Availability Modeling for Blockchain Provisioning in Private Clouds

Jamilson Dantas[‡], Priscila Silva[†], Lance Fiondella[†], Carlos Melo[*], and Paulo Maciel[‡]
[‡]Centro de Informática (CIN), Universidade Federal de Pernambuco, Recife, Brazil
[†]Electrical and Computer Engineering, University of Massachusetts Dartmouth, MA, USA
[*]Centro de Ciências da Natureza (CCN), Universidade Federal do Piauí, Picos, Brazil
carlos.alexandre@edu.ufpi.br[*], {psilva4, lfiondella}@umaasd.edu[†], {prmm, jrd}@cin.ufpe.br[‡]

***Abstract*—Blockchain technology has emerged, and many previous studies have assessed its performance issues. However, less attention has been paid to the dependability attributes, which have been a critical topic in service provisioning, considering public or private infrastructures. This paper introduces analytical models to assess the availability of private blockchain infrastructure for Hyperledger Fabric-based applications. Furthermore, a case study will be presented to demonstrate the feasibility of the proposed model, which may assist stakeholders in deciding whether to migrate from old to new technology. Some of the obtained results indicate that, unlike most conventional systems, general availability may decrease as new nodes are added to the environment. This phenomenon occurs due to the adopted endorsement policy, which determines the proportion of required nodes to sign the authenticity of a transaction.**

***Index Terms*—Availability, Blockchain, Hyperledger Fabric.**

I. Introduction

Blockchain is a decentralized technology that emerged in the late 2000s and is responsible for recording transactions across multiple computers securely and inflexibly, facilitating trust and transparency [1]. Despite its foundational role in cryptocurrencies, blockchain remains relatively less known to users than the digital assets it supports, such as Bitcoin and non-fungible tokens (NFT) [1], which may be attributed to the challenge of assigning value to an entire technology rather than to individual applications [2]. However, particularly in permissioned networks that mostly focus on industrial applications, we need to explore its broader feasibility for adoption by organizations that do not share mutual trust. Permissioned networks provide enhanced security and control over data sharing and transactions in these environments; one of the most popular platforms for this mean is the Hyperledger Fabric (HLF) [3].

Previous studies [4]–[6] have examined various aspects of Hyperledger Fabric, primarily focusing on platform performance. However, some research efforts [7], have concentrated solely on system availability concerns, needing a comprehensive evaluation of the overall environment.

This paper's main contributions are as follows:

- An overview of the Hyperledger Fabric environment and how it can be modeled;
- Development of generalized models for assessing the availability of Hyperledger Fabric environments;
- Presentation of a case study illustrating the practical application of the proposed models;
- Conducting simulations using the Mercury Modeling Tool to validate the effectiveness and feasibility of the proposed models.

The remainder of this paper is organized as follows. Section II presents the works that underlie this research. Section III provides an overview of the Hyperledger Fabric platform, its functioning, and how it can be modeled. Section IV presents a high-level overview of an application running on a Hyperledger Fabric environment. Section V introduces the proposed models and their evaluation. Section VI presents the scenarios used as a case study to demonstrate the feasibility of the proposed models. Finally, Section VII presents the final remarks and future directions.

II. Related Works

The previous research on the Hyperledger Fabric platform has predominantly focused on performance metrics, leaving gaps in the exploration of dependability attributes. This paper addresses these gaps and provides an updated understanding of the platform's capabilities.

In [8], we introduced models for evaluating computational resource usage in Hyperledger Fabric (HLF) environments. Our study, which Leveraged Continuous Time Markov Chains (CTMCs) and stochastic Petri nets (SPNs), demonstrated the effectiveness of these formalisms in modeling HLF-based applications and detecting infrastructure bottlenecks. However, our current model focuses specifically on general availability and system uptime.

Other studies, such as [9], employed a hierarchical modeling approach to analyze performance metrics like throughput, latency, and system utilization. In contrast, [10], [11] developed a queue theory-based model focusing on HLF's transaction flow considering various service rates and their impact on the general performance.

Additionally, [12] employed Generalized Stochastic Petri Nets (GSPN) to model the platform and investigate the influence of arrival rates and block sizes on HLF throughput and latency. While Sukhwani et al. [13] used Stochastic Reward Networks (SRN). However, both models are isomorphic to the CTMCs presented in this paper, with a different focus.

