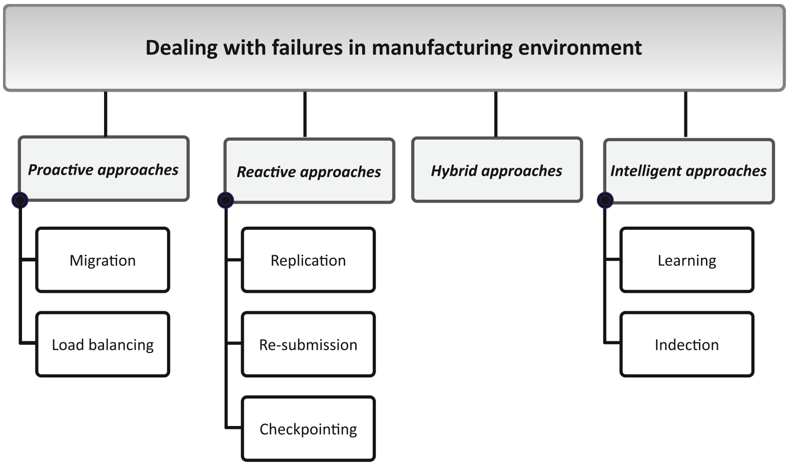

Fig. 2. Dealing with failures.

Proactive: To maintain reliability, some cloud service providers adopt proactive approaches to avoid possible failures before their occurrences. This way is used to predict the faults proactively and substitute the suspected component with some running components. The proactive approaches mainly can be sub-classified into two classes [27]: migration and load balancing.

Several existing robust proactive approaches are presented in the literature. For example: In [20], the authors proposed an approach that aims at preventing system faults within the federated cloud environments. The environment is modeled as a multi-objective optimization problem that maximizes the profit and minimizes the VMs migration cost. The approach redistributes VMs from the expected faulty providers (based on the CPU temperature) to healthy ones within the federation. The work in [29] proposes an approach to predict preemptive migration decisions using a Generative Adversarial Network (GAN) in a containerized edge environment. The proposed model, called *PreGAN*, detects and classifies faults to schedule migrations to obviate the potential system faults. In [27], the authors proposed an approach, called Automated Pipeline for Advanced Fault Tolerance (APAFT), to monitor the task production rates at the edge layer and check the ability of the edge computational nodes to serve these tasks. APAFT predicts the potential bottlenecks in task execution that may result in potential system faults, and accordingly triggers proactive node replication. In [17], the authors compare different proactive approaches based on control theory and probabilistic model checking for the autoscaling of cloud-based systems.

Reactive: This way is mainly used to decrease the influence of failure and provide reliability to the system after the failures have occurred. Reactive approaches take some measures in order to react accordingly. In general, such approaches may result in a large overhead and expensive implementation, but cloud service providers need to utilize reactive ways to manage any potential failures. The reactive approaches mainly can be sub-classified into three classes [27]: replication, re-submission, and checkpointing.

Several existing robust reactive approaches are presented in the literature. For example: The authors in [33] presented a redundant VM placement optimization approach to secure a fault-tolerant cloud system. The optimization function considers the huge network resource consumption issues related to failure recovery mode. Three different algorithms were employed in the process of VM (re)placement to provide reliable cloud services. In [31], the authors presented a two-stage fault tolerance approach (offline and online) to improve the reliability of the manufacturing network. The off-line stage ranks the manufacturing services according to their importance in fault tolerance, then the critical services are replicated. While the online stage performs a heuristic algorithm for replacing the failed services. A two-stage unsupervised fault recognition approach, called Deep Adaptive Fuzzy Clustering (DAFC), is presented in [11]. DAFC integrates two clustering algorithms, Stacked Sparse AutoEncoder (SSAE) and Adaptive Weighted Gath-Geva (AWGG), aiming at proposing an unsupervised fault recognition framework to cluster unlabeled big data in the manufacturing environment and then extract fault features from the clusters. In [15,19], the authors analyze the usage of control-based reactive load balancing techniques for masking potential faults in the data center.

Hybrid: To maintain the maximum possible level of reliability and availability in cloud manufacturing, several approaches consider both proactive and reactive ways to deal with system failure.

