Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)
Vol. 8, No. 2, June 2022, pp. 282-289
ISSN: 2338-3070, DOI: 10.26555/jiteki.v8i2.23629

Towards Service Level Agreement Quantification on Service-Based Computing

Irving Vitra Paputungan, Akmal K Denna, Devi Rachmawati
Universitas Islam Indonesia, Sleman, Yogyakarta 55584, Indonesia

ARTICLE INFO

Article history:

Received March 20, 2022
Revised June 30, 2022
Accepted August 17, 2022

Keywords:

Service Level Agreement;
Quantification;
Weightage;
Violation

ABSTRACT

A service Level Agreement is an agreement between service providers and consumers that contains the rights and obligations of both parties, particularly in terms of the delivery of services provided during the subscription period on service-based computing. Once approved, normally, the Service Level Agreement will not change until the end of the subscription period. SLA violations are often positioned between yes and no. As a result, service providers must deal with severe penalties or compensation. In this paper, the use of weightage for each SLA parameter is introduced in this paper. Such quantification using weightage is the main contribution. SLA violation detection cases in service-based computing are used to demonstrate how SLA quantification works. In the simulation scenario of SLA quantification, the presence of weightage and its aggregates along with the upper and lower bound is able to help the SLA violation detection process more appropriate. Violations are no longer seen between Yes and No, but the severity of the violation can also be determined. The number of violated parameters is not very influential in determining the level because the main determinant is the weightage. At the same time, the upper and lower limits are also very helpful in determining the level of violation. It is believed that SLA quantification is the way forward for better SLA management.

Corresponding Author:

Irving V Paputungan, Universitas Islam Indonesia, Sleman, Yogyakarta 55584, Indonesia
Email: irving@uii.ac.id

1. INTRODUCTION

A service level agreement (SLA) is a contract between a service provider and a consumer that specifies both parties' rights and obligations, especially in terms of the delivery of services provided during the subscription period [1][2][3][4]. SLA is normally applied to service-based and business-related computing. An example of a common SLA is “availability = 95%,” which means that within 365 days (1 year), only 5% of service interruptions will occur. SLA also contains penalty issues for service providers in the event of a violation or disruption [5][6][7][8]. With the SLA, which is usually agreed upon by both parties at the beginning of the subscription period, the level of consumer confidence or trust in the service will be better [9][10].

Once approved, normally, the SLA will not change until the end of the subscription period [11]. This is in contrast to the situations in which the service is delivered via the Internet, where performance is highly volatile and influenced by a variety of factors [12]. This means that interference and violations of the SLA are very likely to occur. In fact, violations are often positioned between yes and no. It makes service providers have to deal with penalties or compensation [13]. If there are too frequent interruptions or violations, the level of consumer confidence or trust will decrease in the service provider [14][15].

There are several forms of compensation or penalties that are usually stated in the SLA, for example, “If there is a disruption in the provision of services, the consumer will be compensated according to a certain calculation.” For the availability case, an example of the calculation is “amount of compensation = (interruption time x number of consumers affected) / promised service time.” Compensation can also be financial or other

additional services. On the other hand, the number of parameters in the SLA is normally more than one. Thus there is a certain complexity in determining the overall level of violation of a service that has several SLA parameters.

This paper presents a brief review of how best to quantify each SLA parameter along with the sample of its application. The motivation, related works, and contributions are presented in Section 1. Section 2 contains the method of how the SLA is quantified. An example of applying SLA quantification is briefly presented in section 4. Section 5 concludes this paper with a brief analysis and conclusion.

SLA violations often occur due to highly dynamic conditions and loads in service-based computing environments [16][17]. It is also sometimes caused by a lack of resources or hardware and software failures. The negative impact of breaching the SLA is a loss of trust in services, as well as financial damage to customers [18][19]. Violation of the Service Level Agreement is a major challenge in the research community of SaaS cloud providers [21].

