DevOps: Disrupsi Pengelolaan ICT Pendidikan Tinggi

Wahyuni Puji Lestari[1], Ari Sujarwo[2]
Jurusan Teknik Informatika Fakultas Teknologi Industri
Universitas Islam Indonesia
Yogyakarta
[1]14523106@students.uii.ac.id, [2]ari.sujarwo@uii.ac.id

***Abstrak*—Disrupsi sangat penting untuk diperhatikan dalam inovasi teknologi di industri maupun pendidikan. Pengadopsian teknologi untuk proses pembelajaran dan penelitian sangat membantu proses bisnis di pendidikan tinggi. Namun masih banyak kendala yang sering dihadapi dalam penggunaan teknologi untuk ICT pendidikan tinggi. Penelitian ini dilakukan dengan kajian pustaka dari berbagai sumber yang berkaitan. Kemudian didapatkan kesimpulan bagaimana infrastruktur dan arsitektur DevOps dapat menjadi inovasi di era disrupsi teknologi.**

Kata kunci—disrupsi, ICT, infrastruktur, arsitektur, pendidikan tinggi, DevOps

I. PENDAHULUAN

Disrupsi menurut KBBI adalah hal yang tercabut dari akarnya. Namun istilah tersebut dinilai kurang tepat dalam penelitian ini, pada buku Clayton Christensen *"The innovator's Dilemma"* memperkenalkan gagasan "*disruptive innovation*" yang merupakan cara berpikir bagaimana perusahaan yang sukses, tidak hanya memenuhi kebutuhan pelanggan saat ini saja, tetapi mengantisipasi kebutuhan yang akan datang [1]. Topik disrupsi ini juga disinggung dalam "*Making Indonesia 4.0*" pada awal April 2018 oleh Kementrian Perindustrian Republik Indonesia, karena dampak revolusi industri 4.0. Menurut Lestari [2] revolusi industri 4.0 tidak hanya berdampak pada dunia perindustrian saja, namun juga pendidikan tinggi. Pola pikir revolusi industri 4.0 perlu diterapkan dalam pengelolaan pendidikan tinggi baik dalam proses bisnis, kurikulum, pembelajaran maupun riset [2]. Dalam artikel [2] dijelaskan bahwa ICT pendidikan tinggi semestinya harus berciri 5.0 di saat industri baru memasuki 4.0, hal ini membuktikan bahwa inovasi pengelolaan ICT pendidikan tinggi sangat penting, karena menjadi tonggak kemajuan teknologi perindustrian.

Banyak pendidikan tinggi di Indonesia yang telah memanfaatkan teknologi informasi dalam pengelolaan kegiatan maupun proses administrasi akademik, keuangan dan kepegawaian [3]. Selain itu banyak negara maju maupun berkembang yang juga mengadopsi teknologi untuk membantu pembelajaran di ICT pendidikan tinggi masing-masing, misalnya Inggris, Amerika serikat, Jerman, India dan Pakistan [4]–[6]. Nugroho [3] menyebutkan bahwa pengolahan ICT di Indonesia masih kurang optimal, baik dalam hal teknis, infrastruktur serta konten yang dikembangkan. Kurangnya ruang untuk memenuhi permintaan yang tinggi, sumber daya terbatas, biaya administrasi tinggi dan kesulitan mengelola populasi pengguna yang semakin membesar adalah tantangan pengelolaan ICT di pendidikan tinggi [7].

Salah satu upaya yang telah dilakukan beberapa pendidikan tinggi international adalah dengan menggunakan cloud computing sebagai infrastrukturnya, contohnya Universitas Harvard University dan University of Toronto [4], [5]. Namun, Perkembangan teknologi *cloud* dan virtualisasi yang sangat cepat membuat pengelolaannya menjadi sulit, sehingga perlu teknologi manajemen infrastruktur IT baru yang dapat menanganinya [8]. Teknologi *Infrastructure as code* menjadi jawaban dalam masalah ini [8], Konsep ini juga membawa budaya baru dalam penerapannya yaitu DevOps [9].

Penelitian ini akan membahas seberapa pentingnya ICT di pendidikan tinggi, dan teknologi infrastruktur serta arsitektur baru yang dapat diadopsi untuk menangani masalah disrupsi pengelolaan ICT pendidikan tinggi. Selain itu, budaya baru yang dapat diadaptasi oleh lembaga pengelolaan IT membawa dampak positif bagi pengelolaan ICT terutama di pendidikan tinggi. Penelitian ini diakhiri dengan serangkaian rekomendasi dan rencana untuk pekerjaan di masa depan.

