RANCANG BANGUN ARSITEKTUR *MICROSERVICES* MENGGUNAKAN *GOOGLE REMOTE PROCEDURE CALL (GRPC) APPLICATION PROGRAMMING INTERFACE (API)* DAN METODE *DOMAIN-DRIVEN DESIGN*

SKRIPSI

Diajukan untuk Memenuhi Sebagian dari
Syarat Memperoleh Gelar Sarjana Komputer
pada Program Studi Ilmu Komputer

oleh
Sekar Madu Kusumawardani
2007703

PROGRAM STUDI ILMU KOMPUTER
FAKULTAS PENDIDIKAN MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS PENDIDIKAN INDONESIA
2024

RANCANG BANGUN ARSITEKTUR *MICROSERVICES* MENGGUNAKAN *GOOGLE REMOTE PROCEDURE CALL (GRPC) APPLICATION PROGRAMMING INTERFACE (API)* DAN METODE *DOMAIN-DRIVEN DESIGN*

Oleh
Sekar Madu Kusumawardani
2007703

Sebuah Skripsi yang Diajukan untuk Memenuhi Salah Satu Syarat Memperoleh Gelar Sarjana Komputer pada Fakultas Pendidikan Matematika dan Ilmu Pengetahuan Alam

SEKAR MADU KUSUMAWARDANI

2007703

RANCANG BANGUN ARSITEKTUR *MICROSERVICES* MENGGUNAKAN *GOOGLE REMOTE PROCEDURE CALL (GRPC) APPLICATION PROGRAMMING INTERFACE (API)* DAN METODE *DOMAIN-DRIVEN DESIGN*

Disetujui dan disahkan oleh:

Pembimbing I,

Dr. Asep Wahyudin, M.T.

NIP. 197112232006041001

Pembimbing II,

Herbert Siregar, M.T.

NIP. 197005022008121001

Mengetahui,

Kepala Program Studi Ilmu Komputer

Dr. Muhamad Nursalman, M.T.

NIP 197909292006041002

RANCANG BANGUN ARSITEKTUR *MICROSERVICES* MENGGUNAKAN *GOOGLE REMOTE PROCEDURE CALL (GRPC) APPLICATION PROGRAMMING INTERFACE (API)* DAN METODE *DOMAIN-DRIVEN DESIGN*

Oleh

Sekar Madu Kusumawardani – sekarmadu@upi.edu

2007703

ABSTRAK

Aplikasi Tracer Study UPI merupakan suatu sistem *online* yang dapat digunakan oleh Perguruan Tinggi untuk melacak aktivitas para lulusannya setelah masa pendidikan tinggi. Penting bagi UPI sebagai suatu organisasi pendidikan untuk terus mengembangkan sistem tersebut dengan mengikuti perkembangan teknologi agar sistem mampu beradaptasi terhadap kemajuan yang ada serta menjadi perguruan tinggi yang adaptif dan tidak kehilangan relevansinya. Aplikasi Tracer Study UPI saat ini masih menerapkan arsitektur *monolithic* yang memiliki banyak kekurangan yang mengurangi poin adaptif organisasi, yaitu dari sisi *scalability, maintainability, fault tolerant,* dan *performance.* Penelitian ini bertujuan untuk merancang dan membangun arsitektur *microservices* pada aplikasi Tracer Study UPI menggunakan *gRPC API* untuk mengatasi masalah penerapan arsitektur *monolithic* pada aplikasi Tracer Study UPI. Metode *Domain-Driven Design* pada penelitian ini digunakan untuk mendekomposisi aplikasi Tracer Study yang sudah ada dan merancang ulang sistem tersebut dengan menerapkan arsitektur *microservices*. Penelitian ini menguji dan mengevaluasi hasil pengimplementasian arsitektur *microservices* pada aplikasi Tracer Study UPI dengan berfokus pada pengujian kinerja atau *performance testing.* Hasilnya yaitu implementasi arsitektur *microservices* pada Tracer Study UPI yang berjalan baik sesuai dengan fungsionalitas awal. Dari pengujian kinerja yang dilakukan juga ditemukan bahwa terjadi peningkatan dari sisi kinerja pada Tracer Study UPI setelah diterapkannya arsitektur *microservices.*

Kata Kunci: *microservices*, *tracer study*, *domain-driven design, DDD, gRPC*, perguruan tinggi adaptif, sistem informasi perguruan tinggi, migrasi *monolithic*

MICROSERVICES ARCHITECTURE DESIGN AND DEVELOPMENT USING GOOGLE REMOTE PROCEDURE CALL (GRPC) APPLICATION PROGRAMMING INTERFACE (API) AND DOMAIN-DRIVEN DESIGN METHOD