As previously stated, normally, violations are only divided into 2 (two), namely 'violated' or 'normal.' However, given the nature of service-based computing, this classification becomes less relevant. There are 3 (three) categories of service fulfillment in the contract law in Europe, i.e. [22]: a) All-or-nothing provisioning, that is, service fulfillment will only be considered successful if all SLA parameters are met; b) Partial provisioning, which means that as long as some of the SLA parameters have been met, the service is considered successful; and c) Weighted Partial provisioning, which means that the success of service is measured by SLA parameters with a certain weightage that has exceeded the threshold.

The 3rd category (Weighted Partial provisioning) was selected and used as a basic reference in SLA quantification based on SLA@SOI (European Project on SLA Management) report. It says that changes in the quality of Internet or network performance can have an impact on the likelihood of SLA violations occurring [23]. If an unavoidable violation occurs, then a penalty needs to be given based on the severity of the violation. Furthermore, the 3rd category was also chosen to overcome differences in performance measurement of several SLA parameters that exist in service [24].

SLA parameters in service-based computing are divided into 2 (two), namely functional and non-functional parameters, see Fig. 1. Functional parameters define how services are maintained and how services are delivered to consumers. Data security and privacy matters fall into this category. Meanwhile, parameters such as availability and throughput are included in the non-functional category. Non-functional ones consist of parameters that are easy to measure. It is also called Service Level Objective (SLO).

Non-functional parameters play a very important role in service-based computing because these parameters will always be monitored and evaluated regularly to ensure the performance of the services provided [25]. Each service will have different parameters, depending on the application or data stored [26]. Such measurement will be a consumer expectation of the service providers. In many services, SLO is often shown in the form of Quality of Service (QoS) [27], or the results of calculations from QoS, such as Availability, are calculated based on uptime and downtime. Some QoS parameters are shown in Table 1.

Fig. 1. SLA Taxonomy

Table 1. Some QoS parameters

Parameter	Definition
Uptime	The time when the system is on and running normally
Downtime	The time when the system crashes
Execution Time	The time it takes for the service provider to turn on the system
Latency	The time it takes for a client request to reach the service provider
Response Time	The time it takes the system to react to the input
Round trip Time	Total time required for a request to return to the client
Throughput	Actual data rate in network

Service-based computing that is subscribed to by consumers often has more than 1 (one) non-functional parameter. Such a condition is inappropriate to determine a violation using only one of those parameters. Because it is possible that only a small number of parameters are disturbed when at the same time other parameters are running normally. In this case, the concept of a utility function that uses weightage [28] can be used in service-based computing.

From the economic point of view of a service, utility is a measure of the preference for a number of goods (or services that people normally prefer). Utilities represent the satisfaction experienced by consumers with goods [29]. If such representation is adopted in service-based computing, both the service provider and the consumer can be assumed to have a preference [30] in the form of service objective, where normally, the service provider will try their best to meet the objectives of the consumer. This means that the preference of one party can be mapped on a utility value [15][31][32], and the higher the utility value, the higher the preference value.

According to [33], the utility cannot be measured or observed directly. Economists try a way of inferring the value of utility by dividing it into cardinal and ordinal utilities. For cardinal utilities, the level of difference in utility values is determined ethically or locally. For example, beverage A has a utility value of u(A) = 120, B has a utility value of u(B) = 80, and D has a utility value of u(D) = 40. From a cardinal utility point of view, it can be concluded that A is better than B by the same amount and B is better than D. Whereas in ordinal utility, value differences are not determined ethically or culturally but based on priorities. Therefore in the previous beverage example, it can be concluded that A will be prioritized over B and D.

In service-based computing, consumers are bound by the ability to pay based on their preferred service performance preferences. The consumers will determine preferences based on the priorities needed. Thus, ordinal utility is feasible to be adopted in SLA quantification. Such quantification is the main contribution of this paper. Another contribution is the implementation within a service-based computing scenario.