II. PENTINGNYA ICT BAGI PENDIDIKAN TINGGI

Dunia digital dan informasi bergerak semakin cepat, peran ICT dalam pendidikan menjadi lebih penting dan semakin berkembang di abad ke-21 [4]. Dalam penelitian Sarkar [4] juga dijelaskan seberapa pentingnya perkembangan TIK di abad ke 21. ICT mengubah banyak aspek di kehidupan, begitu juga peran ICT dalam pendidikan semakin berkembang. Dalam berbagai penelitian dijelaskan peran pendidikan tinggi dalam konteks ekonomi berbasis pengetahuan dan *globalisasi* adalah untuk memberikan individu kemampuan untuk mengubah informasi menjadi pengetahuan yang bermanfaat secara sosial, memodernisasi masyarakat dan meningkatkan taraf hidup [10], [11]. Sehingga, ICT dianggap menjadi garis kehidupan untuk pertumbuhan abad ke-21, seperti yang terjadi di India [4] dan Pakistan [5].

Meningkatkan kualitas pendidikan dan pengajaran merupakan masalah penting, terutama pada saat penyebaran pendidikan di negara berkembang [4]. Dua puluh negara telah mengadopsi "*The constitution of the United Nations*

Educational (UNESCO)" dalam penerapan ICT mereka. Secara garis besar prinsip-prinsip UNESCO pada ICT dalam pendidikan dapat diringkas sebagai berikut ini [12]:

- Teknologi lama dan baru perlu digunakan secara seimbang.
- Memenuhi tujuan pendidikan internasional pada tahun 2015, yang membutuhkan investasi besar di lembaga pelatihan guru.
- Permintaan untuk akses pendidikan di negara maju dan berkembang tidak dapat dipenuhi, tanpa memangkas jarak dengan mode virtual pembelajaran (*universitas* virtual).
- Tujuan pendidikan tidak dapat dipenuhi tanpa adanya isu gender. Jika memungkinkan, diusulkan indikator yang akan mengatasi kebutuhan untuk mengukur gap gender.

ICT sangat berpotensi untuk memperluas kesempatan pendidikan baik formal maupun non-formal, meningkatkan pembelajaran dan penelitian baik dari konstruktivis dan instruktivis teori-teori belajar. Secara umum manfaat ICT di dunia pendidikan sangat banyak, dalam penelitian [4] disebutkan bahwa mengadopsi teknologi dapat meningkatkan akses ke pendidikan. Pengaksesan tidak terbatas oleh ruang, waktu dan konten, hal ini berkaitan dengan teknologi virtual *learning* yang digunakan seperti *Hughes Net Global Educations Interaktif platfrorm* yang berusaha mencirikan pendidikan di masa depan sebagai pendidikan *Real Time Interaktif* [4]. Pemanfaatan teknologi juga berpengaruh dalam sistem pengajaran *offline*. Jumlah pertumbuhan masa pendidikan yang tinggi membuat kelas semakin besar dan sulit untuk siswa mendapatkan pengetahuan jika guru hanya mengandalkan *audio-visual* sebagai media pengantar pembelajaran. Tidak dipungkiri bahwa penggunaan teknologi dapat meningkatkan kognitif siswa dan prestasi, jika digunakan secara tepat dan sesuai [4].

Dalam studi [11], [13] menyatakan bahwa ICT tidak hanya membantu pendidikan tinggi, namun juga mempersempit kesenjangan digital dan membantu meningkatkan kualitas pembelajaran dan hasil pendidikan. ICT mempengaruhi banyak aspek dalam kehidupan tidak hanya pendidikan, melainkan pembangunan, pekerjaan, pertumbuhan ekonomi, administrasi, pengurangan kemiskinan, penelitian dan *globalisasi* [11], [14].

Efektivitas, biaya, ekuitas dan keberlanjutan adalah empat isu yang harus ditangani ketika mempertimbangkan dampak keseluruhan dari penggunaan ICT dalam pendidikan. Dalam penelitian [4] disebutkan permasalahan dalam pengadopsian ICT dalam dunia pendidikan sangat beragam misalnya, membutuhkan biaya yang tinggi dalam pemasangan, pengoperasian dan pemeliharaan ICT. Banyak negara berkembang yang ketersediaan listrik dan jaringan telepon belum tersedia, sehingga penerapan ICT yang baik masih terganggu. Sarkar [4] menyatakan bahwa kurangnya fasilitas ICT dan infrastruktur adalah hambatan yang signifikan untuk penggunaan ICT. Dapat disimpulkan bahwa infrastruktur ICT yang kuat di pendidikan tinggi adalah hal kritis yang harus diperhatikan dan prasyarat bagi pembangunan berbasis pengetahuan.