It is worth noting that in both [7], [14], we evaluated availability and costs related to deploying blockchain applications in cloud computing environments. These works serve as the foundation for our current models, which form the basis of the framework proposed in this paper. Furthermore, our analysis extends beyond previous evaluations by considering endorsement policies, providing a comprehensive resource for blockchain service provisioning decisions.

TABLE I: Summary of Differences and Contributions

Paper	**Main Differences and Contributions**
[8]	Introduced models for evaluating computational resource usage in HLF environments using CTMCs and SPNs. Focused on general availability and system uptime.
[9]	Employed a hierarchical modeling approach to analyze performance metrics such as throughput, latency, and system utilization.
[10], [11]	Developed a queue theory-based model focusing on HLF's transaction flow and considering various service rates' impact on general performance.
[12]	Used GSPNs to model the platform and investigated the influence of arrival rates and block sizes on HLF throughput and latency.
[13]	Used SRNs to model the platform's performance.
[7], [14]	Evaluated availability and costs related to deploying blockchain applications in cloud computing environments. Considered endorsement policies, providing a comprehensive resource for blockchain service provisioning decisions.
This paper	Evaluates availability of blockchain applications in private environments with a generalizable model.

III. A Hyperledger Fabric Overview

Hyperledger Fabric (HLF) is a platform for building and deploying solutions based on shared ledgers. Developed and maintained as an open-source standard by the Hyperledger Consortium under the auspices of the Linux Foundation, HLF offers a framework for creating distributed ledger applications. Various approaches exist for deploying a private or permissioned Hyperledger Fabric infrastructure. In a typical HLF environment, there are at least two key actors: the client and the service provider. The service provider often comprises a consortium of organizations that may not inherently trust each other.

This paper focuses solely on evaluating the service provider side, with changes on the client side having no bearing on the overall system availability. On the service provider side, HLF environments leverage container technology and are centered around managing smart contracts, known as chaincodes. These chaincodes must be pre-installed during environment deployment and are responsible for enforcing business rules and orchestrating application activities. Conversely, the client side utilizes an SDK, typically implemented in Node.js or Java, to facilitate communication between the server and client components.

Figure 1 provides a high-level depiction of a server hosting the minimal deployment of Hyperledger Fabric. As each component's failure leads to the service's failure, there are no explicit dependencies between them regarding service provisioning.

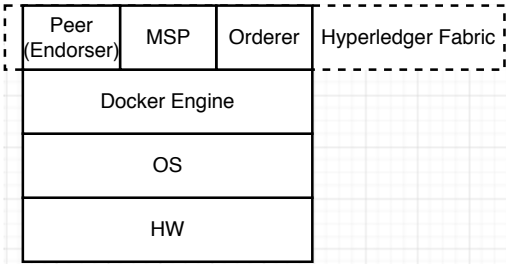


Fig. 1: Service Stack

The high-level view encompasses four distinct components: the server's hardware (HW), the operating system (OS), the container engine (Docker Engine), and the deployed containers. Within the Hyperledger Fabric environment, three types of containers are prominent:

- Peer nodes, which execute endorsement tasks;
- MSP (Membership Service Provider) nodes manage membership and platform access;
- Orderer nodes receive transactions, group them into batches (blocks), and distribute them back to the peer nodes to persist it (commit).

This architectural overview forms the foundation for this paper's models based on the relationships among these components.

IV. A Blockchain-based Application

This paper evaluates a basic application deployed over the three Hyperledger Fabric's containers. Usually, a client uses its SDK to send a transaction to the service provider. However, many other steps are required and must be first accomplished on the other side in order for a transaction to be performed.

Figure 2 shows how the client and the service provider communicates, as well as which steps must be followed by a transaction in order for it to be fully accepted by the system [15], [16].