2. METHOD

In this section, design SLA quantification in the case of SLA violation detection is presented. Several previous studies regarding the detection of SLA violations have been carried out. For example, the development of the DeSVi architecture that functions as a monitoring tool and SLA violation detector [34]. The detection is based on a predefined performance threshold. An SLA violation detection framework is also described in [35] and [36]. However, those proposed detectors use the All-or-Nothing provisioning approach, where the violation will be determined from the fulfillment of only based on one of the SLA parameters.

In order to detect SLA violations with the concept of Weighted Partial Provisioning, service consumers must define an initial weightage W for each parameter they used, and the total weightage is $\sum W = 1$. It reflects the adoption of ordinal utility. Each W is in the range from 0 to 1 ($0 < W \leq 1$).

After determining the weightage, it should also be noted that SLA violations in service-based computing should not only be between Yes and No decisions but also in between. This means that it is necessary to propose and determine the limit of the violation T on the total weightage when the violation occurred. In this case, consumers must determine not only the upper limit but also the lower limit in order to distinguish the level of violations that occurred. Violation levels are commonly used in some network monitoring tools, as in Cacti [37], Nagios [38], and PRTG [39]. Violations on such tools are normally divided into 4 (four) levels: No Violation, Low-Level Violation, Moderate Level Violation, and High-Level Violation. Such categorization of violations can be merely adopted in the violation detection of service-based computing cases [40][41].

The SLA violation detection process is depicted in Fig. 2 as a diagram. The weightage of each selected SLA preference and the threshold for determining the level of violation will be determined by consumers. Each parameter's weightage will be aggregated and compared to the upper and lower limits.

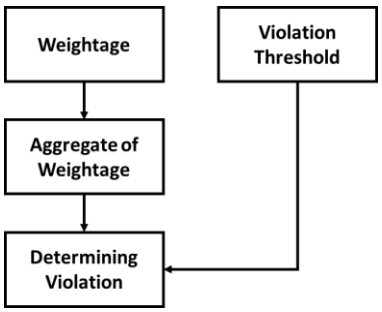

Fig. 2. Violation Detection Method

If the violated weightage is $W_i^{violated}$ and those violated parameter is multiplied by a constant p, the weightage aggregation is:

$$W^{violated} = \sum p_i \times W_i^{violated} \quad (1)$$

such that

$$p_i \begin{cases} 1, if\ it\ is\ violated \\ 0, if\ not\ violated \end{cases} \quad (2)$$

If T_1 is the lower bound and T_2 is the upper bound, then the categorization of violation can be seen in Table 2. Violations are divided into 3 (three) levels, namely low, medium, and high levels. The division is based on the aggregate weightage of the parameters that occur during the violation.

Table 2. SLA Violation Level

Level	Category	Condition
1	Low	$0 < W^{violated} \leq T_1$
2	Moderate	$T_1 < W^{violated} \leq T_2$
3	High	$T_2 < W^{violated} \leq 1$

3. RESULTS AND DISCUSSION

Assume there is a service A with SLA: Availability 98% per day, Response Time 8 ms per day, and Daily Throughput 50% of the maximum bandwidth. The service is more concerned with fast Response time than Availability and Throughput. If the service bandwidth is 10 Mbps, then the minimum data speed (throughput) is 5 Mbps. With different priorities for each parameter, it is decided that the weight for Availability is 0.3, Response Time is 0.4, and Throughput is 0.3. If you look at the data in Fig. 3, Fig. 4, and Fig. 5, there have been several violations of all parameters, but not at the same time. Table 3 shows the level of violations during the 5 days of the service with $T_1 = 0.3$ and $T_2 = 0.8$. On day one there were no violations. In the next day there was a violation on Response Time, the aggregate weightage of the violation was 0.4, where the weightage is categorized as Moderate Violation. The same level of violations also occurred on the 4th day, where violations occurred in Availability and Throughput so that the aggregate weightage of the violations was 0.6. Meanwhile, on the 3rd and 5th days, there was a low-level violation where the aggregate weight was only 0.3. Violations on the 3rd day occur in Availability, and on the 5th day occur in Throughput. The data used in this preliminary experiment are artificial data.