III. TEKNOLOGI ZAMAN MODERN

A. Tantangan Infrastruktur IT di Masa Kini

Pada tahun 2008 Daryl Plummer dan Homs Bittman menyatakan bahwa "tahun 2012, 80% dari 1000 perusahaan akan menggunakan beberapa layanan *cloud computing*, dan 30% dari mereka akan membayar untuk infrastruktur *cloud computing*" [6]. Prediksi ahli dari IDC, Forrester, The Yankee Group dan RedMonk semua mendukung pernyataan tersebut [6]. Perkembangan *cloud computing* ini, tidak terlepas dari perkembangan internet yang menjadi tonggak awal kemunculannya.

Banyaknya kalangan industri yang mengadopsi *cloud computing* untuk infrastruktur mereka, membuat banyak lembaga pendidikan memutuskan untuk pindah sebagian atau seluruh infrastruktur mereka ke *cloud* seperti yang dilakukan Amerika Serikat [6], [15]. Hal ini kemudian diikuti oleh lembaga pendidikan tinggi di Jerman pada awal tahun 2011 [6]. Pengadopsian infrastruktur *cloud computing* ini juga dilakukan oleh pendidikan tinggi di negara berkembang, seperti National University of Rwanda dan Kigali Institute of Education di Rwanda, Universitas Nairobi dan Kenya Methodist University di Kenya dan Universitas Mauritius [16], [17].

Cloud computing adalah sebuah abstraksi yang didasarkan pada gagasan penyatuan sumber daya fisik dan menjadikan sumber daya virtual [6]. Sedangkan menurut Ercan [18] *cloud computing* adalah gaya komputasi dalam skala besar yang mampu menyediakan layanan untuk struktur eksternal menggunakan teknologi internet. *Cloud computing* didasarkan pada konvergensi virtualisasi, komputasi *utilitas* dan layanan perangkat lunak yang dapat diakses melalui internet [15]. National Institute of Standards and Technology (NIST) membagi tiga model layanan pada teknologi *cloud computing*, yaitu: *Sofware as a Service* (SaaS), *Platform as a Service* (PaaS) dan *Infrastructure as a Service* (IaaS) [19].

Penerapan *cloud computing* tidak selamanya berjalan lancar, terdapat beberapa tantangan dan permasalahan yang harus diperhatikan. Tantangan *cloud computing* di pendidikan tinggi, antara lain: sulitnya mengontrol layanan IT untuk organisasi, isu keamanan atas data dan aplikasi, masalah hukum mengenai yurisdiksi kontrak jika layanan berada di luar negeri, masalah perjanjian tingkat layanan, kecepatan dan infrastruktur internet juga dapat mempengaruhi layanan, dukungan organisasi dan beberapa aplikasi mungkin tidak berjalan pada infrastruktur *cloud* dan isu-isu hak kekayaan intelektual [18], [20], [21].

Permasalahan yang harus diperhatikan adalah bagaimana mengontrol infrastruktur *cloud computing*, tidak dapat dipungkiri perkembangan teknologi *cloud* dan virtualisasi semakin cepat dan pengelolaannya semakin sulit dikendalikan. Teknologi *Infrastructure as Code* kemudian muncul untuk mengatasi permasalahan pengelolaan infrastruktur *cloud* ini [8]. Kemudian semakin berkembang dan muncul teknologi otomasi yang juga dapat diadopsi dengan menerapkan *Infrastructure as Code*.