The client connects to the system through the Membership Service Provider (MSP), which should provide no Single Point of Failure (SPoF) and no Single Point of Truth (SPoT). The MSP verifies the credentials of this client and allows access to the service provider. Later, the client proposes a transaction to the node known as Peer. There are many kinds of peers. We focus on endorsement peers, which simulate a transaction and send it back to the client with a signature that determines if the system can perform this transaction. As an example of a transaction, we may cite transferring assets between two clients, which requires both clients to exist and that the sender

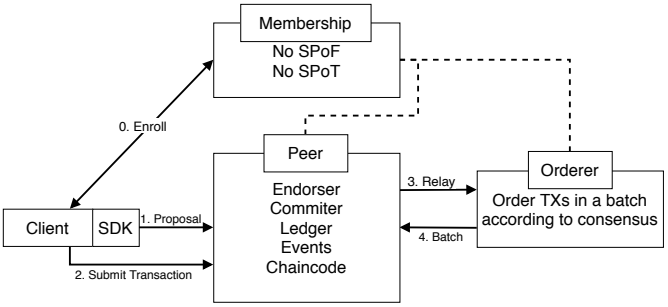


Fig. 2: Hyperledger Fabric's Overview

has as much balance as the value that he wants to transfer, discounting additional fees.

The transaction endorsement is based on a set of policies described by the chaincode. These policies specify how many and how these nodes agree with a transaction state. There are three main endorsement policies: AND, OR, and K-out-of-N (KooN). Suppose you have three servers on the service provisioning side, each hosting an endorsement peer. If an AND policy is used, all three nodes must endorse the transaction and sign it back to the client. If you use an OR policy, at least one of the three nodes must sign the transaction. The same applies to the KooN policy, where you determine how many available peers must sign the transaction, 1-out-of-3, 2-out-of-3, or even 3-out-of-3. After the endorsement, the transaction is submitted to the Orderer container. The orderer container joins all transactions in a batch (block) and sends it to the peers. Peers are responsible for executing transactions on the ledger following the requirements outlined in the chaincode, a fundamental aspect of Hyperledger Fabric applications.

V. Availability Models

This section presents the proposed availability models that may represent environments that can host Hyperledger Fabric's nodes and blockchain-based applications.

We considered a two-stage hierarchical modeling to represent a Hyperledger Fabric-based architecture and the fact that all components must be operational to perform service provisioning. We adopted a Reliability Block Diagram (RBD) model in the first stage, representing the primary system's components. The RBD depicts the primary server components (hardware, operating system, and container engine (Docker Engine)).

The system's components are presented in a serial RBD, meaning that if at least one of its components fails, the whole system will also fail. After evaluating the primary component's RBD, we obtained their respective Mean Time to Failure (MTTF) and Mean Time to Repair (MTTR). The system's next modeling step deals with Hyperledger Fabric containers. Figure 3 presents the Continuous Time Markov Chain (CTMC) model, which should conduct the second hierarchical modeling stage.

The CTMC first level deals with the previously stated RBD, which may have up to M machines, each with its Hardware, Operating System, and Docker Engine. It is important to highlight that each machine in this model runs three containers

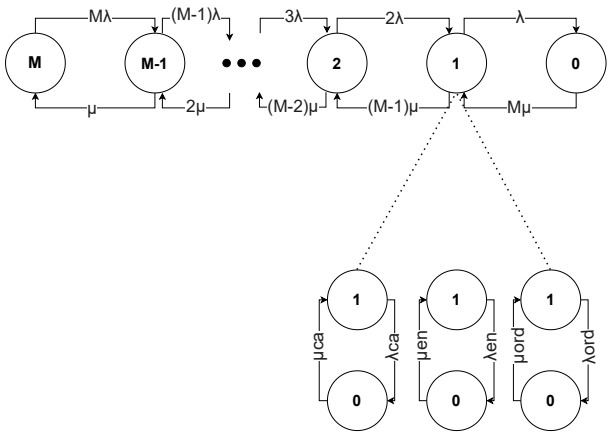


Fig. 3: Availability CTMC Model

(Endorser, MSP, and Orderer), represented by the CTMC's second level. The machines may enter a failure state or be repaired following an exponential distribution-based rate λ and μ, respectively. These rates are the inverse of the MTTF and MTTR values obtained from the RBD. The Hyperledger Fabric's containers already have their rates, which are the λ_{ca} and μ_{ca} that stands for the MSP fail. Repair rates, λ_{en} and μ_{en} for the endorser container, and λ_{ord} and μ_{ord} represents the orderer rates.

At this point, we could formulate a set of equations to evaluate the system's availability based on the aimed endorsement policy by using the State Diagrams package in Wolfram Mathematica [17] and concepts presented by [2]. Equation 1 depicts a model with only one physical machine and its three containers, which enable availability (A) computations of a single server. Later, we generalized this equation using a binomial distribution.