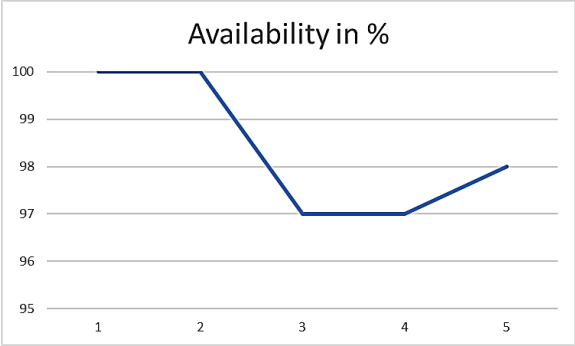

Fig. 3. Availability Data for 5 Days

Table 3. Violation using $T_1 = 0.3$ and $T_2 = 0.8$

Parameters	Day 1	Day 2	Day 3	Day 4	Day 5
Availability	0	0	0.3	0.3	0
Response Time	0	0.4	0	0	0
Throughput	0	0	0	0.3	0.3
Violation Level	n/a	2	1	2	1

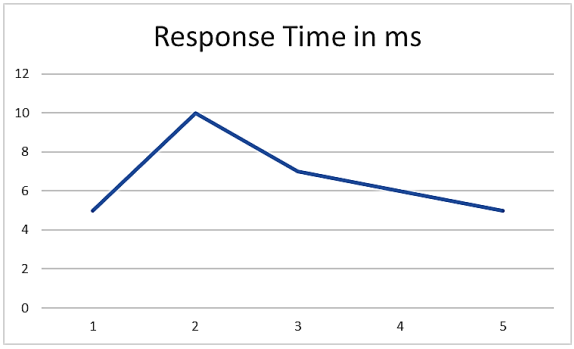

Fig. 4. Response Time Data for 5 Days

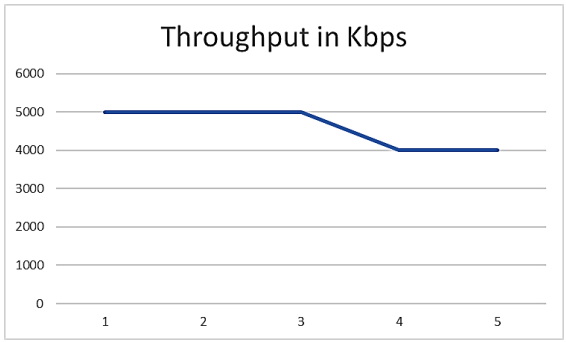

Fig. 5. Throughput Data for 5 Days

Another example is a service B with an SLA containing Availability, Response Time, Throughput, Execution Time, and Packet Loss. The agreement for the first 3 (three) parameters is assumed to be the same as service A, while the Execution Time is 2 ms, and Packet Loss is 3%. It can be seen in Fig. 6 and Fig. 7 that a violation occurred on the 3rd day for Execution Time, and only on that day was there no Packet Loss violation.