Arranged by
Sekar Madu Kusumawardani – sekarmadu@upi.edu
2007703

ABSTRACT

The UPI Tracer Study application is an online system that can be used by universities to track the activities of their graduates after higher education. It is important for UPI as an educational organization to continue to develop the system by following technological developments so that the system is able to adapt to existing advances and become an adaptive university and not lose its relevance. The current UPI Tracer Study application still implements a monolithic architecture that has many shortcomings that reduce the adaptive points of the organization, namely in terms of scalability, maintainability, fault tolerant, and performance. This research aims to design and build a microservices architecture on the UPI Tracer Study application using the gRPC API to overcome the problem of implementing a monolithic architecture in the UPI Tracer Study application. The Domain-Driven Design method in this study was used to decompose the existing Tracer Study application and redesign the system by implementing a microservices architecture. This research tests and evaluates the results of implementing microservices architecture on the UPI Tracer Study application by focusing on performance testing. The result is the implementation of microservices architecture on Tracer Study UPI which runs well according to the initial functionality. From the performance testing conducted, it was also found that there was an increase in terms of performance on Tracer Study UPI after the implementation of microservices architecture.

Keywords: microservices, tracer study, domain-driven design, DDD, gRPC, adaptive organization, information system, monolithic migration

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

DAFTAR PUSTAKA

Abbas, R., Sultan, Z., & Bhatti, D. S. N. (2017). Comparative Analysis of Automated Load Testing Tools: Apache JMeter, Microsoft Visual Studio (TFS), LoadRunner, Siege. *2017 International Conference on Communication Technologies (ComTech)*, 39–44.

Abbot, M. L., & Fisher, M. T. (2015). *The Art of Scalability, Second Edition* (Second). Addison-Wesley Professional.

Abgaz, Y., Mccarren, A., Elger, P., Solan, D., Lapuz, N., Bivol, M., Jackson, G., Yilmaz, M., Buckley, J., & Clarke, P. (2023). Decomposition of Monolith Applications Into Microservices Architectures: A Systematic Review. *IEEE Transactions on Software Engineering*, *49*(8), 4213–4242. https://doi.org/10.1109/TSE.2023.3287297

Ahmadvand, M., & Ibrahim, A. (2017). Requirements Reconciliation for Scalable and Secure Microservice (De)composition. *Proceedings - 2016 IEEE 24th International Requirements Engineering Conference Workshops, REW 2016*, 68–73. https://doi.org/10.1109/REW.2016.14

Akbar, R., & Mukhtar. (2020). Perancangan E-Tracer Study Berbasis Sistem Cerdas. *Sistem Informasi Dan Komputer*, *09*, 8–12. https://doi.org/10.32736/sisfokom.v9.i1.631

Al Hilmi, M. A., Muhamad, F. P. B., Cahyanto, K. A., & Mutahari, S. R. (2022). Penerapan Microservices pada Pengembangan Aplikasi Manajemen Kegiatan Rumah Tahfiz Qur'an Ulil Albab Kabupaten Indramayu. *IKRA-ITH Informatika: Jurnal Komputer Dan Informatika*, *6*(2), 63–72.

Alankar, B., Sharma, G., Kaur, H., Valverde, R., & Chang, V. (2020). Experimental Setup for Investigating the Efficient Load Balancing Algorithms on Virtual Cloud. *Sensors (Switzerland)*, *20*(24), 1–26. https://doi.org/10.3390/s20247342

Al-Debagy, O., & Martinek, P. (2018). A Comparative Review of Microservices and Monolithic Architectures. *18th IEEE International Symposium on Computational Intelligence and Informatics (CINTI)*, 149–154. https://doi.org/10.1109/CINTI.2018.8928192

Amiri, M. J. (2018). Object-Aware Identification of Microservices. *Proceedings - 2018 IEEE International Conference on Services Computing, SCC 2018 - Part of the 2018 IEEE World Congress on Services*, 253–256. https://doi.org/10.1109/SCC.2018.00042

Anderson, Ross. (2020). *Security Engineering: A Guide to Building Dependable Distributed Systems* (3rd ed.). John Wiley & Sons.

Apache. (1999). *Apache JMeter*. The Apache Software Foundation. https://jmeter.apache.org/

Arevalo, M., Escobar, C., Monasse, P., Monzon, N., & Colom, M. (2017). The IPOL Demo System: A Scalable Architecture of Microservices for Reproducible Research. In B. Kerautret, M. Colom, & P. Monasse (Eds.), *Kerautret, B., Colom, M., Monasse, P. (eds) Reproducible Research in Pattern Recognition* (Vol. 10214, pp. 3–16). Springer International Publishing. https://doi.org/10.1007/978-3-319-56414-2_1

Ashby, D., & Jensen, C. T. (2018). *APIs for Dummies* (3rd ed.). John Wiley & Sons, Inc. http://www.wiley.com/go/permissions.