Equation 2 denotes the K-out-of-N endorsement policy, where someone can establish a K number of an M total of components that must be operational to accomplish service provisioning, which means the system is available and the endorsement is performed. This expression is a generalization of the previous one and can be used to calculate any number of servers and their associated container.

$$A_{\text{Server}} = \left(\frac{\mu}{\mu+\lambda}\right) \times \left(\frac{\mu_{ca}}{\mu_{ca}+\lambda_{ca}}\right) \times \left(\frac{\mu_{en}}{\mu_{en}+\lambda_{en}}\right) \times \times \left(\frac{\mu_{ord}}{\mu_{ord}+\lambda_{ord}}\right) \tag{1}$$

$$A_{\text{KooN}} = \sum_{i=k}^{M} \binom{M}{k} A_{\text{Server}}^{k} (1 - A_{\text{Server}})^{M-k} \tag{2}$$

where K stands for the number of components expected to be operational, and M is the total of resources that we have. Some specific scenarios may be extracted from the KooN policy, meaning that some other expressions can be obtained to calculate a combination of K and N values. The third Expression 3 represents the AND endorsement policy, which requires that all components in the first and second levels of

the CTMC be operational, which means that it is an N-out-of-N.

$$A_{\text{NooN}} = \left(\frac{\mu}{\mu+\lambda}\right)^{M} \times \left(\frac{\mu_{ca}}{\mu_{ca}+\lambda_{ca}}\right)^{M} \times \left(\frac{\mu_{en}}{\mu_{en}+\lambda_{en}}\right)^{M} \times \times \left(\frac{\mu_{ord}}{\mu_{ord}+\lambda_{ord}}\right)^{M} \tag{3}$$

VI. Results and discussion

This section provides a case study that demonstrates the proposed model's feasibility. We are considering an availability evaluation scenario.

The initial action involves gathering the necessary input values from the system to input into the models and conducting the availability assessment. These values comprise a mix of sources: some were sourced from relevant literature [18]–[20], while others were drawn from manufacturer charts and white papers. Table II presents the input values utilized in this study. Table III shows ten scenarios employed for illustration, where each considers a possible combination of endorsement policy and several nodes varying from one up to four servers hosting the three containers.

TABLE II: Input Parameters for Availability Evaluation

Component	MTTF (h)	MTTR (h)
Hardware (HW)	8760	1.66
Operating System (OS)	2893	0.15
Docker Engine (DE)	2516	0.15
Containers	1258	0.15

TABLE III: Evaluated Scenarios

Scenario	Policy	Required Servers	Total Servers
1	AND	1	1
2	AND	2	2
3	AND	3	3
4	AND	4	4
5	OR	1	2
6	OR	1	3
7	OR	1	4
8	KooN	2	3
9	KooN	2	4
10	KooN	3	4

After enumerating input availability values and exploring ten distinct scenarios following OR, AND, or K-out-of-N endorsement policy, we assessed their respective availabilities using Equation 2. Table IV showcases the resultant general availability for each scenario proposed.

The availability of an AND policy decreases with the addition of more resources (nodes), as expected, due to the necessity for all components to remain operational. Failure of any component prevents transaction endorsement. Conversely, an OR endorsement policy experiences increased availability with the addition of more resources. The KooN endorsement policy demonstrates intermediate availability compared to AND and OR policies. Its availability diminishes with the requirement for more nodes, resembling an AND policy. This outcome aligns with the Annual Downtime analysis for each scenario and endorsement policy.

TABLE IV: Availability Results

Scenario	Av. (%)	Av. (#9s)	A. Downtime (h)
1	99.9341	3.18	5.77
2	99.8683	2.88	11.53
3	99.8026	2.70	17.29
4	99.7369	2.58	23.05
5	99.9998	5.70	0.0038
6	99.9999	9.45	0.000003
7	99.9999	12.66	0.00000002
8	99.9987	4.89	0.011
9	99.9999	8.90	0.000009
10	99.9997	5.52	0.022

