



## **ANALISA IMPLEMENTASI ARSITEKTUR MICROSERVICES BERBASIS KONTAINER PADA KOMUNITAS PENGEMBANG PERANGKAT LUNAK SUMBER TERBUKA (OPENDAYLIGHT DEVOPS COMMUNITY)**

**Rahmad Ade Putra**

Universitas Bina Nusantara, Magister Manajemen Sistem Informasi  
Jakarta, Indonesia

email : [rahmad.putra@binus.ac.id](mailto:rahmad.putra@binus.ac.id)

### **Abstrak**

Pada penelitian yang ditulis dalam Tesis ini, bertujuan untuk dapat meneliti dari faktor keberhasilan terhadap implementasi arsitektur microservices kontainer pada komunitas pengembangan perangkat lunak sumber terbuka Opendaylight project. Faktor-faktor tersebut adalah source code commit, merge source code, installed dependency, connection latency to master node, construction Time. Hasil dari uji test pada data yang terpublish untuk impelemtasi arsitektur microservices menunjukkan bahwa dengan menggunakan model regresi dapat menemukan tingkat pengaruh dari variabel yang ada saat proses implementasi berlangsung. Faktor yang paling mempengaruhi dari tingkat keberhasilan implementasi menurut uji coba yang dilakukan adalah merge source code, installed dependency, connection latency dan constrction time.

**Kata kunci** : Microservices Arsitektur, Opendaylight, Kontainer

### **PENDAHULUAN**

Salah satu topik terhangat Teknologi Informasi dan Komunikasi saat ini adalah cloud computing atau yang biasa disebut dengan komputasi awan. Sebagai satu trend dan buzzword pada teknologi informasi dan telekomunikasi yang masih terus dikembangkan oleh para praktisi ICT sampai saat ini. Teknologi cloud computing dihadirkan sebagai bentuk usaha untuk memungkinkan akses sumber daya dan aplikasi dari mana saja melalui jaringan komputer baik internet ataupun intranet, sehingga keterbatasan dan kekurangan pemanfaatan infrastruktur ICT yang sebelumnya ada dapat diatasi. Pada penelitian ini akan dibahas mengenai analisa dan pengkajian terhadap salah satu area yang mencakup teknologi cloud computing yaitu private cloud computing berbasis kontainer. Teknologi kontainer merupakan suatu konsep teknologi dimana dapat mengisolasi sebuah proses dari proses yang lainnya dengan

memecahnya kedalam library dan aplikasi dependensi atau paket-paket software penunjang yang digunakan pada sistem operasi tanpa menggunakan seluruh rangkaian sistem operasi, atau dengan kata lain shared sistem operasi. Perangkat lunak utama pada teknologi kontainer biasanya menggunakan Docker yang terdapat pada sistem operasi Linux dan menggunakan Kubernetes sebagai orkestrasi atau kontainer manajemen yang memungkinkan untuk men-deploy atau menerapkan project aplikasi software kedalam berbagai lingkungan siolasi virtual yang terisolasi pada level sistem operasi. OpenDaylight adalah sebuah pengembangan proyek open source kolaboratif di Linux Foundation dengan berlisensi Apache 2.0 yang mana bertujuan untuk mempercepat adopsi serta penerapan Software-Defined Networking (SDN) pada lingkungan enterprise. Komunitas pengembang Opendaylight menggunakan Bitergia Analytics untuk menampilkan seluruh

statistik proses pengembangan aplikasi dan proses pembuatan komponen-komponen *opendaylight* berdasarkan proses otomatisasi CI/CD pada Jenkins. Pada penulisan penelitian ini penulis akan mengambil data yang ada pada Jenkins yang mana data tersebut berupa statistik otomatisasi sumber kode yang dibagikan secara berkala dan berkelanjutan oleh para developer atau kontributor pengembang menjadi kontainer-kontainer *microservices*. Data yang digunakan sebagai sumber untuk analisa merupakan data yang ada dalam kurun waktu 1 tahun terhitung dari tanggal 1 Oktober 2017 hingga 1 Oktober 2018. Data tersebut berupa file CSV atau Comma-separated Values yang terbentuk dari data mining dalam statistik kontributor pembangunan komponen *Microservices*. Pada tingkat implementasi *microservices* dalam lingkungan pengembangan *Opendaylight* hanya sebatas *development environment*, sehingga belum adanya standar *de facto*, oleh karena itu diperlukan tahap analisa guna mengetahui faktor kelayakan akan arsitektur *microservices*. Faktor yang mempengaruhi tingkat keberhasilan dan juga faktor kegagalan didalam proses implementasi *microservices* pada lingkungan *development* dari *Opendaylight project* adalah Average Source Code Commit in Git yang merupakan jumlah rata-rata commit atau yang berarti perubahan sumber kode yang ada sebelum dikirim ke master repository pada Git. Average Merge Source Code merupakan jumlah rata-rata penggabungan antar branch sumber kode pada git untuk dijadikan stau kedalam sebuah image container yang berbentuk master branch. Average Installed Dependency Packages merupakan jumlah nilai rata-rata paket dependensi baik itu modul atau library yang ada pada kontainer *microservices* dan terenkapsulasi mejadi satu paket kontainer image yang melewati proses otomatisasi Jenkins. Average Connection Latency to Master Node merupakan jumlah rata-rata letency yang ada pada kontainer *Microservices* saat sedang dalam proses build, test dan deploy sehingga menghasilkan sebuah kesatuan paket kontainer yang diproses oleh Jenkins master server. Average Construction Time merupakan jumlah rata-rata waktu yang diperlukan saat proses cron atau otomatisasi berlangsung, proses tersebut berupa build, test dan deploy. Total source code on Container merupakan jumlah seluruh sumber kode aplikasi utama

yang ada dan diimplementasikan pada kontainer *microservices*, dalam hal ini total jumlah sumber kode dihitung berdasarkan intergrasi plugin git dengan Jenkins master node dalam kurun waktu 1 Oktober 2017 hingga 1 Oktober 2018.

## LANDASAN TEORI

### A. Arsitektur *Microservices*

*Microservice* adalah kumpulan proses independen dan kecil yang berkomunikasi antara satu dengan lainnya untuk membentuk aplikasi kompleks yang agnostik terhadap bahasa API apa pun. Servis-servis ini terdiri dari blok-blok kecil, terpisah, dan fokus pada tugas-tugas ringan untuk memfasilitasi metode modular dalam pembangunan sistem. *Microservice* sendiri merupakan pengembangan lanjutan dari *Service-oriented architecture (SOA)* karena *Microservice* merupakan sistem yang terdiri dari komponen-komponen berupa *services* yang modular, *autonomous* yang memiliki tujuan masing-masing namun ter-*orkestrasi* melalui protokol *light-weight* dengan satu sama lain untuk mencapai satu tujuan tertentu terutama di dalam pengembangan *software*.