Auer, F., Lenarduzzi, V., Felderer, M., & Taibi, D. (2021). From Monolithic Systems to Microservices: An Assessment Framework. *Information and Software Technology*, *137*. https://doi.org/10.1016/j.infsof.2021.106600

Bagci, H., & Kara, A. (2016). A Lightweight and High Performance Remote Procedure Call Framework for Cross Platform Communication. *ICSOFT 2016 - Proceedings of the 11th International Joint Conference on Software Technologies*, *1*, 117–124. https://doi.org/10.5220/0005931201170124

Bakhtiar, M. I., & Latif, S. (2017). Tracer Study Alumni: Upaya Pengembangan Prodi Bimbingan Konseling Universitas Negeri Makassar. *Jurnal Kajian Bimbingan Dan Konseling*, *2*(1), 32–40. http://journal2.um.ac.id/index.php/jkbk

Balalaie, A., Heydarnoori, A., & Jamshidi, P. (2016). Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture. *IEEE Software*, *33*(3), 42–52. https://doi.org/10.1109/MS.2016.64

Balalaie, A., Heydarnoori, A., Jamshidi, P., Tamburri, D. A., & Lynn, T. (2018). Microservices Migration Patterns. *Software - Practice and Experience*, *48*(11), 2019–2042. https://doi.org/10.1002/spe.2608

Banijamali, A., Kuvaja, P., Oivo, M., & Jamshidi, P. (2020). Kuksa * : Self-Adaptive Microservices in Automotive Systems. *Product-Focused Software Process Improvement, 21st International Conference, PROFES 2020*, 367–384. https://doi.org/10.1007/978-3-030-64148-1_23

Baresi, L., Garriga, M., & De Renzis, A. (2017). Microservices Identification Through Interface Analysis. In F. De Paoli, S. Schulte, & E. Broch Johnsen (Eds.), *European Conference on Service-Oriented and Cloud Computing* (Vol. 10465, pp. 19–33). Springer International Publishing. https://doi.org/10.1007/978-3-319-67262-5

Blinowski, G., Ojdowska, A., & Przybylek, A. (2022). Monolithic vs. Microservice Architecture: A Performance and Scalability Evaluation. *IEEE Access*, *10*, 20357–20374. https://doi.org/10.1109/ACCESS.2022.3152803

Bolanowski, M., Żak, K., Paszkiewicz, A., Ganzha, M., Paprzycki, M., Sowiński, P., Lacalle, I., & Palau, C. E. (2022). Eficiency of REST and gRPC Realizing Communication Tasks in Microservice-Based Ecosystems. *The 21st International Conference on Intelligent Software Methodologies, Tools, and Techniques (SoMeT 2022)*. https://orcid.org/0000-0002-

Chaieb, M., & Saied, M. A. (2023). Automate Migration to Microservices Architecture using Machine Learning Techniques. *Journal of Systems and Software*, 1–60. https://doi.org/10.48550/arXiv.2301.06508

Chen, R., Li, S., & Li, Z. (2017). From Monolith to Microservices: A Dataflow-Driven Approach. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, *2017-December*, 466–475. https://doi.org/10.1109/APSEC.2017.53

Clarke, P., O'Connor, R. V., & Leavy, B. (2016). A Complexity Theory Viewpoint on the Software Development Process and Situational Context. *Proceedings - International Conference on Software and System Process, ICSSP 2016*, 86–90. https://doi.org/10.1145/2904354.2904369

Cojocaru, M., Uta, A., & Oprescu, A. M. (2019). MicroValid: A Validation Framework for Automatically Decomposed Microservices. *Proceedings of the International Conference on Cloud Computing Technology and Science, CloudCom*, *2019-December*, 78–86. https://doi.org/10.1109/CloudCom.2019.00023

Cooksey, B. (2014). *An Introduction to APIs*. Zapier, Inc. https://zapier.com/learn/apis

Cox, R., Griesemer, R., Pike, R., Taylor, I. L., & Thompson, K. (2022). The Go Programming Language and Environment. *Communications of the ACM*, *65*(5), 70–78. https://doi.org/10.1145/3488716

Crispin, L., & Gregory, J. (2009). *Agile Testing: A Practical Guide for Testers and Agile Teams*. Addison-Wesley. www.XProgramming.com

D Dikti. (2012). *Buku Panduan Pusat Karir* (Vol. 2). Kementerian Riset Teknologi dan Pendidikan Tinggi Republik Indonesia.