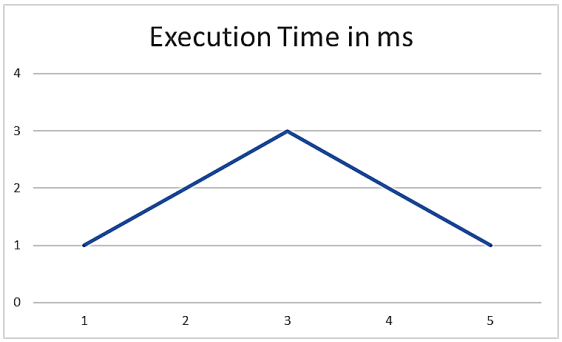

Fig. 6. Execution Time Data for 5 Days

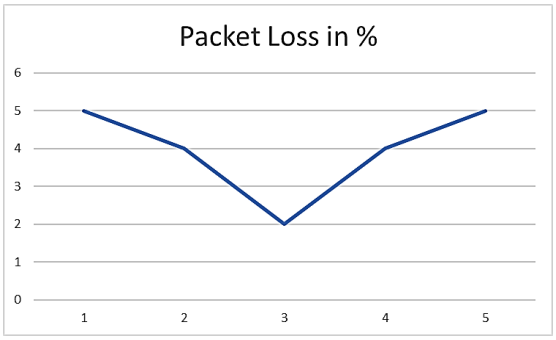

Fig. 7. Packet Loss Data for 5 Days

If the weightage for each parameter is set as 0.2, while $T_1 = 0.3$ and $T_2 = 0.8$, the level of violations is now as seen in Table 4. On the first day, there was a low level of violation because the aggregate $W^{violated}$ was only 0.2. In contrast to days 2 to 5, there has been a moderate level of violation because $W^{violated}$ is between T_1 and T_2. Whereas on the 4th and 5th days, there were violations on 3 (three) parameters, compared to the previous 2 (two) days.

Table 4. Violation using $T_1 = 0.3$ and $T_2 = 0.8$

Parameter	Day 1	Day 2	Day 3	Day 4	Day 5
Availability	0	0	0.2	0.2	0
Response Time	0	0.2	0	0	0
Throughput	0	0	0	0.2	0.2
Execution Time	0	0	0.2	0	0
Packet Loss	0.2	0.2	0	0.2	0.2
Violation Level	1	2	2	2	2

4. CONCLUSION

SLA quantification on service-based computing has been successfully carried out to address differences in performance measurement or SLA parameters on a service. Using weightage per parameter determined by the consumer at the beginning of the subscription period can help the process of handling disruptions to the service.

In sample scenarios 1 and 2 of the application of SLA quantification, it can be seen that the presence of weightage and its aggregates can solve the problem of detection of violations. Violations are no longer seen between Yes and No, but the severity of the violation can also be determined. The number of violated parameters is not very influential in determining the level because the main determinant is the weightage. At the same time, the upper and lower limits are also very helpful in determining the level of violation.

There are still many implementation scenarios that can be done from this quantification process, such as prediction [42][43] or classification of violations. Even further testing needs to be done from this quantification so that it can actually be applied to real-world cases and obtain a deeper analysis. It is believed that SLA quantification is the way forward for better SLA management.

REFERENCES

[1] Y. Ruan and A. Durresi, "A Trust Management Framework for Cloud Computing Platforms," *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, 2017, pp. 1146-1153, https://doi.org/10.1109/AINA.2017.108.

[2] A. Omezzine, N. B. B. Saoud, S. Tazi, and G. Cooperman, "Adaptive and concurrent negotiation for an efficient cloud provisioning," *International Journal of High Performance Computing and Networking*, vol. 15, no. 3-4, pp. 145-157, 2020, https://doi.org/10.1504/IJHPCN.2019.106088.

[3] N. R. Tadapaneni, "Cloud Computing Security Challenges," *International Journal of Innovations in Engineering Research and Technology*, vol. 7, no. 6, 2020.

[4] H. Lyu, Y. Xiao, R. Yan, Y. Jin, R. Shen and B. Sheng, "High Availability Evaluation Utilizing Service Level Agreement," *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2017, pp. 841-844, https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData.2017.129.

[5] J. Comden, S. Yao, N. Chen, H. Xing, and Z. Liu, "Online Optimization in Cloud Resource Provisioning: Predictions, Regrets, and Algorithms," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 1, 2019, https://doi.org/10.1145/3322205.3311087.

[6] S. Kumar and N. Kumar, "Conceptual Service Level Agreement Mechanism to Minimize the SLA Violation with SLA Negotiation Process in Cloud Computing Environment," in *Baghdad Science Journal*, vol. 18, no. 2, 2021, https://doi.org/10.21123/bsj.2021.18.2(Suppl.).1020.