### B. Jenkins

Jenkins adalah sebuah *open source automation server* untuk mengotomatiskan tugas-tugas di dalam proses *continuous integration and delivery* sebuah perangkat lunak. Jenkins merupakan aplikasi berbasis Java yang dapat dipasang dari repository *Ubuntu* atau dengan mengunduh dan menjalankan file *Web applicatino ARchive (WAR)*, sebuah koleksi file yang sudah lengkap dan tinggal dijalankan disebuah server.

### C. Jenkins Build

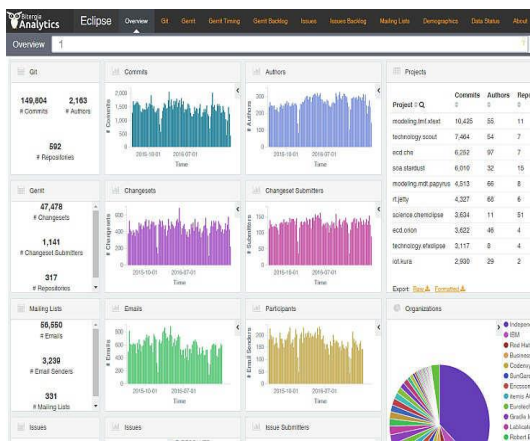
Build pada jenkins merupakan fitur yang dapat menjalankan dan membangun sebuah pekerjaan untuk otomatisasi sumber kode secara terkontrol dan termonitor untuk dijadikan *Docker image*, setiap aktifitas yang ada didalamnya. Salah satu contoh build atau pembangunan otomatisasi sumber kode yang akan dibangun menjadi kontainer image adalah kompilasi sumber kode, menjalankan test, implementasi paket modul dan library.

**D. Jenkins Pipeline**

Pipeline yang ada pada Jenkins merupakan rangkaian plugins yang memiliki dukungan terhadap pengembangan dan proses integrasi aplikasi dengan continuous delivery. Dalam hal ini Pipeline yang ada pada CI/CD merupakan proses otomatisasi dari proses untuk mendapatkan versi aplikasi yang dikembangkan berdasarkan manajemen sumber kode seperti Git atau Subversion.

**E. Bitergia Analytics**

Merupakan aplikasi berbasis web untuk melakukan analisa terhadap proses pengembangan perangkat lunak secara keseluruhan proyek. Pada Bitergia Analytics menyediakan data statistik berdasarkan data final dan insight untuk ditampilkan ke pengembang software yang berkolaborasi antara satu sama lainnya. Penggunaan Bitergia Analytics biasanya terdapat pada tingkat korporasi atau komunitas perangkat lunak terbuka yang ingin menampilkan secara terbuka data mengenai proses, reportase dan data analisis.



Gambar 1. Bitergia Analytics Dashboard

**F. Kibana**

Kibana adalah tool untuk visualisasi data pengembangan aplikasi dalam sebuah organisasi dengan berlisensi open source yang mana merupakan front-end untuk Elasticsearch. Kibana menyediakan antarmuka web dashboard yang menarik dan bersifat user-friendly. Kibana dapat digunakan untuk mengelola dan memvisualisasikan data dari Elasticsearch sehingga membantu para

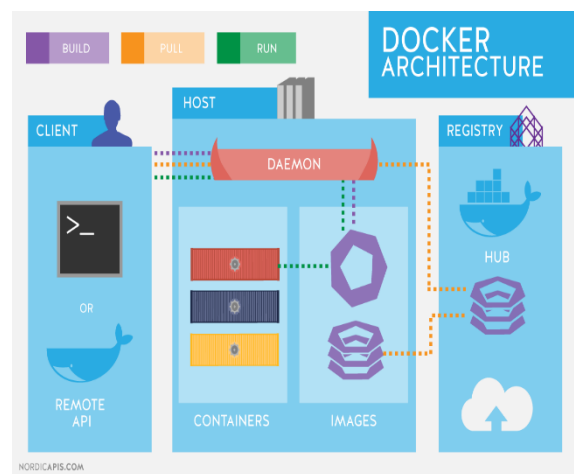
pengembang software atau DevOps untuk melakukan pelacakan terhadap masalah atau mencari log yang timbul saat proses uji coba aplikasi.

**G. OpenDaylight Project**

OpenDaylight Project adalah sebuah proyek open source kolaboratif di Linux Foundation yang bertujuan untuk mempercepat adopsi Software-Defined Networking (SDN) dan menciptakan pondasi yang solid untuk Network Function Virtualization (NFV) dengan pendekatan yang lebih transparan untuk mendorong inovasi baru dan mengurangi risiko. Linux Foundation Collaborative Projects sendiri merupakan proyek perangkat lunak didanai secara independen yang bertujuan untuk memanfaatkan kekuatan pengembangan kolaboratif untuk bahan bakar inovasi di industri dan ekosistem.

**H. Docker Container**

Docker adalah sebuah project open source yang ditujukan untuk software engineer, DevOps engineer atau sysadmin untuk membangun, mengemas dan menjalankan aplikasi dimana pun di dalam sebuah container. Docker berfungsi sebagai virtualisasi sebuah sistem operasi atau sebuah server atau sebuah web server atau bahkan sebuah database server, dimana dengan menggunakan virtualisasi ini, diharapkan developer dapat mengembangkan aplikasi sesuai dengan spesifikasi server.



Gambar 2. Docker Kontainer Arsitektur

### I. Git

Git adalah version control system yang digunakan para developer untuk mengembangkan software secara bersama-sama. Fungsi utama git yaitu mengatur versi dari source code program anda dengan mengasih tanda baris dan code mana yang ditambah atau dihapus atau bahkan dimodifikasi serta melakukan penggabungan antar branch atau proyek kode.