Dahri, F., Hanafi, A. M. El, Handoko, D., & Wulan, N. (2022). Implementation of Microservices Architecture in Learning Management System E-Course Using Web Service Method. *Sinkron*, *7*(1), 76–82. https://doi.org/10.33395/sinkron.v7i1.11229

De Camargo, A., Dos Santos Mello, R., Salvadori, I., & Siqueira, F. (2016). An Architecture to Automate Performance Tests on Microservices. *ACM International Conference Proceeding Series*, 422–429. https://doi.org/10.1145/3011141.3011179

Di Francesco, P., Lago, P., & Malavolta, I. (2018). Migrating Towards Microservice Architectures: An Industrial Survey. *Proceedings - 2018 IEEE 15th International Conference on Software Architecture, ICSA 2018*, 29–38. https://doi.org/10.1109/ICSA.2018.00012

Direktorat Jenderal Pendidikan Tinggi. (2011). *Tracer Study Kemendikbud: Tentang Tracer Study*. Https://Tracerstudy.Kemdikbud.Go.Id/. https://tracerstudy.kemdikbud.go.id/

Dragoni, N., Giallorenzo, S., Lafuente, A. L., Mazzara, M., Montesi, F., Mustafin, R., & Safina, L. (2017). Microservices: Yesterday, today, and tomorrow. In *Present and Ulterior Software Engineering* (pp. 195–216). Springer International Publishing. https://doi.org/10.1007/978-3-319-67425-4_12

Elgheriani, N. S., & Ahmed, N. A. S. (2022). Microservices Vs. Monolithic Architectures [The Differential Structure Between Two Architectures]. *MINAR International Journal of Applied Sciences and Technology*, *4*(3), 500–514. https://doi.org/10.47832/2717-8234.12.47

Escobar, D., Cardenas, D., Amarillo, R., Castro, E., Garces, K., Parra, C., & Casallas, R. (2016). Towards the Understanding and Evolution of Monolithic Applications as Microservices. *2016 XLII Latin American Computing Conference (CLEI)*, 1–11. https://doi.org/10.1109/CLEI.2016.7833410

Evans, E. (2003). *Domain-Driven Design Tackling Complexity in the Heart of Software*. www.domainlanguage.com

Fan, C. Y., & Ma, S. P. (2017). Migrating Monolithic Mobile Application to Microservice Architecture: An Experiment Report. *2017 IEEE 6th International Conference on AI and Mobile Services (AIMS)*, 109–112. https://doi.org/10.1109/AIMS.2017.23

Ferdinand, J., Syahrina, A., & Musnansyah, A. (2021). Perancangan Arsitektur Perangkat Lunak Microservices pada Aplikasi Open Library Telkom University Menggunakan gRPC. *E-Proceeding of Engineering*, *8*, 9543–9550.

Fowler, M. (2004, June). *Strangler Fig Application*. Https://Martinfowler.Com/Bliki/StranglerFigApplication.Html. https://martinfowler.com/bliki/StranglerFigApplication.html

Fritzsch, J., Bogner, J., Zimmermann, A., & Wagner, S. (2019). From Monolith to Microservices: A Classification of Refactoring Approaches. *International Workshop on Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment*, 128–141. https://doi.org/10.48550/arXiv.1807.10059

Google. (2015a). *gRPC*. Https://Grpc.Io. https://grpc.io

Google. (2015b). *Protocol Buffer Documentation*. Https://Protobuf.Dev/. https://protobuf.dev/

Guamán, D., Yaguachi, Lady, Samanta, C. C., Danilo, J. H., & Soto, F. (2018). Performance Evaluation in the Migration Process from a Monolithic Application to Microservices. *2018 13th Iberian Conference on Information Systems and Technologies (CISTI)*, 1–8. https://doi.org/10.23919/CISTI.2018.8399148

Guo, D., Wang, W., Zhang, J., Xiang, Q., Huang, C., Chang, J., & Zhang, L. (2016). Cloudware: An Emerging Software Paradigm for Cloud Computing. *ACM International Conference Proceeding Series*, *18-September-2016*, 1–10. https://doi.org/10.1145/2993717.2993718

Gysel, M., Kölbener, L., Giersche, W., & Zimmermann, O. (2016). Service Cutter: A Systematic Approach to Service Decomposition. *Aiello, M., Johnsen, E., Dustdar, S., Georgievski, I. (Eds) Service-Oriented and Cloud Computing. ESOCC 2016. Lecture Notes in Computer Science*, *9846 LNCS*, 185–200. https://doi.org/10.1007/978-3-319-44482-6_12

Hammad, H., Sahmoud, T., & Ghazala, A. A. R. A. (2023). Convert Monolithic Application to Microservice Application. *ArXiv*. https://doi.org/10.48550/arXiv.2306.08851

Hermawan, I., & Suharnomo, S. (2020). Information Technology as a Strategic Resource in Encouraging Organizational Change Readiness through the Role of the Human Capital Effectiveness. *Jurnal Dinamika Manajemen*, *11*(2), 242–254. https://doi.org/10.15294/jdm.v11i2.23700

Hoffer, J. A., Ramesh, V. (Venkataraman), & Topi, H. (2019). *Modern Database Management* (13th ed.). Pearson Education.