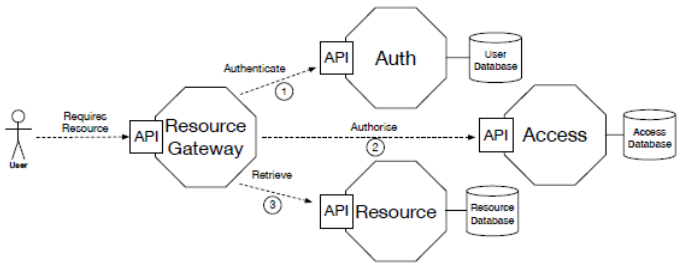
B. Kepunahan Arsitektur Tradisional

Infrastruktur *cloud* semakin populer digunakan di industri maupun pendidikan tinggi [6], [15]. Sebagian besar perusahaan

yang menggunakan sistem *cloud* harus mampu berinovasi dengan cepat terhadap produk aplikasi yang mereka kembangkan. Ini adalah alasan mengapa *continuous delivery* menjadi populer, terutama di startup, perusahaan internet besar dan penyedia layanan SaaS. Pengembangan produk aplikasi yang menggunakan *cloud* sangat terbatas dan sulit, jika masih menggunakan arsitektur tradisional atau *monolith* [22].

Monolith adalah aplikasi perangkat lunak yang terdiri dari modul yang tidak independen[23]. Arsitektur ini sulit untuk mendistribusikan tanpa menggunakan kerangka tertentu atau ad hoc seperti, Network Objects, RMI atau CORBA [23]. Selain itu masih banyak hambatan dan masalah dalam pengembangan arsitektur ini, yang di paparkan dalam beberapa paper, antara lain: Dragoni [23] mengatakan *monolith* berukuran besar sehingga sulit untuk dikembangkan dan sangat kompleks serta membatasi *scalability*. Pada paper [24] menambahkan *monolith* sering dikaitkan dengan "*dependency hell*", karena sistem aplikasi yang tidak konsisten dan terkompilasi secara penuh ketika ditambahkan atau diperbarui perpustakaannya. Dragoni [23] juga menyatakan setiap perubahan satu modul dari arsitektur *monolith* membutuhkan *restart* seluruh aplikasi, dan *restart* ini biasanya memerlukan *downtime* yang lama.

Arsitektur *microservices* diusulkan untuk menggantikan arsitektur *monolith* dan mengatasi setiap masalahnya [25]. Menurut Dragoni [23] arsitektur *microservices* adalah aplikasi terdistribusi yang semua modul-modulnya independen minimal dan berinteraksi melalui pesan . Istilah "*microservices*" pertama kali diperkenalkan tahun 2011 pada lokakarya arsitektur, sebagai cara untuk menggambarkan ide-ide umum peserta dalam pola arsitektur perangkat lunak [25]. Prinsip arsitektur *microservices* membantu manajer proyek dan pengembang untuk menyediakan pedoman atau desain dan implementasi aplikasi terdistribusi. Gambaran arsitektur *microservices* dapat dilihat di Gambar 1:



Gambar 1. Arsitektur *microservices*

Banyak penelitian yang mendukung arsitektur *microservices* dapat mengatasi masalah pada arsitektur sebelumnya, antara lain: Fowler [26] menyatakan *microservices* mendorong *continuous integration* dan sangat memudahkan perawatan perangkat lunak. Pernyataan itu didukung juga oleh pendapat Dragoni [23] yang mengatakan *microservices* menerapkan jumlah fungsionalitas terbatas yang membuat kode dasar menjadi kecil dan inheren sehingga membatasi ruang lingkup *bug*. Dalam penelitian Merkel (2014) menyebutkan *microservices* dapat diterapkan untuk *containerization*. Dragoni [23] juga menambahkan *microservices* tidak memerlukan *restart* lengkap dari seluruh sistem, tetapi hanya sebagian objek *microservices*, sehingga meminimalkan waktu *downtime*.

Microservices sekarang sudah menjadi tren baru [23], banyak perusahaan yang sudah mengadopsi arsitektur ini, seperti Amazon [27], Netflix [28], Gilt [29], Linkedln [30] dan SoundCloud [31] untuk mendukung *scalability* aplikasi dan produk mereka. Berdasarkan studi kasus yang dilakukan dalam penelitian Villamizar [22] ini, menyebutkan *microservices* lebih cocok digunakan pada aplikasi berskala besar dengan pengguna antara ratusan ribu atau jutaan. Adopsi *microservices* harus dilaksanakan sebagai strategi bisnis jangka panjang dan tidak dipandang hanya sebuah proyek saja, karena adopsi mereka memerlukan upaya dan kemampuan yang harus dikembangkan secara bertahap. Penggunaan *microservices* dapat mengurangi biaya infrastruktur sebesar 70% atau lebih sehingga sangat menghemat anggaran [32].