### J. JASP

JASP merupakan suatu program grafis dan open-source untuk analisis statistik yang dirancang mudah digunakan sama seperti SPSS. JASP banyak menyediakan metode statistik Bayesian dan dapat menghasilkan table serta plot yang mudah untuk dipahami. Perkembangan JASP didukung secara finansial oleh beberapa universitas dan dana penelitian. JASP menawarkan kesimpulan sesuai pada model statistik yang menggunakan nilai p dan interval kepercayaan untuk mengendalikan tingkat kesalahan dalam batas replikasi sempurna yang tak terbatas sehingga dapat memperkirakan nilai parameter yang kredibel dan bukti model yang diberikan data sesuai pengetahuan yang ada.

## METODOLOGI PENELITIAN

Dalam melakukan penelitian ini, penulis menggunakan pendekatan penelitian dengan menerapkan metode pendekatan deskriptif dan metode pendekatan verifikatif, karena adanya variabel-variabel yang akan olah keterkaitan dan hubungannya antar satu dan lainnya, serta tujuan untuk menyajikan gambaran secara terstruktur, dan akurat mengenai terhadap pengelompokan variable serta hubungan anatar variabel yang diteliti, yaitu pengaruh kesuksesan terhadap proses implementasi pada pengembangan ODL.

Tujuan dari penelitian deskriptif adalah untuk mengetahui dan menganalisis pengaruh dari variabel-variabel independen yang ada pada Bitergia Analytics Opendaylight terhadap faktor yang mempengaruhi tingkat keberhasilan dalam implementasi Opendaylight. Metode penelitian verifikatif pada dasarnya ingin menguji kebenaran dari suatu hipotesis yang dilaksanakan melalui pengumpulan data pada komunitas.

### K. Objek Penelitian

Objek penelitian merupakan acuan terhadap sumber data yang menjadi tolak ukur dan juga perhatian dalam suatu rangkaian penelitian, adapun objek penelitian menjadi sasaran dalam penelitian yaitu untuk mendapatkan jawaban atau solusi dari permasalahan yang sedang terjadi, yang mana solusi tersebut berguna kedepannya setelah penelitian selesai dilakukan dan dapat bermanfaat untuk dikembangkan.

### L. Jenis Penelitian

Penelitian ini adalah penelitian terapan yang bertujuan untuk menerapkan, menguji, dan mengevaluasi kemampuan analisa data yang diterapkan dalam memecahkan masalah-masalah praktis dengan pendekatan eksperimen terhadap komunitas proyek kolaborasi pengembangan perangkat lunak sumber terbuka. Pendekatan eksperimen dipilih untuk mencari pengaruh setiap variabel independen terhadap variabel dependen dalam kondisi terkendali secara ketat dan teliti. Terkait dengan tingkat eksplanasinya, penelitian ini menggunakan penelitian eksplanatif untuk menguji berbagai hipotesa tertentu dengan maksud membenarkan atau memperkuat hipotesa, mencari sebab musabab dari suatu gejala dan menentukan sifat dari hubungan antara satu atau lebih gejala atau variabel terikat dengan satu atau lebih variabel bebas. Disamping itu, penelitian ini menggunakan desain korelasional, yang mana dapat bertujuan untuk mengetahui hubungan variabel independen dengan variabel dependennya.

Sehingga akan dapat ditarik kesimpulan tentang validasi untuk penggunaan dan standarisasi arsitektur Microservices yang bertujuan untuk menyederhanakan konsumsi sumber daya pada lingkungan produksi pada enterprise sehingga dapat menghemat sumber daya.

### M. Pengumpulan Data dan Variable Penelitian

Variabel yang dijabarkan didalam penelitian ini didefinisikan secara jelas sehingga tidak menimbulkan pengertian ganda yang memiliki tujuan lain. Pengertian variabel itu sendiri merupakan konsep yang memiliki berbagai macam nilai.

Pada penelitian ini, penulis mengambil data yang berupa data kuantitatif dan kualitatif dalam bentuk golongan data sekunder yang berasal dari statistik otomatisasi continuous integration dan continuous delivery dalam arsitektur microservices berdasarkan Jenkins master node server pada komunitas pengembang Opendaylight. Data tersebut antara lain berupa : 1. Statistik Jenkins master server dalam proses build, test deploy untuk arsitektur microservices dengan periode dari 1 Oktober 2017 hingga 1 Oktober 2018, data yang didapat berupa comma-separated values (CSV) yang merupakan salah satu bagian dari database kontribusi pengembangan Opendaylight project; 2. Data statistik otomatisasi CI/CD pipeline atau build, test dan deploy berdasarkan Jenkins master server pada komunitas pengembang Opendaylight terdiri dari variable dependen Success Node Creation, Failure Node Creation dan Total Node Creation serta variable independen yang tidak saling terkait yaitu berupa Average Source Code Commit in Git, Average Merge Source Code, Average Installed Dependency Packages, Average Connection Latency to Master Node, Average Construction Time, dan Total source code on Container. Dalam dataset berbentuk file .CSV ini terdiri dari 80 nodes microsrvice yang terdapat pada tiap-tiap komponen atau sub-project dari Opendaylight, kesatuan dari kontainer ini membentuk interkoneksi antar satu dengan lainnya, sehingga dapat menjadi kesatuan aplikasi besar untuk Software Defined Network / Network Function Virtualization berlisensi open source dengan tujuan centralisasi manajemen jaringan komputer untuk skala enterprise maupun operator telekomunikasi. Proses penampilan atau visualisasi data dapat menggunakan Bitergia Analytics pada komunitas pengembang Opendaylight yang menampilkan keseluruhan proses otomatisasi CI/CD pipeline, data tersebut lalu dapat diunduh dengan file berformat .CSV untuk diolah sebagai bahan dasar analisa dan penkajian.