Hosea, E., Novianus Palit, H., & Dewi, L. P. (2021). Fault Tolerance pada Microservice Architecture dengan Circuit Breaker dan Bulkhead Pattern. *Jurnal Infra*, *9*.

Husaini, M., & IAIN, R. I. (2014). Pemanfaatan Teknologi Informasi dalam Bidang Pendidikan (E-Education). *Jurnal Mikrotik*, *2*(1).

I. J. Munezero, D. -T. Mukasa, B. Kanagawa, & J. Balikuddembe. (2018). Partitioning microservices: A domain engineering approach. *2018 IEEE/ACM Symposium on Software Engineering in Africa (SEiA)*, 43–49. https://doi.org/10.1145/3195528.3195535

Imran, M., Sasudin, M., Rusli, H. M., & Kama, N. (2022). Monolith Application to Microservices Model Driven Analysis Migration: State-of-The-Art Techniques Article history. *Open International Journal of Informatics (OIJI*, *10*(2). https://doi.org/10.11113/oiji2022.10n2.231

Indrasiri, K., & Kuruppu, D. (2020). *gRPC: Up & Running (Building Cloud Native Applications with Go and Java for Docker and Kubernetes)*. O'Reilly Media, Inc.

Ismail, A., Ananta, A. Y., Arief, S. N., & Hamdana, E. N. (2023). Performance Testing Sistem Ujian Online Menggunakan JMeter pada Lingkungan Virtual. *JIP (Jurnal Informatika Polinema)*, *9*(2), 159–164.

Janes, A., & Russo, B. (2019). Automatic performance monitoring and regression testing during the transition from monolith to microservices. *2019 IEEE 30th International Symposium on Software Reliability Engineering Workshops (ISSREW)*, 163–168. https://doi.org/10.1109/ISSREW.2019.00067

Joko, B. S. (2010). Sistem Informasi Manajemen Perguruan Tinggi Dalam Bidang Pendataan Pendidikan Tinggi. *Jurnal Pendidikan Dan Kebudayaan*, *16*(2), 146–156. https://doi.org/10.24832/jpnk.v16i2.442

Kalske, M., Mäkitalo, N., & Mikkonen, T. (2018). Challenges When Moving from Monolith to Microservice Architecture. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10544 LNCS*, 32–47. https://doi.org/10.1007/978-3-319-74433-9_3

Karwowski, W., Rusek, M., Dwornicki, G., & Orłowski, A. (2018). Swarm Based System for Management of Containerized Microservices in a Cloud Consisting of Heterogeneous Servers. *Advances in Intelligent Systems and Computing*, *655*, 262–271. https://doi.org/10.1007/978-3-319-67220-5_24

Khoirunnisa, L. (2019). *Rancang Bangun Sistem E-Learning Berbasis Microservices dan Domain Driven Design (Studi Kasus Probistek UIN Maulana Malik Ibrahim Malang)*. UIN Maulana Malik Ibrahim Malang.

Kim, S.-H., Lee, Y., & Kim, J.-S. (2007). FlexRPC: A Flexible Remote Procedure Call Facility for Modern Cluster File Systems. *2007 IEEE International Conference on Cluster Computing*, 275–284. https://doi.org/10.1109/CLUSTR.2007.4629241

Kore, P. P., Lohar, M. J., Surve, M. T., & Jadhav, S. (2022). API Testing Using Postman Tool. *International Journal for Research in Applied Science and Engineering Technology*, *10*(12), 841–843. https://doi.org/10.22214/ijraset.2022.48030

Koschel, A., Astrova, I., & Dotterl, J. (2017). Making the Move to Microservice Architecture. *International Conference on Information Society (i-Society)*, 71–79. https://doi.org/10.23919/i-Society.2017.8354675

Laudon, K. C., & Laudon, J. P. (2012). *Management Information System: Managing Digital Firm* (12th ed.). Pearson Education.

Levcovitz, A., Terra, R., & Valente, M. T. (2016, May 10). Towards a Technique for Extracting Microservices from Monolithic Enterprise Systems. *3rd Brazilian Workshop on Software Visualization, Evolution and Maintenance*. https://doi.org/10.48550/arXiv.1605.03175

Lewis, J., & Fowler, M. (2014, March 25). *Microservices: A Definition of This New Architectural Term*. Https://Martinfowler.Com/Articles/Microservices.Html. https://martinfowler.com/articles/microservices.html

Li, C. Y., Ma, S. P., & Lu, T. W. (2020). Microservice Migration Using Strangler Fig Pattern: A Case Study on the Green Button System. *Proceedings - 2020 International Computer Symposium, ICS 2020*, 519–524. https://doi.org/10.1109/ICS51289.2020.00107

Li, S., Zhang, H., Jia, Z., Li, Z., Zhang, C., Li, J., Gao, Q., Ge, J., & Shan, Z. (2019). A Dataflow-Driven Approach to Identifying Microservices from Monolithic Applications. *Journal of Systems and Software*, *157*. https://doi.org/10.1016/j.jss.2019.07.008

Ma, S. P., Li, C. Y., Lee, W. T., & Lee, S. J. (2022). Microservice Migration Using Strangler Fig Pattern and Domain-Driven Design. *Journal of Information*

Science and Engineering, *38*(6), 1285–1303. https://doi.org/10.6688/JISE.202211_38(6).0010

Martin, R. C. (2014). *Agile Software Development, Principles, Patterns, and Practices* (1st ed., Vol. 1). Pearson Education Limited.