IV. DevOPS: Budaya Baru yang Membawa Kebaikan

Organisasi tradisional sering melakukan pengembangan aplikasi yang dilakukan oleh tim *development* dan tim *operations*, yang masing-masing memiliki tujuan tersendiri [33]. Dalam situasi seperti itu, *development* bertugas menciptakan fungsi baru dan memperbaiki *bug*, sementara *operations* membuat infrastruktur yang handal yang memungkinkan perangkat lunak dapat berjalan dengan stabil. Dapat disimpulkan, peran *development* dan *operations* sangat bertentangan, *development* berusaha untuk perubahan sedangkan *operations* berusaha untuk kestabilan. Hal ini menjadi masalah ketika merilis perangkat lunak untuk waktu yang cepat dan tepat, karena keduanya sangat bergantung satu sama lain. Oleh karena itu, budaya tradisional dinilai kurang efektif untuk pengembangan aplikasi di masa sekarang.

Berkembangnya pengadopsian arsitektur *microservices* di banyak perusahaan, juga mendorong perubahan budaya pengembangan perangkat lunak yang diterapkan. Arsitektur *monolith* membatasi frekuensi rilis [34] tetapi arsitektur microservices tidak membatasi dan dapat terus dikembangkan [35]. Menurut penelitian [36] "*Deliver Software Faster*", kerjasama antara tim IT untuk pengembangan dan pemeliharaan perangkat lunak sangat dibutuhkan, contohnya dengan mengadopsi DevOps. Budaya DevOps mengaburkan batas antara peran *development* dan *operations* dan akhirnya dapat menghilangkan perbedaan [37]. Gambaran budaya kolaborasi DevOps dapat dilihat di Gambar 2 [38]. Sikap tanggung jawab bersama adalah aspek budaya DevOps yang mendorong kolaborasi yang lebih erat. Beberapa pergeseran organisasi diperlukan untuk mendukung budaya tanggung jawab bersama, Sehingga tidak ada silo atau batas antara *development* dan *operations.* Pergeseran organisasi dilakukan untuk mendukung tim otonomi, dimana dalam mengambil keputusan tidak berbelit-belit. Untuk memastikan perubahan dalam produksi tim perlu menilai *building quality* dalam proses pengembangan. Penting juga bagi tim untuk menilai umpan balik, landasan gerakan DevOps adalah otomasi.

Gambar 2. Kolaborasi DevOps

Pengadopsian DevOps di beberapa perusahaan maupun organisasi menunjukkan manfaat bagi pengembangan produk aplikasi mereka, seperti yang ditunjukkan pada penelitian Ellen [35]. Devops mendukung kecepatan produksi perangkat lunak dan otomasi mengurangi usaha yang dibutuhkan ketika menyiapkan perilisan, sehingga memungkinkan organisasi untuk rilis lebih sering sesuai keperluan. Pada penelitian yang dilakukan Ellen [35] penerapan DevOps dinilai menghemat penggunaan sumber daya untuk pengembangan dan pemeliharaan, karena adanya otomatisasi, salah satunya *continuous delivery*. DevOps juga berdampak pada karyawan, dimana pada penelitian Ellen [35] mengungkapkan sering rilis membantu mengurangi stress karena kecemasan sehingga meningkatkan kesejahteraan. Hal itu membuktikan DevOps tidak hanya membawa manfaat untuk perusahaan, namun juga bagi pekerja.

Pengadopsian DevOps tidak mudah dan memerlukan perubahan yang kompleks sehingga harus memiliki strategi yang mendukung. Menurut Hamunen [39] tantangan ketika mengadopsi DevOps secara garis besar dikelompokkan menjadi empat yaitu: kurangnya kesadaran, kurangnya dukungan, masalah dengan implementasi teknologi DevOps dan masalah dengan proses pengadaptasian DevOps untuk organisasi. Sedangkan, pada penelitian Ellen [35] menambahkan tantangan pengadopsian DevOps adalah budaya organisasi yang tidak mudah dibentuk, kurangnya komunikasi dan rasa tanggung jawab. Sehingga sangat perlu diperhatikan tantangan-tantangan tersebut untuk penerapan DevOps yang baik dan tepat.

V. Disrupsi Pengelolaan Kultur Pengelolaan ICT: Refactoring Arsitektur Modern

Migrasi arsitektur *monolith* ke *microservices* membawa banyak mafaat seperti fleksibilitas untuk beradaptasi dengan perubahan teknologi dan manajemen sumber daya tunggal untuk komponen sistem yang berbeda. DevOps dapat digunakan dalam arsitektur *monolith*, tapi arsitektur *microservices* lebih baik, karena memungkinkan pelaksanaan yang lebih efektif dari DevOps melalui tim kecil [40]. Dalam melakukan proses migrasi arsitektur, harus dilakukan terlebih dahulu mengidentifikasi arsitektur sebelumnya dan arsitektur yang direncanakan kedepannya. Secara garis besar Tabel I memuat ringkasan pola migrasi arsitektur *microservices* yang terdapat pada artikel [40].