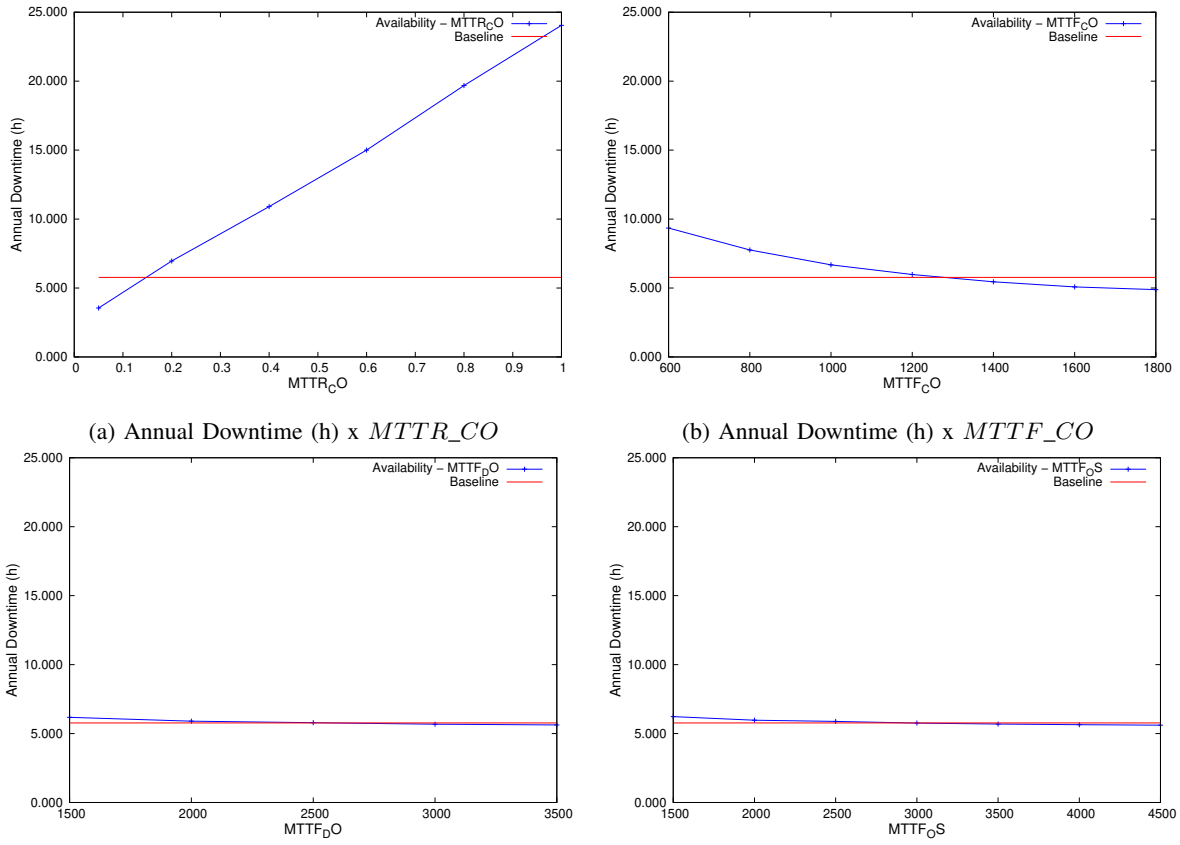
We have experimented to understand the impact of the input parameters on the overall system's annual downtime. To accomplish this task, we varied and rounded each parameter value in +50% and -50% as in the given time interval.

The bottleneck findings are illustrated in Figure 4, where we highlight the parameters exerting the greatest and least influence on system availability. We can see from this figure that the higher the MTTF, the lower the annual downtime, while the lower the MTTR, the higher the obtained downtime values, which should be expected behavior. Among these results, we may highlight the Container's MTTR and MTTF (4a and 4b subfigures). Those parameters are the bottleneck regarding the system's availability-associated metrics. On the other hand, the Docker MTTF and the Operating System MTTF do not mean to impact the interesting metric (see Figures 4c and 4d).

VII. Conclusions and Future Works

This paper presented a set of hierarchical models to evaluate the availability of the Hyperledger Fabric platform within private cloud computing infrastructures. The results may help stakeholders and decision-makers assess the impact of varying the parameters on a generalizable model representing the HLF blockchain environment to determine the effects on general availability and associated annual downtime. We presented a case study that dealt with the availability evaluation of a set of scenarios varying the number of nodes responsible for endorsement, an important characteristic of a Hyperledger Fabric infrastructure.