[7] H. El-Kassabi, M. A. Serhani, R. Dssouli, N. Al-Qirim and I. Taleb, "Cloud Workflow Resource Shortage Prediction and Fulfillment Using Multiple Adaptation Strategies," *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, 2018, pp. 974-977, https://doi.org/10.1109/CLOUD.2018.00149.

[8] R. Kumar, M. F. Hassan, and M. H. M. Adnan, "A Principled Design of Intelligent Agent for the SLA negotiation process in cloud computing," *2022 2nd International Conference on Computing and Information Technology (ICCIT)*, 2022, pp. 383-387, https://doi.org/10.1109/ICCIT52419.2022.9711663.

[9] M. Dave and A. B. Saxena, "Loss of trust at IAAS level: Causing factors & mitigation techniques," *2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN)*, 2017, pp. 137-143, https://doi.org/10.1109/IC3TSN.2017.8284465.

[10] Z. Shi, S. Farshidi, H. Zhou, and Z. Zhao, "An Auction and Witness Enhanced Trustworthy SLA Model for Decentralized Cloud Marketplaces," Proceedings of the Conference on Information Technology for Social Good, 2021, pp. 109-114, https://doi.org/10.1145/3462203.3475876.
[11] R. Cunha, B. Veloso, and B. Malheiro, "Renegotiation of Electronic Brokerage Contracts," In: Rocha, Á., Correia, A., Adeli, H., Reis, L., Costanzo, S. (eds) Recent Advances in Information Systems and Technologies, *WorldCIST 2017, Advances in Intelligent Systems and Computing*, vol. 570, 2017, https://doi.org/10.1007/978-3-319-56538-5_5.
[12] S. Demigha, "A Cloud Management Case-Based Teaching System for Radiology," *in ECEL 2018 17th European Conference on e-Learning*, p. 134. Academic Conferences and publishing limited, 2018, https://books.google.co.id/books?id=Jox5DwAAQBAJ.
[13] P. K. Upadhyay, A. Pandita, and N. Joshi, "Scaled Conjugate Gradient Backpropagation based SLA Violation Prediction in Cloud Computing," *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, 2019, pp. 203-208, https://doi.org/10.1109/ICCIKE47802.2019.9004240.
[14] K. Lu, R. Yahyapour, P. Wieder, E. Yaqub, M. Abdullah, B. Schloer, and C. Kotsokalis, "Fault-tolerant Service Level Agreement lifecycle management in clouds using actor system," *Future Generation Computer System Journal*, vol. 54, pp. 247-259, 2016, https://doi.org/10.1016/j.future.2015.03.016.
[15] B. B. Rad, T. Diaby, and M. E. Rana, " Cloud Computing Adoption: A Short Review of Issues and Challenges," *2017 International Conference on E-commerce, E-Business and E-Government*, 2017, pp. 51-55. https://doi.org/10.1145/3108421.3108426.
[16] A. F. M. Hani, I. V. Paputungan, M. F. Hassan, and V. S. Asirvadam, "Manifold learning in SLA violation detection and prediction for cloud-based system," in *Proceedings of the Second International Conference on Internet of things, Data and Cloud Computing*, 2017, pp. 1-5, https://doi.org/10.1145/3018896.3056800.
[17] H. Sabireen and V. Neelanarayanan, "A Review on Fog Computing: Architecture, Fog with IoT, Algorithms and Research Challenges," *ICT Express*, vol. 7, no. 2, pp. 162-176, 2021, https://doi.org/10.1016/j.icte.2021.05.004.
[18] A. Omezzine, N. B. B Saoud, S. Tazi, and G. Cooperman, "Towards a generic multilayer negotiation framework for Efficient application provisioning in the cloud," *Concurrency and Computation: Practice and Experience*, 2017, https://doi.org/10.1002/cpe.4182.