In the literature, very few hybrid approaches have been proposed. For example, in [5], the authors presented a hybrid model to take fault tolerance actions, including proactive actions after predicting the failure probability, and reactive actions that employ replication and checkpointing techniques. The work in [24] presented a fault-tolerance approach that utilizes two directions. The first direction is to perform VM migrations based on a failure prediction technique, while the second direction is to implement the checkpointing process. Moreover, the work in [1] presented an approach, called TOLERANCER, that aims to solve the software and hardware-related failures in a cloud manufacturing environment. TOLERANCER composes of connected components that are collaborating with each other to detect stress situations or node failures, and accordingly, trigger actions to avoid and solve potential system failures. The work presented in [25] describes a hybrid checkpointing mechanism implemented in OpenStack, that can be used for optimizing the usage of resources based on the incoming workload while improving the fault-tolerance capacity of the virtualized environment.

Intelligent: Such approaches try to handle application requirements when faults happen and, at the same time, improve the service within an appropriate time frame. Intelligent systems are resilient and include smart elements that are able to deal with the application requirements during any disruptions aiming at reaching a system's safe status. Such approaches share common features with the proactive approaches, such as monitoring the system and predicting faults to avoid them, but they differ from proactive approaches in utilizing intelligent learning [18]. The Intelligent approaches can be sub-classified into two classes [18,22]: learning and induction.

Selected intelligent approaches can be mentioned: In [6] the authors stated that fault diagnosis is essential in offering stable services in industry. Aiming at identifying and preventing system failures, they used simple vibration data and applied different unsupervised learning algorithms that test the performance and robustness of the system. The work in [32] utilized deep learning to propose a fault identification approach in industrial systems. To diagnose faults, the approach extracts features from the system, and then a classification model is used to detect fault information.

In [23] the authors stated that most of the existing load balancing algorithms in cloud environments ignore fault tolerance in their design. Thus, they presented an algorithm that employed fault tolerance metrics in a load balancing approach. The approach works in three phases: *Observing phase* to collect system information aiming at identifying any disorganized behaviors, *Inspection phase* to specify the relation or differentiate between defects aiming at false diagnosis, and *Organize and Implementation phase* to reassign a correct weight to the damaged elements aiming at storing healthy system state.

3 Findings and Research Gaps

Studying the fault tolerance approaches presented in Sect. 2.5, and exploring the state of the art, *e.g.*, [13,14], led us to define the following issues to be tackled when designing new fault tolerance approaches in the edge-cloud layers.

- In the context of cloud manufacturing, manufacturers are utilizing both the edge and the cloud layers. Hence, the designed fault tolerance approach must be holistic in considering different aspects, e.g., long-distance network connectivity (vertical network) between different layers, network connectivity (horizontal network) within the same layer, security and privacy issues, and all related failures discussed in Sect. 2.4.
- Edge and cloud resources have different features, as discussed in Sect. 2.2. Thus, the designed fault tolerance approaches need to have separate implementations for the edge and the cloud layers.
- IIoT applications are commonly real-time applications that have certain timing requirements. Therefore, the decisions of the designed fault tolerance approaches need to be made at run-time to avoid failures.
- The cloud manufacturing environment is dynamic and serves different IIoT applications which are accompanied by diverse functional and non-functional requirements. Thus, fault tolerance approaches in such environments need to be smart and able to learn and adapt to the system environment. The use of learning and induction algorithms seems promising in providing solid fault-tolerant systems for manufacturers.

- Hybrid approaches combine the advantages of both proactive and reactive approaches, hence it is always better to be adopted compared with considering proactive or reactive approaches individually.
- There are many technical issues in the existing fault tolerance approaches to tackle in order to meet their design goals. Among many, we can identify the following technical issues:
    1. Checkpointing [9] in the reactive approaches performs efficiently at the edge layer as the resulting checkpoint files have small sizes which can be stored on the edge nodes of fixed storage capacity. However, it is possible to further speed up the checkpointing process, which results in faster service retrieval and continuing the ongoing application(s).
    2. Function as a Service (FaaS) [21] is a promising platform under the umbrella of cloud computing as it allows developers to run their applications without considering the complexities related to building or maintaining the infrastructure. In cloud manufacturing, manufacturers could get benefit from FaaS if it is used at the edge layers. However, there are crucial drawbacks with FaaS at the edge layer which need to be solved. For instance, hosting long-running function instances on constrained edge devices may not be feasible due to the memory requirements of Dockers which deliver software in packages called containers. In addition, the computation cannot be paused and continued later (stateless). Therefore, novel methods to manage function containers are needed to overcome these drawbacks.
    3. Theoretically, integrating FaaS with checkpointing looks promising in providing solid fault-tolerant systems. However, some challenges may appear with this combination. For instance, how to specify the sleep and active timing for the function containers? Proposing visible solutions to such challenges is needed, for example, injecting the system through an external monitoring process to examine file descriptors and incoming network connections.