Tabel 1. Sample Populasi Data Penelitian

Node	Y	X1	X2	X3	X4	X5
------	---	----	----	----	----	----

	2,29	177.7	14.		4.6	12.
aaa	8	07	5	532	3	85
		142.5	19.			11.
alto	957	78	27	433	4.5	64
	1,17	407.5	8.9		1.4	5.6
ansible	9	72	5	673	1	4
archetyp		57.04	8.2		3.2	6.9
es	228	3	7	142	5	1
		27.94	8.9		8.9	0.7
atrium	1	2	2	1	2	7
autorelea		157.0	124		0.7	82.
se	923	52	.7	346	9	89
	7,46	252.6	41.	1,06	4.6	31.
bgpcep	9	45	24	2	2	54
	1,03	126.8	22.		4.6	13.
bier	3	67	84	394	6	52
	22,3	4,895.	5.4	11,7		3.6
builder	44	43	3	47	0	5
		118.2	12.		6.7	7.9
cardinal	377	85	94	315	5	4
		31.09	14.		14.	0.8
centinel	1	1	13	1	13	8
		84.23			3.4	7.4
coe	880	5	13	241	4	3
controlle	7,25	201.1	93.		4.4	25.
r	9	22	8	765	1	11
coretutor		28.32	63.		42.	
ials	15	8	41	10	19	1
	1,78	106.9	10.		3.5	9.4
daexim	3	44	43	393	9	1
		91.47	15.		9.3	1.2
didm	37	3	46	33	8	2
distributi	10,5	1,216.	23.	4,10		19.
on	91	93	02	9	0	35
		35.67	17.		5.9	11.
dlux	256	1	98	106	1	82

		174.6	19.		6.6	12.
dluxapps	385	52	56	272	8	26

Berdasarkan table diatas, penelitian ini menggunakan sample data statistik kontainer microservices yang cukup besar yaitu sebanyak 80 komponen nodes kontainer dan tidak dapat dimungkinkan untuk ditampilkan secara keseluruhan pada halaman lampiran atau penulisan ini. Data dimiminalisir sehingga hanya dapat ditampilkan sebanyak 20 sample atau dengan kata lain berada pada N1 hingga N-20, setelah itu tabel menuju baris sample ke 80 akan dimasukan kedalam table index pada akhir halaman. Dari keseluruhan data diatas berdasarkan statistik Jenkins master terdapat Microservice Container Images yang merupakan nama-nama kontainer microservices berupa sample populasi data dan juga data dependen atau (Y), sedangkan untuk faktor kesuksesan dalam implementasi CI/CD pipeline pada setiap sample kontainer microservices dituliskan dengan Success Node Creation sebagai bagian dari variable dependen (Y). Data sekunder tersebut merupakan hasil visualisasi dari Bitergia Analytics dengan menggunakan Kibana sebagai software development tracking progress atau aplikasi yang dapat memantau seluruh kolaborasi pengembangan software antar developer didalam organisasi atau komunitas, dalam hal ini adalah Opendaylight Project microservices kontainer tracking.

Berdasarkan variable dependen atau (Y) diatas, maka selanjutnya merupakan statistik data dengan variable Independen pada proses continuous integration dan continuous delivery pipeline. Average Source Code Commit atau Trigger on Commit pada master Jenkins server adalah bagian dari variable independen atau (X1) yang merupakan jumlah rata-rata commit atau yang berarti perubahan sumber kode yang ada sebelum dikirim ke master repository pada Git. Average Merge Source Code atau trigger on merge diklasifikasikan sebagai variable independen (X2) pada Jenkins server merupakan jumlah rata-rata penggabungan antar branch sumber kode pada git untuk dijadikan stau kedalam sebuah image container yang berbentuk master branch. Average Installed Dependency

Packages merupakan jumlah nilai rata-rata paket dependensi baik itu modul atau library yang ada pada kontainer microservices dan terenkapsulasi mejadi satu paket kontainer image yang melewati proses otomatisasi CI/CD pipleine dari Jenkins server Opendaylight project, variable ini dapat diklasifikasikan menjadi independen (X3). Average Connection Latency to Master Node merupakan jumlah rata-rata letency yang ada pada kontainer Micorservices saat sedang dalam proses build, test dan deploy sehingga menghasilkan sebuah kesatuan paket kontainer yang diproses oleh Jenkins master server, variable ini dapat diklasifikasikan menjadi independen variable atau (X4). Average Construction Time merupakan jumlah rata-rata waktu yang diperlukan saat proses cron atau otomatisasi berlangsung, proses tersebut berupa build, test dan deploy, pada variable ini dapat diklasifikasikan menjadi variable independen (X5).


#### N. Teknik Analisis Data

Metode analisis data yang digunakan dalam penelitian ini adalah dengan menggunakan analisis kuantitatif pada dataset yang diperoleh. Analisis data dapat dilakukan dengan menggunakan uji regresi linier berganda yang terdiri dari uji statistik t dan uji statistik F untuk membuktikan hipotesis yang dibentuk dalam penelitian dan untuk melihat tingkat signifikansinya.

Analisis regresi pada dasarnya sebuah pendekatan yang digunakan untuk mendefinisikan hubungan matematis yang menghubungkan antara variabel dependen (Y) dengan satu atau beberapa variabel independen (X). Hubungan matematis digunakan sebagai suatu model regresi yang digunakan untuk meramalkan atau memprediksi nilai output (Y) berdasarkan nilai input (X) tertentu. Dengan metode analisis regresi akan diketahui variabel independen yang benar-benar signifikan dapat mempengaruhi antar satu variable dengan yang lainnya sehingga dapat ditarik kesimpulan sebagai bukti data pembanding. Sebelum dilakukan uji model, peneliti melakukan analisis statistik deskriptif, serta uji asumsi klasik model regresi yang meliputi uji normalitas, uji autokorelasi, uji

heterokedastisitas, uji multikolinearitas, dan uji linieritas.

Sebagai jalan untuk memperkecil kesalahan atau human error dalam mengolah data statistik dan data absolute penelitian, peneliti menggunakan program atau aplikasi JASP Statistics. Program JASP adalah sebuah program aplikasi berlisensi Open Source yang mana merupakan perangkat lunak terbuka yang mampu menganalisis statistik secara lengkap. Program ini dipilih oleh peneliti karena memiliki keunggulan dibandingkan program atau software lainnya yaitu program ini berbasis user-interface bejalan pada sistem banyak sistem operasi terutama windows dan program ini sangat mudah dioperasikan (user-friendly) serta lengkapnya teknik-teknik analisis statistik yang tersedia. Program ini telah umum digunakan oleh para peneliti sebelumnya untuk menganalisis data penelitian. Berbagai disiplin ilmu pengetahuan, baik lingkup manajemen (riset pemasaran), biologi, pertanian, teknik, industri, psikologi, ilmu sosial maupun bidang lainnya, menggunakan software ini sebagai alat bantu mengolah/menganalisis data penelitian.