Maulana, H. (2023). Refactoring Arsitektur Microservice pada Aplikasi Management Information System of LP3I Menggunakan Strangler Pattern. *Jurnal Explora Informatika*, *11*, 140–148. https://doi.org/10.30864/eksplora.v11i2.888

Mazlami, G., Cito, J., & Leitner, P. (2017). Extraction of Microservices from Monolithic Software Architectures. *IEEE 24th International Conference on Web Services (ICWS 2017)*, 524–531. https://doi.org/10.1109/ICWS.2017.61

Megargel, A., Shankararaman, V., & Walker, D. K. (2020). Migrating from Monoliths to Cloud-Based Microservices: A Banking Industry Example. In *Research Collection School Of Information Systems* (pp. 85–108). Springer. https://doi.org/10.1007/978-3-030-33624-0_4

Meier, J. D., Farre, C., Banshode, P., Barber, S., & Rea, D. (2007). *Performance Testing Guidance for Web Applications* (1st ed.). Microsoft Press.

Mendonca, N. C., Box, C., Manolache, C., & Ryan, L. (2021). The Monolith Strikes Back: Why Istio Migrated from Microservices to a Monolithic Architecture. *IEEE Software*, *38*(5), 17–22. https://doi.org/10.1109/MS.2021.3080335

Mili, A., & Tchier, F. (2015). *Software Testing: Concepts and Operations* (Vol. 1). John Wiley & Sons. www.allitebooks.com

Mithas, S., & Krishnan, M. S. (2008). Human Capital and Institutional Effects in the Compensation of Information Technology Professionals in the United States. *Management Science*, *54*(3), 415–428. https://doi.org/10.1287/mnsc.1070.0778

Molyneaux, I. (2009). *The Art of Application Performance Testing* (1st ed., Vol. 1). O'Reilly Media.

Mufrizal, R., & Indarti, D. (2019). Refactoring Arsitektur Microservice Pada Aplikasi Absensi PT. Graha Usaha Teknik. *Jurnal Nasional Teknologi Dan Sistem Informasi*, *5*(1), 57–68. https://doi.org/10.25077/teknosi.v5i1.2019.57-68

Myers, G. J., Badgett, T., & Sandler, C. (2012). *The Art of Software Testing* (3rd ed.). John Wiley & Sons, Inc.

Namiot, D., & Sneps-Sneppe, M. (2014). On Micro-services Architecture. *International Journal of Open Information Technologies*, *2*(9), 2307–8162. https://www.researchgate.net/publication/265292970

Newman, S. (2020). *Monolith to Microservices Evolutionary Patterns to Transform Your Monolith*. O'Reilly Media.

Newman, S. (2021). *Building Microservices: Designing Fine-Grained Systems* (2nd ed.). O'Reilly Media.

Nizam. (2021). *Membangun Sistem Pendidikan Tinggi Indonesia 4.0*.

Noviyantono, E., & Aidil. (2012). Integration System of Web Based and SMS Gateway for Information System of Tracer Study. *International Conference on Engineering and Technology Development (ICETD)*, *12*(1).

Park, J., Moon, M., & Keunhyuk, K. (2019). Approach to Identify Microservices Based on Analysis Class Model. *International Journal of Advanced Science and Technology*, *28*(4), 8–14. http://sersc.org/journals/index.php/IJAST/article/view/289

Phatak, J. J. (2022). An Overview of Microservice Architecture Impact in Terms of Scalability and Reliability in E-Commerce: A Case Study on Uber and Otto.De. *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT*, *2*(1). https://doi.org/10.48175/568

Poerwanto, Sisbintari, I., & Suhartono. (2013). Transformasi Organisasi Basis Peningkatan Sumber Daya Manusia dalam Memperkuat Daya Saing. *Jurnal Al-Azhar Indonesia Seri Pranata Sosial*, *2*(2), 119–132.

Posta, C. (2016). *Microservices for Java Developers: A Hands-On Introduction to Frameworks and Containers* (1st ed.). O'Reilly Media.