[19] J. Bendriss, I. G. B. Yahia and D. Zechlache, "Forecasting and Anticipating SLO Breaches in Programmable Networks," in *International Conference on Innovations in Clouds, Internet and Networks*, 2017, pp. 127-134, https://doi.org/10.1109/ICIN.2017.7899402.
[20] M. R. Mesbahi, A. M. Rahmani, and M. Hosseinzadeh, "Reliability and high availability in cloud computing environments: a reference roadmap," *Human-centric Computing and Information Sciences*, vol. 8, 2018, https://doi.org/10.1186/s13673-018-0143-8.
[21] P. Pradeepa and R. PushpaLakshmi, "Violation Detection in Service Level Agreement to Ensure the Privacy in Cloud Community using Chicken Spider Monkey Optimization-Based Deep Belief Network," *Wireless Personal Communications*, pp. 1659–1683, 2021, https://doi.org/10.1007/s11277-020-07940-9.
[22] O. F. Rana, M. Warnier, T. B. Quillinan, F. Brazier, and D. Cojocarasu, "Managing Violations in Service level agreements," in *5th International Workshop on Grid Economics and Business Models*, 2008, pp. 349-358, https://doi.org/10.1007/978-0-387-78446-5_23.
[23] Y. Sun and N. Zhang, "A resource-sharing model based on a repeated game in fog computing," *Saudi Journal of Biological Sciences*, vol. 24, no. 3, pp. 687-694, 2017, https://doi.org/10.1016/j.sjbs.2017.01.043.
[24] I. V. Paputungan, A. F. M. Hani, M. F. Hassan, and V. S. Asirvadam, "Real-time and proactive SLA renegotiation for a cloud-based system," *IEEE Systems Journal*, vol. 13, no. 1, pp. 400-411, 2018, https://doi.org/10.1109/JSYST.2018.2805293.
[25] S. Maheswari and J. Selwyn, "A Review on The Quality of Service of Web Services," *International Journal of Mechanical Engineering and Technology (IJMET)*, vol. 9, no. 12, pp. 414-424, 2018, https://iaeme.com/MasterAdmin/Journal_uploads/IJMET/VOLUME_9_ISSUE_12/IJMET_09_12_045.pdf.
[26] ISO/IEC, "Information Technology – Cloud Computing – Service Level Agreement (SLA) Framework – Part 1: Overview and concepts," *International Standard*, ISO/IEC 19086-1, 2016.
[27] S. Shukla, M. F. Hassan, D. C. Tran, R. Akbar, I. V. Paputungan, and M. K. Khan, "Improving latency in Internet-of-Things and cloud computing for real-time data transmission: a systematic literature review (SLR)," *Cluster Computing*, pp. 1-24, 2021, https://doi.org/10.1007/s10586-021-03279-3.
[28] R. Loomba, T. Metsch, L. Feehan, and J. Butler, "Utility-Driven Deployment Decision Making," *The 10th International Conference on Utility and Cloud Computing*, 2017, pp. 207–208. https://doi.org/10.1145/3147213.3149375.
[29] H. Morshedlou and M. R. Meybodi, "Insurance for Improving User Satisfaction Level," *Business & Information Systems Engineering*, Vol. 60, pp. 513–524, 2018, https://doi.org/10.1007/s12599-017-0492-2.
[30] W. Fdhila, C. Indiono, S. Rinderle-Ma, and R. Vetschera, "Multi-criteria Decision Analysis for Change Negotiation in Process Collaborations," *2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC)*, 2017, pp. 175-183, https://doi.org/10.1109/EDOC.2017.31.
[31] S. Adabi, M. Mosadeghi, and S. Yazdani, "A real-world inspired multi-strategy based negotiating system for cloud service market," *Journal of Cloud Computing*, vol. 7, 2018, https://doi.org/10.1186/s13677-018-0116-5.
[32] B. Pittl, W. Mach, and S. Schikuta, "A Classification of Autonomous Bilateral Cloud SLA Negotiation Strategies," in *International Conference on Information Integration and Web-based Applications and Services*, 2016, pp. 379-388, https://doi.org/10.1145/3011141.3011159.