	Name	Success	Source Code Commit	Merge Code (tag)
1	ada	2299	177707	145
2	ata	957	142258	1827
3	arabid	1179	407872	896
4	arabidp	229	97043	827
5	arabid	1	27842	892
6	arabid	923	197052	1847
7	arabid	7469	252345	4124
8	arabid	1893	106897	2084
9	arabid	22344	489943	549
10	arabid	377	118295	1284
11	arabid	1	31081	1413
12	arabid	860	64235	13
13	arabid	7259	281122	908
14	arabid	15	28209	6841
15	arabid	1789	198844	1043
16	arabid	37	91473	1546
17	arabid	16591	121889	2262
18	arabid	256	35071	1788
19	arabid	385	174852	1936
20	arabid	3744	396997	585

Gambar 3. Sample Data Dengan JASP

### O. Analisis Statistik Deskriptif

Analisis statistik ini digunakan untuk menganalisis data dengan cara mendeskripsikan atau menggambarkan data yang telah terkumpul sebagaimana adanya tanpa bermaksud membuat kesimpulan yang berlaku untuk umum atau generalisasi. Analisis data awal dilakukan untuk

menggolongkan, mengurutkan dan menyederhanakan data sehingga mudah dibaca dan diinterpretasikan. Gambaran umum ini bisa menjadi acuan untuk melihat karakteristik data yang dapat diperoleh peroleh. Hal ini sangat penting karena dengan analisis deskriptif memungkinkan untuk bisa mengoreksi secara cepat data yang sudah dientri. Analisis statistik deskriptif yang digunakan di dalam penelitian ini terdiri dari Tabulasi dan dan presentase. Dalam hal ini, data pada nama-nama kontainer microservices dikategorikan berdasarkan kontainer microservices pod, yang berupa grup-grup kontainer berdasarkan jenis sistem operasi dan modul serta library yang ada.

Tabulasi distribusi frekuensi dibuat untuk data nominal yang menunjukkan besarnya masing-masing variabel independen di dalam sampel. Persentase menunjukkan besarnya proporsi variabel independen di dalam sampel. Tabel distribusi frekuensi berguna untuk mendeskripsikan ciri-ciri atau karakteristik dari suatu variabel, mempelajari distribusi dari variabel pokok dan memilih klasifikasiklasifikasi pokok untuk tabulasi silang.

- Minimum, maksimum, mean dan standar deviasi

Digunakan untuk menunjukkan tingkat variabel dependen. Minimum menunjukkan nilai terendah dari data yang ada di dalam sampel, sebaliknya maximum menunjukkan nilai tertinggi dari data yang ada di dalam sampel. Mean digunakan untuk menentukan rata-rata data yang ada di dalam sampel. Semakin kecil standar deviasi suatu variabel di dalam penelitian ini berarti semakin kecil sebarannya yang berarti nilai data makin bersifat homogen. Sebaliknya jika semakin besar sebarannya berarti makin bervariasi nilai datanya.

### P. Uji Regresi Linear Berganda

Pengujian statistik yang digunakan adalah dengan menggunakan uji statistik regresi linear berganda. Analisis ini digunakan untuk menghitung dan memperoleh gambaran bagaimana pengaruh antara variabel independen (X) dan variabel dependen (Y).



*Q. Uji Korelasi Ganda dan Uji Koefisien Determinasi*

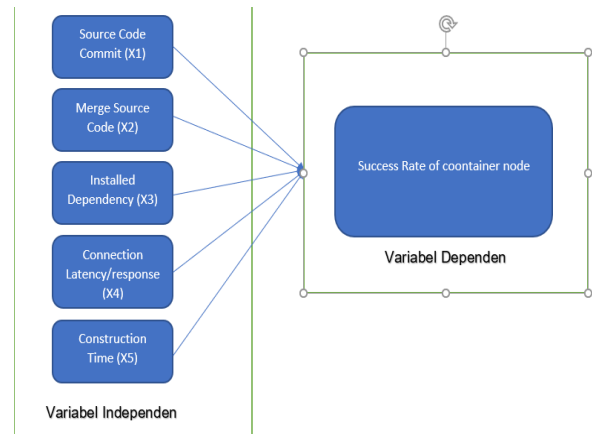
Koefisien determinasi (R<sup>2</sup>) mengukur seberapa jauh kemampuan model menerangkan variasi variabel dependen yang dirumuskan sebagai berikut:

$$R_y.X_1.X_2.X_3.X_4.X_5.X_6 = \frac{\sqrt{r^2y_1x_1+r^2y_1x_2+r^2y_1x_3+r^2y_1x_4+r^2y_1x_5+r^2y_1x_6}}{1-r^2x_1x_2x_3x_4x_5x_6}$$

Diantara nol dan satu. Nilai R<sup>2</sup> yang kecil berarti kemampuan variabel-variabel independen dalam menjelaskan variasi variabel dependen sangat terbatas. Nilai yang mendekati satu berarti variabel-variabel independen tersebut mampu memberikan hampir semua informasi yang dibutuhkan untuk memprediksi variasi variabel dependen yang terkait. Untuk mengatasi kelemahan R<sup>2</sup>, Henry Theil menyempurnakan persamaan R<sup>2</sup> tersebut yang dinamakan adjusted R<sup>2</sup>. Dengan adjusted R<sup>2</sup> menunjukkan bahwa dengan bertambahnya variabel-variabel independen akan semakin memperkecil nilai adjusted R<sup>2</sup>. Nilai adjusted R<sup>2</sup> masih bisa bertambah apabila nilai t absolut variabel yang ditambahkan lebih besar daripada 1, semakin besar nilai adjusted R<sup>2</sup> semakin baik pula modelnya.

**PEMBAHASAN**

Dalam penelitian ini, penulis menguraikan model penelitian ini dengan diagram yang akan digunakan sebagai tolak ukur dari bahasan yang akan ditulis. Model penelitian pada dasarnya adalah nilai abstraksi dari variabel-variabel yang sedang diteliti. Dalam hal ini model penelitian yang sesuai dengan judul penelitian ini. Perumusan variabel-variabel dengan keterkaitannya dengan dirumuskan dalam diagram sebagai berikut :



Gambar 4. Metode dan Skema Penelitian

*R. Analisis Statistik Deskriptif*

Pada pengujian analisis statistik deskriptif ini akan dibahas mengenai tahapan klasifikasi data berdasarkan dengan jenis sistem operasi yang digunakan oleh setiap node untuk otomatisasi Continuous Delivery / Continuous Integration pada kontainer microservices yang ada. Klasifikasi data dapat dilihat pada tabel :