Pradeep, S., & Sharma, Y. K. (2019). A Pragmatic Evaluation of Stress and Performance Testing Technologies for Web Based Applications. *Proceedings - 2019 Amity International Conference on Artificial Intelligence, AICAI 2019*, 399–403. https://doi.org/10.1109/AICAI.2019.8701327

Prasojo, L. D. (2009). Sistem Manajemen Perguruan Tinggi Modern. *Dinamika Pendidikan*, *16*(1), 98–109. https://journal.uny.ac.id/index.php/dinamika-pendidikan/article/view/4121

Pressman, R. F. (2010). *Software Engineering: A Practitioner's Approach* (7th ed.). McGraw-Hill. www.mhhe.com/pressman.

Rademacher, F., Sorgalla, J., & Sachweh, S. (2018). Challenges of Domain-Driven Microservice Design A Model-Driven Perspective. *IEEE Software*, *35*(3), 36–43. https://doi.org/10.1109/MS.2018.2141028

Radhiyan, M. F. (2020). *Analisis dan Desain Arsitektur Microservices dengan GraphQL Sebagai API Gateway untuk Sistem Informasi Akademik AIS UIN Jakarta (Studi Kasus : AIS untuk Mahasiswa)*. UIN Syarif Hidayatullah.

Rajesh RV. (2016). *Spring Microservices: Build Scalable Microservices with Spring, Docker, and Mesos*. Packt Publishing.

Ramachandran, A. T., R, A., G.S., M., R., R., K., B., & Parmar, M. (2021). Understanding Migration from Monolithic to Microservice Architecture and its Challenges. *International Journal of Scientific Research and Engineering Development*, *4*, 1742–1752. www.ijsred.com

Reiser, R. A., & Dempsey, J. V. (2018). *Trends and Issues in Instructional Design and Technology* (4th ed.). Pearson Education.

Rencana Strategis (RENSTRA) Universitas Pendidikan Indonesia Tahun 2021-2025, Peraturan Majelis Wali Amanat Universitas Pendidikan Indonesia Nomor 04 Tahun 2020 Tentang Rencana Strategis Universitas Pendidikan Indonesia Tahun 2021-2025 (2021).

Rezaldy, M., Asror, I., & Sardi, I. L. (2017). Desain dan Analisis Arsitektur Microservices Pada Sistem Informasi Akademik Perguruan Tinggi Dengan Pendekatan Architecture Tradeoff Analysis Method (ATAM) (Studi Kasus: iGracias Universitas Telkom). *E-Proceeding of Engineering*, *4*(2).

Richardson, C. (2018). *Microservices Patterns: With Examples in Java* (1st ed.). Manning Publications Co.

Richardson, C. (2020). *Monolithic Architecture Pattern*. https://microservices.io/patterns/monolithic.html

Richardson, C., & Smith, F. (2016). *Microservices: From Design to Deployment*. Nginx. https://www.nginx.com/blog/microservices-from-design-to-deployment-ebook-nginx/

Riyanto, Hermadi, I., & Nurhadryani, Y. (2023). Analisis Uji Performa Aplikasi Dari Hasil Implementasi Refactoring Arsitektur Monolitik Ke Mikroservis dengan Decomposition dan Strangler Pattern. *Jurnal Sistem Cerdas*, *6*(3), 189–203.

Rizki, M., Fajar, A. N., & Retnowardhani, A. (2020). Microservices Architecture Design: Proposed for Online HealthCare. *International Journal of Emerging Trends in Engineering Research*, *8*(4), 1040–1046. https://doi.org/10.30534/ijeter/2020/14842020

Samsiah, S., Marlina, E., & Ardi, H. A. (2018). Pengaruh Knowledge Management Dan Teknologi Informasi Terhadap Keunggulan Bersaing Dan Kinerja Universitas. *Jurnal Manajemen*, *22*(2), 154–167. https://doi.org/10.24912/jm.v22i2.356

Schmidt, R. A., & Thiry, M. (2020). Microservices Identification Strategies: A Review Focused on Model-Driven Engineering and Domain Driven Design Approaches. *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, 1–6. https://doi.org/10.23919/CISTI49556.2020.9141150

Schomburg, H. (2003). *Handbook for Graduate Tracer Studies* (2nd ed.). InWent. www.uni-kassel.de/incher

Singh, V., & Peddoju, S. K. (2017). Container-based Microservice Architecture for Cloud Applications. *2017 International Conference on Computing, Communication and Automation (ICCCA)*, 847–852. https://doi.org/10.1109/CCAA.2017.8229914

Siti Rochimah, & Bintang Nuralamsyah. (2023). Decomposing Monolithic to Microservices: Keyword Extraction and BFS Combination Method to Cluster Monolithic's Classes. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *7*(2), 263–270. https://doi.org/10.29207/resti.v7i2.4866

Sivathanu, M., Arpaci-Dusseau, A. C., & Arpaci-Dusseau, R. H. (2002). Evolving RPC for Active Storage. *ACM SIGOPS Operating Systems Review*, *36*, 264–276. https://doi.org/10.1145/635508.605425

Sommerville, I. (2011). *Software Engineering* (9th ed.). Pearson Education.