[33] Y. Li and H. Shang, "Service quality, perceived value, and citizens' continuous-use intention regarding e-government: Empirical evidence from China," *Information & Management*, vol. 57, no. 3, pp. 1-15, 2020, https://doi.org/10.1016/j.im.2019.103197.
[34] V. C. Emeakaroha, M. A. S. Netto, R. N. Calheiros, I. Brandic, R. Buyya, and C. A. F. De Rose, "Towards Autonomic Detection of SLA Violations in Cloud Infrastructures," *Future Generation Computer Systems*, vol. 28, no. 7, pp. 1017 – 1029, 2012, https://doi.org/10.1016/j.future.2011.08.018.
[35] L. Bodenstaff, A. Wombacher, M. Reichert, and M. C. Jaeger, "Monitoring dependencies for SLAs: the MoDe4SLA approach," in *International Conference on Services Computing*, 2008, pp. 21–29, https://doi.org/10.1109/SCC.2008.120.
[36] A. Michlmayr, F. Rosenberg, P. Leitner, and S. Dustdar, "Comprehensive QoS monitoring of Web services and event-based SLA violation detection," in *4th International Workshop on Middleware for Service Oriented Computing*, ACM, 2009, pp. 1-6, https://doi.org/10.1145/1657755.1657756.
[37] C. C. Li, Z. S. Ji, F. Wang, P. Wang, Y. Wang, and Z. C. Zhang, "The network monitoring system based on Cacti for EAST," *2016 IEEE-NPSS Real Time Conference (RT)*, 2016, pp. 1-5, https://doi.org/10.1109/RTC.2016.7543086.
[38] J. Renita and N. E. Elizabeth, "Network's server monitoring and analysis using Nagios," *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2017, pp. 1904-1909, https://doi.org/10.1109/WiSPNET.2017.8300092.
[39] Paessler, "PRTG Manual, Comprehensive IT Monitoring," in *PRTG The Monitoring Experts*, 2022.
[40] S. M. Musa, A. Yousif, and M. B. Bashi, "SLA Violation Detection Mechanism for Cloud Computing," *International Journal of Computer Applications*, vol. 133, no. 6, 2016, https://doi.org/10.5120/ijca2016907483.
[41] P. Predeepa and P. Pushpalakshmi, "Violation Detection in Service Level Agreement to Ensure the Privacy in Cloud Community using Chicken Spider Monkey Optimization-Based Deep Belief Network," *Wireless Personal Communications*, vol. 117, no. 3, pp. 1-25, 2021, https://doi.org/10.1007/s11277-020-07940-9.
[42] T. S. Wong, G. Y. Chan, and F. F. Chua, "A Machine Learning Model for Detection and Prediction of Cloud Quality of Service Violation, " in *Computational Science and Its Applications – ICCSA 2018*, Lecture Notes in Computer Science, vol. 10960, 2018, https://doi.org/10.1007/978-3-319-95162-1_34.
[43] W. Hussain, F. K. Hussain, O. Hussain, R. Bagia, and E. Chang, "Risk-based framework for SLA violation abatement from the cloud service provider's perspective," *The Computer Journal*, vol. 61, no. 9, pp. 1306–1322, 2018, https://doi.org/10.1093/comjnl/bxx118.

BIOGRAPHY OF AUTHORS

Irving Vitra Paputungan received his B.S. degree in IT from Universitas Islam Indonesia in 2003 and his M.S. degree in IT from the Universitas Teknologi PETRONAS Malaysia in 2008, where he also obtained his Ph.D. degree in 2020. From 2004 to date, he has been a Lecturer with the Universitas Islam Indonesia. His research interests include the areas of modeling, optimization, data management, and sports science.

Akmal Kurniadi Denna was a student of Informatics Department at Universitas Islam Indonesia. He obtained his B. S. degree in 2021.

Devi Rachmawati is currently pursuing a master's degree in Informatics Department at Universitas Islam Indonesia after receiving her degree in Nursing Science from Gadjah Mada University.