4 Conclusion and Future Directives

Cloud Manufacturing (CMfg) utilizes the resources at the edge and cloud layers, aiming at providing holistic manufacturing services to maximize manufacturers' profits. Such services should be stable and fault tolerant. Many fault tolerance approaches have been proposed in the literature, however, there are challenges that must be addressed and solved in order to attain reliable and fault tolerant systems in the context of CMfg.

This work presented an overview of the fault tolerance-related issues in CMfg environments, along with the research gaps in such environments. As future directives, we are working on: (1) expanding this work to include more topics related to system failures in CMfg, and (2) solving some of the identified technical issues using hybrid and/or smart approaches.

References

1. Al-Dulaimy, A., Christian, S., Papadopoulos, A.V., Galletta, A., Villari, M., Ashjaei, M.: Tolerancer: a fault tolerance approach for cloud manufacturing environments. In: IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA 2022) (2022)
2. Al-Dulaimy, A., Itani, W., Taheri, J., Shamseddine, M.: bwslicer: a bandwidth slicing framework for cloud data centers. Futur. Gener. Comput. Syst. **112**, 767–784 (2020)
3. Al-Dulaimy, A., Sharma, Y., Khan, M.G., Taheri, J.: Introduction to edge computing. Edge Comput. Models Technol. Appl. 3–25 (2020)
4. Amazon: manufacturing: simplifying digital transformation. https://aws.amazon.com/manufacturing/ (2022). Accessed 2022
5. Amoon, M.: A framework for providing a hybrid fault tolerance in cloud computing. In: 2015 Science and Information Conference (SAI), pp. 844–849. IEEE (2015)
6. Amruthnath, N., Gupta, T.: A research study on unsupervised machine learning algorithms for early fault detection in predictive maintenance. In: 2018 5th International Conference on Industrial Engineering and Applications (ICIEA), pp. 355–361. IEEE (2018)
7. Bakhshi, Z., Rodriguez-Navas, G., Hansson, H.: Dependable fog computing: a systematic literature review. In: 2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), pp. 395–403. IEEE (2019)
8. Du, W., et al.: Fault-tolerating edge computing with server redundancy based on a variant of group degree centrality. In: Kafeza, E., Benatallah, B., Martinelli, F., Hacid, H., Bouguettaya, A., Motahari, H. (eds.) ICSOC 2020. LNCS, vol. 12571, pp. 198–214. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-65310-1_16
9. Egwutuoha, I.P., Levy, D., Selic, B., Chen, S.: A survey of fault tolerance mechanisms and checkpoint/restart implementations for high performance computing systems. J. Supercomput. **65**(3), 1302–1326 (2013). https://doi.org/10.1007/s11227-013-0884-0
10. Google: Google cloud for manufacturing. https://cloud.google.com/solutions/manufacturing/ (2022). Accessed 2022
11. Hu, X., Li, Y., Jia, L., Qiu, M.: A novel two-stage unsupervised fault recognition framework combining feature extraction and fuzzy clustering for collaborative AIoT. IEEE Trans. Industr. Inf. **18**(2), 1291–1300 (2021)
12. Javadi, B., Thulasiraman, P., Buyya, R.: Enhancing performance of failure-prone clusters by adaptive provisioning of cloud resources. J. Supercomput. **63**(2), 467–489 (2013)
13. Javed, A., Heljanko, K., Buda, A., Främling, K.: Cefiot: a fault-tolerant IoT architecture for edge and cloud. In: 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), pp. 813–818. IEEE (2018)
14. Karhula, P., Janak, J., Schulzrinne, H.: Checkpointing and migration of IoT edge functions. In: Proceedings of the 2nd International Workshop on Edge Systems, Analytics and Networking, pp. 60–65 (2019)
15. Klein, C., et al.: Improving cloud service resilience using brownout-aware load-balancing. In: IEEE 33rd International Symposium on Reliable Distributed Systems (SRDS), pp. 31–40. IEEE, New York (2014). https://doi.org/10.1109/SRDS.2014.14