Tabel I. Pola migrasi arsitektur *microservices*

Pola	Dampak ke DevOps
- Aktifkan *Continuous Integration* (CI)	CI adalah langkah pertama menuju *continuous delivery* (CD), praktik DevOps.
- Memulihkan Arsitektur Saat Ini - Mengurai Monolith - Mengurai Monolith Berdasarkan Kepemilikan Data - Ubah Ketergantungan Kode ke *Service Call*	Pola-pola ini memungkinkan penguraian sistem ke dalam layanan yang lebih kecil, mengarah ke tim yang lebih kecil.
- Memperkenalkan *Service Discovery* - Memperkenalkan *Service Discovery Client* - Memperkenalkan *Internal Load Balancer* - Memperkenalkan *External Load Balancer*	Memperkenalkan Penemuan *Dynamic Service Discovery* menghilangkan kebutuhan untuk pengkabelan manual, dengan demikian meningkatkan penyebaran *pipeline* yang lebih independen.
- Memperkenalkan *Circuit Breaker* - Memperkenalkan Server Konfigurasi - Memisahkan konfigurasi dari kode adalah praktik terbaik CD.	Gagal cepat dapat mengurangi kopling antar layanan, sehingga berkontribusi dalam penyebaran layanan independen.
- Memperkenalkan *Edge Server*	*Edge Server* tidak hanya memungkinkan tim pengembangan untuk lebih mudah mengubah struktur internal sistem tetapi juga memungkinkan tim operasi untuk memantau lebih baik status keseluruhan setiap layanan.
- *Containerize the Services*	Wadah dapat menghasilkan lingkungan yang sama dalam produksi dan pengembangan, sehingga mengurangi konflik antara tim pengembangan dan operasi.
- Menerapkan *Cluster* dan *Orchestrate Containers*	Alat manajemen *cluster* mengurangi kesulitan penyebaran banyak orang contoh dari berbagai layanan dalam produksi, sehingga mengurangi tim operasi, resistensi terhadap perubahan tim pengembangan.
- Pantau Sistem dan Berikan Masukan	Pemantauan kinerja memungkinkan pengumpulan data kinerja dan berbagi untuk meningkatkan pengambilan keputusan. Misalnya, tim pengembangan dapat menggunakan informasi tersebut untuk refactor arsitektur jika menemukan anomali kinerja di sistem.

Pada artikel [41] juga menjelaskan pola proses migrasi ke arsitektur *microservices* dari arsitektur SSaaS (*Server Side as a Service*) yang mengambil beberapa pola dalam Tabel 1, seperti: Mempersiapkan CI *Pipeline*, Transformasi developer data ke service, Memperkenalkan CD, Memperkenalkan Server Edge,

Memperkenalkan *Service Discovery,* Memperkenalkan resource manager, Memperkenalkan *Chat Services* dan *Developer Info Services,* dan Clusterization. Pola pada artikel [41] juga diterapkan dalam artikel [40] untuk migrasi *Backtory*. Pola migrasi arsitektur dapat disesuaikan dengan keperluan dimasa depan.

VI. KESIMPULAN

Isu disrupsi pada pengelolaan ICT menjadi penting untuk dipikirkan dalam manajemen pendidikan tinggi. Bagaimana pendidikan tinggi dituntut untuk selalu inovatif dan tidak tertinggal dengan kemajuan teknologi di industri maupun pendidikan tinggi yang lain. Teknologi DevOps menjadi jawaban untuk tantangan ini, dengan mengubah pola pikir untuk lebih inovatif sehingga bergerak lebih cepat. DevOps bukan hanya tentang infrastruktur dan arsitektur teknologi yang berubah namun juga budaya tim yang lebih baik untuk kebaikan bersama. Proses migrasi arsitektur microservices dapat mengikuti pola yang disesuaikan dengan kebutuhan di masa depan, proses migrasi yang sangat penting dilakukan adalah menerapkan otomasi untuk *continuous delivery* dan *continuous integration* menjadi *continuous development*, mentransformasi developer data ke *service*, menerapkan server Edge, memperkenalkan *Service Discovery* dan *Resource Manager* serta *Clusterization*. Sehingga di masa depan perguruan tinggi menjadi panutan pengembangan teknologi di industri.