Steinegger, R. H., Giessler, P., Hippchen, B., & Abeck, S. (2017). Overview of a Domain-Driven Design Approach to Build Microservice-Based Applications. *The Third International Conference on Advances and Trends in Software Engineering (SOFTENG 2017)*.

Supratman, Defit, S., & Vitriani. (2019). Indeks Kesiapan Perguruan Tinggi dalam Mengimplementasikan Smart Campus. *Jurnal Teknologi Informasi Dan Ilmu Komputer (JTIIK)*, *6*(3), 267–276. https://doi.org/10.25126/jtiik.20196986

Suryotrisongko, H. (2017). Arsitektur Microservice untuk Resiliensi Sistem Informasi. *Jurnal SISFO*, *6*(2), 235–250.

Swarnalatha, K. S., Mallya, A., Mukund, G., & Ujwal Bharadwaj, R. (2022). Solving Problems of Large Codebases Uber's Approach Using Microservice Architecture. *Emerging Research in Computing, Information, Communication and Applications*, 653–662. https://doi.org/10.1007/978-981-19-5482-5_57

Syafiq, A. (2017). *Konsep dan Implementasi Tracer Study*. Lembaga Layanan Pendidikan Tinggi Wilayah IV. https://www.lldikti4.or.id/wp-content/uploads/2017/06/Ahmad-Syafiq_KONSEP-DAN-IMPLEMENTASI-TS-2017.pdf

Taibi, D., Lenarduzzi, V., & Pahl, C. (2017). Processes, Motivations, and Issues for Migrating to Microservices Architectures: An Empirical Investigation. *IEEE Cloud Computing*, 22–32. https://doi.org/10.1109/MCC.2017.4250931

Taibi, D., & Systä, K. (2019). From Monolithic Systems to Microservices: A Decomposition Framework Based on Process Mining. *9th International Conference on Cloud Computing and Services Science*, 153–164. https://doi.org/10.5220/0007755901530164

Tanuska, P., Vlkovic, O., & Spendla, L. (2012). The Usage of Performance Testing for Information Systems. *International Journal of Computer Theory and Engineering*, 144–147. https://doi.org/10.7763/ijcte.2012.v4.439

Tapia, F., Mora, M. ángel, Fuertes, W., Aules, H., Flores, E., & Toulkeridis, T. (2020). From Monolithic Systems to Microservices: A Comparative Study of Performance. *Applied Sciences (Switzerland)*, *10*(17), 1–35. https://doi.org/10.3390/app10175797

Undang-Undang Nomor 12 Tahun 2012 Tentang Perguruan Tinggi, Pub. L. No. 12, Sekretariat Negara (2012).

UPI. (2021). *Tracer Study UPI*. https://tracerstudy.upi.edu

Vale, G., Correia, F. F., Guerra, E. M., Rosa, T. de O., Fritzsch, J., & Bogner, J. (2022). Designing Microservice Systems Using Patterns: An Empirical Study on Quality Trade-Offs. *IEEE 19th International Conference on Software Architecture Companion (ICSA-C)*, 67–79. https://doi.org/10.1109/ICSA53651.2022.00015

Villamizar, M., Garces, O., Castro, H., Verano, M., Salamanca, L., & Casallas, R. (2015). Evaluating the Monolithic And the Microservice Architecture Pattern to Deploy Web Applications in the Cloud. *10th Computing Colombian Conference (10CCC)*, 583–590. https://doi.org/10.1109/ColumbianCC.2015.7333476

Vural, H., & Koyuncu, M. (2021). Does Domain-Driven Design Lead to Finding the Optimal Modularity of a Microservice? *IEEE Access*, *9*, 32721–32733. https://doi.org/10.1109/ACCESS.2021.3060895

Wang, J., & Wu, J. (2019). Research on Performance Automation Testing Technology Based on JMeter. *2019 International Conference on Robots and Intelligent System (ICRIS)*, 55–58. https://doi.org/10.1109/ICRIS.2019.00023

Widajanti, E. (2008). Peran Teknologi Informasi untuk Mencapai Keunggulan Kompetitif. *Jurnal Akuntansi Dan Sistem Teknologi Informasi*, *6*(1), 60–71.

Wolfart, D., Assunção, W. K. G., Da Silva, I. F., Domingos, D. C. P., Schmeing, E., Villaca, G. L. D., & Paza, D. D. N. (2021). Modernizing Legacy Systems with Microservices: A Roadmap. *ACM International Conference Proceeding Series*, 149–159. https://doi.org/10.1145/3463274.3463334