16. Microsoft: Introducing microsoft cloud for manufacturing. https://www.vmware.com/topics/glossary/content/network-configuration.html (2022). Accessed 2022
17. Moreno, G.A., Papadopoulos, A.V., Angelopoulos, K., Cámara, J., Schmerl, B.: Comparing model-based predictive approaches to self-adaptation: Cobra and PLA. In: 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), pp. 42–53 (2017). https://doi.org/10.1109/SEAMS.2017.2
18. Mukwevho, M.A., Celik, T.: Toward a smart cloud: a review of fault-tolerance methods in cloud systems. IEEE Trans. Serv. Comput. **14**(2), 589–605 (2018)
19. Papadopoulos, A.V., et al.: Control-based load-balancing techniques: analysis and performance evaluation via a randomized optimization approach. Control. Eng. Pract. **52**, 24–34 (2016). https://doi.org/10.1016/j.conengprac.2016.03.020
20. Ray, B., Saha, A., Khatua, S., Roy, S.: Proactive fault-tolerance technique to enhance reliability of cloud service in cloud federation environment. IEEE Trans. Cloud Comput. **10**(2), 957–971 (2020)
21. Scheuner, J., Leitner, P.: Function-as-a-service performance evaluation: a multivocal literature review. J. Syst. Softw. **170**, 110708 (2020)
22. Shahid, M.A., Islam, N., Alam, M.M., Mazliham, M., Musa, S.: Towards resilient method: an exhaustive survey of fault tolerance methods in the cloud computing environment. Comput. Sci. Rev. **40**, 100398 (2021)
23. Shahid, M.A., Islam, N., Alam, M.M., Su'ud, M.M., Musa, S.: A comprehensive study of load balancing approaches in the cloud computing environment and a novel fault tolerance approach. IEEE Access **8**, 130500–130526 (2020)
24. Sharma, Y., Si, W., Sun, D., Javadi, B.: Failure-aware energy-efficient VM consolidation in cloud computing systems. Futur. Gener. Comput. Syst. **94**, 620–633 (2019)
25. Souza, A., Papadopoulos, A.V., Tomás Bolivar, L., Gilbert, D., Tordsson, J.: Hybrid adaptive checkpointing for virtual machine fault tolerance. In: IEEE International Conference on Cloud Engineering (IC2E), pp. 12–22 (2018). https://doi.org/10.1109/IC2E.2018.00023
26. Tao, F., Zhang, L., Liu, Y., Cheng, Y., Wang, L., Xu, X.: Manufacturing service management in cloud manufacturing: overview and future research directions. J. Manufact. Sci. Eng. **137**(4) (2015)
27. Theodoropoulos, T., Makris, A., Violos, J., Tserpes, K.: An automated pipeline for advanced fault tolerance in edge computing infrastructures. In: Proceedings of the 2nd Workshop on Flexible Resource and Application Management on the Edge, pp. 19–24 (2022)
28. Thieme, C.A., Mosleh, A., Utne, I.B., Hegde, J.: Incorporating software failure in risk analysis-part 1: software functional failure mode classification. Reliab. Eng. Syst. Saf. **197**, 106803 (2020)
29. Tuli, S., Casale, G., Jennings, N.R.: Pregan: preemptive migration prediction network for proactive fault-tolerant edge computing. In: IEEE INFOCOM 2022-IEEE Conference on Computer Communications, pp. 670–679. IEEE (2022)
30. vmWARE: What is network configuration. https://www.microsoft.com/en-us/industry/manufacturing/microsoft-cloud-for-manufacturing (2022). Accessed 2022
31. Wu, Y., Peng, G., Wang, H., Zhang, H.: A two-stage fault tolerance method for large-scale manufacturing network. IEEE Access **7**, 81574–81592 (2019)

32. Xing, D., Chen, R., Qi, L., Zhao, J., Wang, Y.: Multi-source fault identification based on combined deep learning. In: MATEC Web of Conferences, vol. 309, p. 03037. EDP Sciences (2020)
33. Zhou, A., et al.: Cloud service reliability enhancement via virtual machine placement optimization. IEEE Trans. Serv. Comput. **10**(6), 902–913 (2016)