Tabel 2. Frekuensi Data Berdasarkan Kontainer

No	Variable	Frekuensi	Presentase (%)
1	Jenis Sistem Operasi		
	prd-centos7-autorelease-8c-32g	1	1.2
	prd-centos7-builder-2c-1g	2	2.5
	prd-centos7-builder-2c-2g	5	6.3
	prd-centos7-builder-2c-8g	13	16.4
	prd-centos7-builder-4c-16g	2	2.5
	prd-centos7-builder-4c-4g	42	53.1
	prd-centos7-builder-8c-8g	5	6.3
	prd-centos7-docker-1c-4g	1	1.2
	prd-centos7-docker-2c-8g	1	1.2
	prd-centos7-robot-2c-8g	5	6.3
	prd-queue-intque-2c-1g	2	2.5

Pada tabel di atas dapat diketahui bahwa dari 79 node microservices kontainer yang dijadikan sampel dalam penelitian ini dibagi berdasarkan jenis sistem operasi yang digunakan, klasifikasinya dalam pembagian berdasarkan sistem operasi adalah sebagai berikut, prd-centos7-autorelease-8c-32g sebanyak 1 node atau 1,2%, prd-centos7-builder-2c-1g sebanyak 2 atau 2,5%, prd-



centos7-builder-2c-2g sebanyak 5 atau 6.3%, prd-centos7-builder-2c-8g sebanyak 13 atau 16.4%, prd-centos7-builder-4c-16g sebanyak 2 atau 2.5%, prd-centos7-builder-4c-4g sebanyak 42 atau 53.1%, prd-centos7-builder-8c-8g sebanyak 5 atau 6.3%, prd-centos7-docker-1c-4g sebanyak 1 atau 1.2%, prd-centos7-docker-2c-8g sebanyak 1 atau 1.2%, prd-centos7-robot-2c-8g sebanyak 5 atau 6.3%, prd-queue-intque-2c-1g sebanyak 2 atau 2.5%.

Data tersebut dibagi dengan tolak ukur lurus pada JASP yang diambil pada data sebanyak 79 node kontainers yang digunakan pada Opendaylight project. Tiap kontainer microservices merupakan bagian daripada sistem operasi pada node yang ada.

### S. Hipotesis

Pada penelitian ini, penulis melakukan penelitian terhadap analisa proses penggunaan dan implementasi dari arsitektur microservices berbasis kontainer yang ada pada pengembangan Opendaylight project yang merupakan perangkat lunak software defined network berbasis terbuka. Dalam kriteria ini akan diambil populasi data terhadap tingkat keberhasilan dalam penerapan microservices kontainer dengan menggunakan sample data yang telah terpublish di Bitergia Analytics Opendaylight dalam kurun waktu dari 1 Oktober 2017 - 1 Oktober 2018. Jumlah keseluruhan populasi yang diambil sebagai sample yaitu berupa 79 kontainer microservices yang mana memiliki 5 faktor pengaruh dalam kesuksesan implementasinya saat otomatisasi sumber kode dengan CI/CD pada Jenkins master server. Untuk mendapatkan kriteria yang diperlukan, maka ditetapkan hipotesis sebagai berikut :

- Jumlah node microservices kontainer pada setiap sub-project yang ada pada data statistik pengembangan Opendaylight menggunakan data yang terpublish pada rentang waktu 1 oktober 2017 - 1 oktober 2018. Jumlah kesuksesan dalam implementasi kontainer microservices pada 1 tahun terakhir ditulis sebagai data primer

tanpa melihat jumlah developer yang ada dan statistik pengembangan Opendaylight lainnya. Secara garis besar data hanya berdasarkan Jenkins node creation untuk setiap proses keberhasilan kontainersasi sub komponen dari Opendaylight.

- Jumlah node yang diperiksa datanya memiliki komposisi dan klasifikasi sistem operasi untuk kontainer microservices yang berbeda antar node lainnya.
- Jumlah dari variabel independen (X) terhadap variabel Y memiliki keterkaitan untuk menentukan tingkat kelayakan penerapan sistem kontainer microservices Opendaylight pada skala produksi.
- Variabel independen secara bersama memiliki tingkat pengaruh yang tinggi pada tingkat keberhasilan implementasi kontainer microservices arsitektur pada Opendaylight.

### T. Analisis Uji Asumsi

Sebelum hasil regresi yang diperoleh diinterpretasikan maka terlebih dahulu diuji apakah terdapat pelanggaran asumsi regresi linier berganda. Dalam penelitian ini akan dilakukan pengujian Korelasi Ganda (R) dan Uji Koefisien Determinasi (R<sup>2</sup>), Uji Signifikansi Simultan (Uji Statistik -F), Uji Signifikansi (Uji Statistik -T), Uji Asumsi, Uji Multikolinieritas, Uji Normalitas, dan Uji Heteroskedastisitas.

### U. Uji Normalitas

Normalitas menunjukkan bahwa variabel dependen dan variabel independen dalam model regresi. Uji normalitas bertujuan untuk menguji apakah sampel yang digunakan mempunyai distribusi normal atau tidak. Dalam model regresi linier, asumsi ini ditunjukkan oleh nilai error yang berdistribusi normal. Model regresi yang baik adalah model regresi yang dimiliki distribusi normal atau mendekati normal, sehingga layak dilakukan pengujian secara statistik. Pengujian normalitas data menggunakan Test of Normality Kolmogorov-Smirnov dalam program JASP. Menurut Singgih Santoso (2012:293) dasar pengambilan keputusan bisa

dilakukan berdasarkan probabilitas (Asymtotic Significance).

One-Sample Kolmogorov-Smirnov Test		
		Unstandardized Residual
N		71
Normal Parameters <sup>a,b</sup>	Mean	.0000001
	Std. Deviation	875.3867176
Most Extreme Differences	Absolute	.17
	Positive	.17
	Negative	-.13
Test Statistic		.17

Gambar 5. Uji Normalitas

V. Uji Autokorelasi

Autokorelasi dapat diartikan untuk menunjukkan bahwa terdapat korelasi antara error dengan error periode sebelumnya dimana pada asumsi klasik hal ini tidak boleh terjadi. Uji autokorelasi dilakukan dengan menggunakan metode Durbin Watson yang terdapat pada program aplikasi JASP. Pengujian ada atau tidaknya autokorelasi dilakukan dengan menggunakan metode Durbin-Watson. Adapun cara mendeteksi terjadi autokorelasi dalam model analisis regresi dengan menggunakan Durbin-Watson.

Kriteria pengambilan keputusan :

- Jika nilai Durbin Watson  $d < du$  atau  $(4 - du) < du$ ,  $H_0$  ditolak, terdapat nilai autokorelasi positif atau negatif.
- Jika nilai Durbin Watson  $du < d < 4-du$ ,  $H_0$  diterima, tidak terdapat autokorelasi positif atau negatif.