REFERENSI

[1] "What disruption really means." [Online]. Available: https://www.tonyrobbins.com/career-business/what-disruption-really-means/. [Accessed: 31-May-2018].

[2] A. D. Lestari, "Kampus Disruptif – PUSAT," *2018*, 2018. [Online]. Available: http://aptisi.or.id/2018/04/21/kampus-disruptif/. [Accessed: 30-May-2018].

[3] L. E. Nugroho, "Kerangka Pengembangan Pendidikan tinggi di Indonesia," pp. 112–113, 2013.

[4] S. Sarkar, "The Role of Information and Communication Technology (ICT) in Higher Education for the 21st Century," *Sci. Probe*, vol. 1, no. 1, pp. 30–41, 2012.

[5] Z. A. Shaik and S. A. Khoja, "The role of ICT in shaping the future of Pakistan higher education system.," *turkish Online J. Educ. Technol.*, vol. 10, no. 1, p. 2011, 2011.

[6] A. Jenhani, "Cloud Computing in German Higher Education Institutions," 2011.

[7] A. Ya 'u Gital and F. U. Zambuk, "Cloud Computing: Solution to ICT in Higher Education in Nigeria," *Adv. Appl. Sci. Res.*, vol. 2, no. 6, pp. 364–369, 2011.

[8] K. Morris, *Infrastructure as Code*. United States of America: O'Reilly Media, Inc., 2016.

[9] C. Null, "Infrastructure as code is essential to your DevOps practices," 2015. [Online]. Available: https://techbeacon.com/infrastructure-code-engine-heart-devops. [Accessed: 19-Mar-2018].

[10] S. C. Kong and K. M. Li, "Collaboration between school and parents to foster information literacy: Learning in the information society," *Comput. Educ.*, vol. 52, no. 2, pp. 275–282, 2009.

[11] A. Shaikh, Zaffar, "Usage, Acceptance, Adoption, and Diffusion of Information & Communication Technologies in Higher Education: A Measurement of Critical Factors," vol. 9, no. 2, pp. 63–80, 2009.

[12] Unesco, *Guide To Measuring Information And Communication Technologies (ICT) In Education*, no. 2. 2009.

[13] J. W. Richardson, "ICT in Education Reform in Cambodia: Problems, Politics, and Policies Impacting Implementation," *Inf. Technol. Int. Dev.*, vol. 4, no. 4, pp. 67–82, 2008.

[14] A. Aypay, "Informatıon and Communıcatıon Technology (ICT) Usage and Achıvement of Turkısh Students in PISA 2006," *TOJET Turkish Online J. Educ. Technol.*, vol. 9, no. 2, pp. 116–124, 2010.

[15] F. Makoza, "Cloud computing adoption in Higher Education Institutions of Malawi: An exploratory study," *J. Comput. ICT Res.*, vol. 9, no. 2, pp. 37–54, 2015.

[16] N. Sultan, "Cloud computing for education: A new dawn?," *Int. J. Inf. Manage.*, vol. 30, no. 2, pp. 109–116, 2010.

[17] T. Kihara and D. Gichoya, "Use of cloud computing platform for e-learning in institutions of higher learning in Kenya," *2014 IST-Africa Conf. Proc.*, pp. 1–6, 2014.

[18] T. Ercan, "Effective use of cloud computing in educational institutions," *Procedia - Soc. Behav. Sci.*, vol. 2, no. 2, pp. 938–942, 2010.

[19] P. Mell and T. Grance, "The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology," *Nist Spec. Publ.*, vol. 145, p. 7, 2011.

[20] S. Okai, M. Uddin, A. Arshad, R. Alsaqour, and A. Shah, "Cloud computing adoption model for universities to increase ICT proficiency," *SAGE Open*, vol. 4, no. 3, pp. 1–10, 2014.

[21] C. Low, Y. Chen, and M. Wu, "Understanding the determinants of cloud computing adoption," *Ind. Manag. Data Syst.*, vol. 111, no. 7, pp. 1006–1023, 2011.