The main limitation of the current work is the need for an experimental evaluation to validate the proposed model in a real environment. This means that the obtained results are simulated based on the service provisioning stack and related work's provided data. Continuously, we intend to evaluate and compare different scenarios and applications, such as Industry 4.0, disaster risk management, and security systems.



(a) Annual Downtime (h) x $MTTR\_CO$

(b) Annual Downtime (h) x $MTTF\_CO$

(c) Annual Downtime (h) x $MTTF\_DO$

(d) Annual Downtime (h) x $MTTF\_OS$

Fig. 4: Sensitivity analysis for Availability

REFERENCES

[1] M. V. Kumar, N. C. S. N. Iyengar, and V. Goar, "Employing blockchain in rice supply chain management," in *Advances in Information Communication Technology and Computing*. Springer, pp. 451–461.

[2] P. R. M. Maciel, *Performance, reliability, and availability evaluation of computational systems, volume I: performance and background*. Chapman and Hall/CRC, 2023.

[3] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*. ACM, 2018, p. 30.

[4] H. Sukhwani, J. M. Martínez, X. Chang, K. S. Trivedi, and A. Rindos, "Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric)," in *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2017, pp. 253–255.

[5] S. e. a. Pongnumkul, "Performance analysis of private blockchain platforms in varying workloads," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2017, pp. 1–6.

[6] P. Thakkar, S. Nathan, and B. Viswanathan, "Performance benchmarking and optimizing hyperledger fabric blockchain platform," in *2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. IEEE, 2018, pp. 264–276.

[7] C. Melo, J. Dantas, R. Maciel, P. Pereira, E. Quesado, and P. Maciel, "Blockchain provisioning over private cloud computing environments: Availability modeling and cost requirements," in *2019 IEEE 8th International Conference on Cloud Networking (CloudNet)*. IEEE, 2019, pp. 1–3.

[8] C. Melo, J. Araujo, J. Dantas, P. Pereira, and P. Maciel, "A model-based approach for planning blockchain service provisioning," *Computing*, vol. 104, no. 2, pp. 315–337, 2022.

[9] L. Jiang, X. Chang, Y. Liu, J. Mišić, and V. B. Mišić, "Performance analysis of hyperledger fabric platform: A hierarchical model approach," *Peer-to-Peer Networking and Applications*, vol. 13, pp. 1014–1025, 2020.

[10] O. Wu, S. Li, L. Liu, H. Zhang, X. Zhou, and Q. Lu, "Performance modeling of hyperledger fabric 2.0," in *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering 2022*, 2022, pp. 357–365.

[11] Z. Ke and N. Park, "Performance modeling and analysis of hyperledger fabric," *Cluster Computing*, pp. 1–19, 2022.

[12] P. Yuan, K. Zheng, X. Xiong, K. Zhang, and L. Lei, "Performance modeling and analysis of a hyperledger-based system using gspn," *Computer Communications*, vol. 153, pp. 117–124, 2020.

[13] H. Sukhwani, N. Wang, K. S. Trivedi, and A. Rindos, "Performance modeling of hyperledger fabric (permissioned blockchain network)," in *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2018, pp. 1–8.

[14] C. Melo, J. Dantas, R. Maciel, P. Silva, and P. Maciel, "Models to evaluate service provisioning over cloud computing environments-a blockchain-as-a-service case study," *Revista de Informática Teórica e Aplicada*, vol. 26, no. 3, pp. 65–74, 2019.

[15] Hyperledger, "An introduction to hyperledger," Tech. Rep., 2018.

[16] A. Hyperledger, "Introduction to hyperledger business blockchain design philosophy and consensus," Tech. Rep., 2018.

[17] E. W. Weisstein *et al.*, "Mathworld–a wolfram web resource," 2004.

[18] J. Dantas, "Modelos para analise de dependabilidade de arquiteturas de computação em nuvem," Master's thesis, Centro de Informática - Universidade Federal de Pernambuco (Recife, Brasil), 2013.

[19] C. Melo, J. Dantas, J. Araujo, and P. Maciel, "Availability models for synchronization server infrastructure," in *Proceedings of the IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC'16)*, Budapest, Hungary, 2016.

[20] S. Sebastio, R. Ghosh, and T. Mukherjee, "An availability analysis approach for deployment configurations of containers," *IEEE Transactions on Services Computing*, 2018.