Tabel 3. Hasil Analisis

Model Summary									
Model	R	R <sup>2</sup>	Adjusted R <sup>2</sup>	RMSER <sup>2</sup>	Change F	Change df1	df2	p	Durb
1	0.279	0.078	0.014	277.0	0.078	1.228	5	73	0.305

W. Uji Multikolinearitas

Uji multikolinearitas dapat bertujuan untuk menguji apakah dalam hasil regresi ditemukan adanya nilai korelasi diantara variabel bebas antar satu dengan variable lainnya. Multikolinearitas bisa menunjukkan bahwa antara nilai dari variabel independen yang mempunyai hubungan langsung (korelasi) yang sangat kuat. Tolerance merupakan nilai 1-R<sup>2</sup> dari regresi berbanding dengan nilai tersebut dengan sisa variabel bebas lainnya. Nilai tolerance yang mendekati 0 menyatakan adanya kolinieritas antara suatu variabel bebas tersebut dengan sisa variabel bebas lainnya. Indikator kolinieritas lainnya merupakan Variance Inflation Factor (VIF) yang merupakan kebalikan (resiprokal) dari nilai tolerance. Batasan yang biasa digunakan adalah 0,1 untuk tolerance yang berarti batas angka 10 untuk VIF (Hair et .,1998).

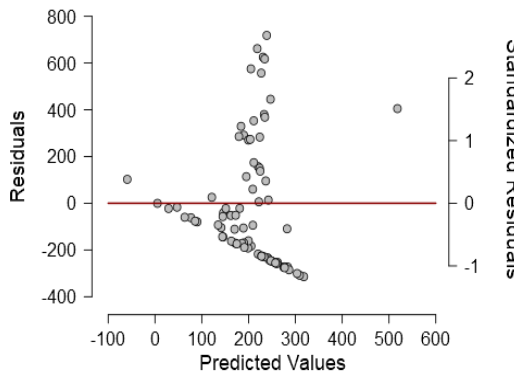
Tabel 4. Uji Multikolinearitas

Coefficients							Collinearity Statistics	
Model		Unstandardized	Standard Error	Standardized	t	p	Tolerance	VIF
1	(Intercept)	190.673	69.315		2.751	0.007		
	Source Code Commit	-0.379	0.255	-0.183	1.489	0.141	0.837	1.195
	Merge Code (Avg)	-2.111	2.197	-0.165	0.961	0.340	0.431	2.322
	Installed Dependency	0.128	0.137	0.111	0.933	0.354	0.895	1.118
	Latency to Master Node	0.526	6.493	0.011	0.081	0.936	0.651	1.536

X. Uji Heteroskedastisitas

Heteroskedastisitas adalah dimana situasi tidak konstan varians. Untuk mendeteksi ada tidaknya heteroskedastisitas dilakukan pengujian dengan menggunakan metode Glejser yang selanjutnya dilakukan perbandingan antara nilai sig-t dengan 0,05. Jika Sig-t merupakan nilai hitung lebih kecil dari 0,05 maka akan terjadi heteroskedastisitas, begitu juga

sebaliknya, jika sig-t<sub>hitung</sub> lebih besar dari 0,05 maka tidak akan terjadi heteroskedastisitas. Hasil uji Glejser dapat ditunjukkan pada tabel berikut:



Gambar 6. Uji Heteroskedastisitas

Berdasarkan hasil scatter plot di atas diketahui bahwa pencaran data tidak menunjukkan suatu pola tertentu. Pencaran data menyebar secara acak sehingga peneliti menyimpulkan tidak adanya masalah heterokedastistas pada residual. Begitu pula dengan hasil uji Glejser yang menunjukkan bahwa nilai signifikansi untuk masing-masing variabel independen pada persamaan model regresi terhadap nilai absolut residualnya > 0,05. Dengan demikian dapat disimpulkan bahwa data tersebut bersifat homokedastis.

Y. Uji Hipotesis Signifikasi (Uji F)

Uji F digunakan untuk menguji apakah secara bersama-sama seluruh variabel independen mempengaruhi secara signifikan terhadap variabel dependen.

H0 :  $\beta_1 = \beta_2 = \beta_3 = \beta_4 = \beta_5 = 0$   
 Source code commit, merge source code, installed dependency, connection latency dan construction time secara bersama-sama tidak berpengaruh terhadap tingkat implementasi arsitektur microservices kontainer dengan otomatisasi CI/CD pada Jenkins node dalam lingkungan development Opendaylight project.

Ha :  $\beta_1 = \beta_2 = \beta_3 = \beta_4 = \beta_5 = 0$

Source code commit, merge source code, installed dependency, connection latency, dan construction time secara bersama-sama berpengaruh terhadap tingkat implementasi arsitektur microservices kontainer dengan otomatisasi CI/CD pada Jenkins node dalam lingkungan development Opendaylight project.

Hasil dari Uji Signifikansi dengan menggunakan JASP dapat dilihat pada tabel berikut ini :

Tabel 5. Hasil Uji Signifikansi

ANOVA						
Model		Sum of Squares	df	Mean Square	F	p
1	Regression	471274	5	94255	2.428	0.025
	Residual	5.603e +6	73	76748		
	Total	6.074e +6	78			

Berdasarkan tabel 5 diperoleh nilai F-hitung sebesar 5,905 dengan nilai Sig sebesar 0,000. Hal ini menunjukkan bahwa nilai F<sub>hitung</sub> lebih besar dari F<sub>tabel</sub> 2,471 dan nilai Sig lebih kecil dari 0,05. Dengan demikian H<sub>0</sub> ditolak dan H<sub>a</sub> diterima. Sehingga memiliki hasil bahwa source code commit, merge source code, installed dependency connection latency dan construction time terdapat pengaruh pada keberhasilan dalam implementasi kontainer untuk DevOps dalam proses pengembangan Opendaylight project yang dilakukan dalam rentan waktu oktober 2017 hingga oktober 2018. Dengan mengacu pada uji F ini seharusnya implementasi arsitektur microservices dapat dijalankan dengan mematuhi kriteria-kriteria yang terdapat didalam penggunaan kontainer pada Docker atau pun aplikasi orkestrasi multiple kontainer dengan kuberntes. Sehingga dapat mengurangi sumber daya yang ada pada lingkungan produksi.