[22] M. Villamizar, O. Garcés, H. Castro, M. Verano, L. Salamanca, and S. Gil, "Evaluating the Monolithic and the Microservice Architecture Pattern to Deploy Web Applications in the Cloud Evaluando el Patrón de Arquitectura Monolítica y de Micro Servicios Para Desplegar Aplicaciones en la Nube," *10th Comput. Colomb. Conf.*, pp. 583–590, 2015.

[23] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, "Microservices: yesterday, today, and tomorrow," no. June, 2016.

[24] N. Kratzke, "A Lightweight Virtualization Cluster Reference Architecture Derived from Open Source PaaS Platforms," *Open J. Mob. Comput. Cloud Comput.*, vol. 1, no. 2, pp. 17–30, 2014.

[25] J. Lewis and M. Fowler, "Microservices," *2014*, 2014. [Online]. Available: https://martinfowler.com/articles/microservices.html. [Accessed: 08-May-2018].

[26] M. Fowler, M. Y. B. Self-testing, I. Machine, A. Deployment, I. C. Integration, F. Thoughts, and F. Reading, "Continuous Integration," pp. 1–13, 2014.

[27] S. Kramer, "Gigaom | The Biggest Thing Amazon Got Right: The Platform," *October*, 2011. [Online]. Available: https://gigaom.com/2011/10/12/419-the-biggest-thing-amazon-got-right-the-platform/. [Accessed: 08-May-2018].

[28] T. Mauro, "Microservices at Netflix: Lessons for Architectural Design," *February*, 2015. [Online]. Available: https://www.nginx.com/blog/microservices-at-netflix-architectural-best-practices/. [Accessed: 08-May-2018].

[29] Y. Goldberg, "Scaling Gilt: from Monolithic Ruby Application to Distributed Scala Micro-Services Architecture," *Oktober*, 2014. [Online]. Available: https://www.infoq.com/presentations/scale-gilt. [Accessed: 08-May-2018].

[30] S. Ihde and K. Parikh, "From a Monolith to Microservices + REST: the Evolution of LinkedIn's Service Architecture," *March*, 2015. [Online]. Available: https://www.infoq.com/presentations/linkedin-microservices-urn. [Accessed: 08-May-2018].

[31] P. Calcado, "Backstage Blog - Building Products at SoundCloud —Part I: Dealing with the Monolith - SoundCloud Developers," *Juni*, 2014. [Online]. Available: https://developers.soundcloud.com/blog/building-products-at-soundcloud-part-1-dealing-with-the-monolith. [Accessed: 08-May-2018].

[32] M. Villamizar, O. Garces, L. Ochoa, H. Castro, L. Salamanca, M. Verano, R. Casallas, S. Gil, C. Valencia, A. Zambrano, and M. Lang, "Infrastructure Cost Comparison of Running Web Applications in the Cloud Using AWS Lambda and Monolithic and Microservice Architectures," *Proc. - 2016 16th IEEE/ACM Int. Symp. Clust. Cloud, Grid Comput. CCGrid 2016*, pp. 179–182, 2016.

[33] R. de Feijter, “Towards the adoption of DevOps in software product organizations: A Maturity Model Approach,” no. May, pp. 1–173, 2017.

[34] A. Balalaie, A. Heydarnoori, and P. Jamshidi, “Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture,” *IEEE Softw.*, vol. 33, no. 3, pp. 42–52, May 2016.

[35] L. Ellen, L. Riungu-kalliosaari, S. Mäkinen, and L. E. Lwakatare, “DevOps Adoption Benefits and Challenges in Practice : A Case Study,” pp. 590–597, 2016.

[36] CaTechnology, “TechInsights Report : What Smart Businesses Know About DevOps,” no. September, p. 300, 2013.

[37] R. Wilsenach, “DevOpsCulture,” 2015. [Online]. Available: https://martinfowler.com/bliki/DevOpsCulture.html#footnote-tools. [Accessed: 15-Apr-2018].

[38] R. Wilsenach, “DevOpsCulture.” [Online]. Available: https://martinfowler.com/bliki/DevOpsCulture.html. [Accessed: 09-Jul-2018].

[39] J. Hamunen, “Challenges in adopting a Devops approach to software development and operations,” May 2016.

[40] A. Balalaie, A. Heydarnoori, and P. Jamshidi, “Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture,” *IEEE Softw.*, vol. 33, no. 3, pp. 42–52, 2016.

[41] A. Balalaie, A. Heydarnoori, and P. Jamshidi, “Migrating to Cloud-Native architectures using microservices: An experience report,” *Commun. Comput. Inf. Sci.*, vol. 567, pp. 201–215, 2016.