### Z. Uji T

Uji t digunakan untuk mengetahui pengaruh secara parsial variabel bebas terhadap variabel terikat. Pengujian ini yaitu dengan membandingkan nilai probabilitas atau p-value (sig-t) dengan taraf signifikansi 0,05. Jika nilai p-value lebih kecil dari 0,05 maka  $H_a$  diterima, dan sebaliknya jika p-value lebih besar dari 0,05 maka  $H_0$  ditolak.

Tabel 6. Hasil Uji T

Coefficients				
Model		Unstandardized	Standard Error Standardized	
1.	(Intercept)	190.673	69.315	
	Source Code Commit	-0.379	0.255	-0.183
	Merge Code (Avg)	-2.111	2.197	-0.165
	Installed Dependency	0.128	0.137	0.111
	Latency to Master Node	0.526	6.493	0.011
	Construction Time	7.304	4.249	0.299

Kesimpulan daripada hasil Uji T diatas adalah sebagai berikut :

- Pengaruh source code commit terhadap faktor keberhasilan  
Berdasarkan dari tabel dapat diperoleh nilai T-hitung sebesar 1.48 dengan nilai Sig sebesar 0.141. Hal ini menunjukkan bahwa nilai t-hitung lebih kecil daripada nilai t-tabel 2.37 dan nilai sig lebih kecil daripada 0,05. Dengan demikian  $H_0$  ditolak dan  $H_a$  ditolak.
- Pengaruh merge source code terhadap faktor keberhasilan  
Berdasarkan tabel 4.9 dapat diperoleh dengan hasil nilai t-hitung sebesar -0.961 dan nilai Sig sebesar 0.340. Hal ini menunjukkan bahwa nilai T-hitung lebih kecil dari nilai t-tabel 2.37 dan nilai Sig 0.340 lebih besar dari 0.05. Dengan demikian dapat diambil kesimpulan bahwa  $H_0$  ditolak dan  $H_a$  diterima.
- Pengaruh installed dependency terhadap faktor keberhasilan  
Berdasarkan tabel 4.9 dapat diperoleh dengan hasil nilai t-hitung sebesar 0.933 dan nilai Sig sebesar 0.354. Hal ini menunjukkan bahwa nilai T-hitung lebih kecil dari nilai t-tabel 2.37 dan nilai Sig 0.354 lebih besar dari 0.05.

Dengan demikian dapat diambil kesimpulan bahwa  $H_0$  ditolak dan  $H_a$  diterima.

- Pengaruh connection latency terhadap faktor keberhasilan  
Berdasarkan tabel 4.9 dapat diperoleh dengan hasil nilai t-hitung sebesar 0.081 dan nilai Sig sebesar 0.936. Hal ini menunjukkan bahwa nilai T-hitung lebih kecil dari nilai t-tabel 2.37 dan nilai Sig 0.936 lebih besar dari 0.05. Dengan demikian dapat diambil kesimpulan bahwa  $H_0$  ditolak dan  $H_a$  diterima.
- Pengaruh construction time terhadap faktor keberhasilan  
Berdasarkan tabel 4.9 dapat diperoleh dengan hasil nilai t-hitung sebesar 1.719 dan nilai Sig sebesar 0.936. Hal ini menunjukkan bahwa nilai T-hitung lebih kecil dari nilai t-tabel 2.37 dan nilai Sig 0.090 lebih besar dari 0.05. Dengan demikian dapat diambil kesimpulan bahwa  $H_0$  ditolak dan  $H_a$  diterima.

### HASIL PEMBAHASAN

Melalui hasil uji –t yang didapat maka dapat disimpulkan bahwa setiap nilai dari variabel-variabel independen memiliki masing-masing peranan nilai yang berbeda dengan dijabarkan antara lain; faktor source code commit, tidak memiliki pengaruh terhadap kesuksesan implementasi arsitektur microservices yang cukup berarti, karena dapat disimpulkan bahwa jumlah nilai rata-rata commit terhadap kode yang terjadi di server git hanya berpengaruh pada setiap repository local yang ada untuk proses development, tetapi tidak mempengaruhi master node pada server jenkins ketika proses enkapsulasi kontainer image berlangsung; faktor merge source code atau repository memiliki pengaruh daripada faktor keberhasilan implementasi arsitektur microservices hal ini bisa di interperasikan dengan jumlah setiap repository yang terdapat pada git sever ketika akan di enkapsulasi menjadi paket kontainer image untuk sistem microservices maka jumlah branch yang ada menentukan jenkins master node untuk dapat melakukan create image atau tidak.

**KESIMPULAN**

Dengan hasil uji dan analisa sebelumnya maka ditetapkan bahwa, variabel source code commit tidak memiliki pengaruh yang signifikan terhadap rasio kesuksesan implementasi kontainer microservices. Uji model regresi menunjukkan bahwa tingkat rasionya tergolong tidak valid dan rendah. Hal ini dapat diartikan bahwa berapapun source code yang di otomatisasi pada server Git tidaklah mempengaruhi implementasi kontainer yang berhasil terdeploy. Oleh karena itu jika didalam lingkungan produksi digunakan, maka sysadmin ataupun DevOps engineer dapat melakukan otomatisasi source code commit secara berkelanjutan.

Conference on Computing, Communication and Automation. (2017).

**DAFTAR PUSTAKA**

- Nuha Alshuqayran, Nour Ali and Roger Evans. A Systematic Mapping Study in Microservice Architecture. In IEEE 9th International Conference on Service-Oriented Computing and Applications (2016).
- Muhammad Waseem, Peng Liang. Microservices Architecture in DevOps. In 24th Asia-Pacific Software Engineering Conference Workshops (2017).
- Kratzke, N. (2015). About Microservices, Containers and their Underestimated Impact on Network Performance. In Cloud Computing Conf (2015).
- Oleg Mironov. DevOps Pipeline with Docker. Helsinki Metropolia University of Applied Sciences, Thesis (2018).
- Dharmendra Shadija and Mo Rezai, Richard Hill. Towards an Understanding of Microservices. In Proceedings of the 23rd International Conference on Automation & Computing, University of Huddersfield, Huddersfield, UK, 7-8 September (2017).
- Zhongxiang Xiao, Inji Wijegunaratne, Infosys Australia, Xinjian Qiang. Reflections on SOA and Microservices. In 4th International Conference on Enterprise Systems (2016).
- Vindeep Singh, Sateesh K Peddoju. Container-based Microservice Architecture for Cloud Applications. In International