PERANCANGAN SISTEM *AUTO-SCALING* PADA *CLOUD COMPUTING* DENGAN MENGGUNAKAN SISTEM PREDIKSI *SIMPLE MOVING AVERAGE*

Praditya Wahyu Wardhana.[1)], **Novian Anggis Suwastika**[2)]

[1), 2)] *Teknik Informatika Universitas Telkom Bandung*
Jl Telekomunikasi Terusan Buah Batu, Dayeuhkolot Bandung 40257
Email : praditya wahyu@gmail.com[1)], *anggis@gmail.com* [2)]

Abstrak

Layanan Cloud Computing bersifat on-demand *dan* rapid-elasticy *untuk memenuhi kebutuhan penyimpanan data maupun komputasi. Untuk mendukung layanan* on-demand *dan* rapid-elastic, *diperlukan kemampuan pengalokasian* resource *secara otomatis yang dikenal sebagai metode* auto-scaling. *Penerapan metode* auto-scaling *dapat dilakukan dengan dua metode yaitu prediksi dan dinamik. Pada penelitian ini metode* auto-scaling *yang digunakan adalah metode prediksi. Metode prediksi dipilih karena mendukung skalabilitas* dan self-adaptive. *Simple Moving Average (SMA) merupakan salah satu metode prediksi menggunakan keterurutan waktu dengan memanfaatkan data dari masa lalu. Dibandingkan dengan metode prediksi lain seperti Exponential Moving Average atau Autoregresive Moving Average, SMA merupakan model prediksi dengan komputasi paling sederhana dan mungkus. Pada paper ini diajukan perancangan sistem* cloud computing *untuk mendukung implementasi* auto-scaling *menggunakan metode SMA. Perancangan sistem ini diharapkan dapat mendukung analisis performasi metode SMA dengan parameter mean time to failure (MTTF), mean time between failure (MTBF), mean time to repair (MTTR), Availability Operational, Down Time dan Up Time serta ketepatan predictive system.*

Kata kunci: *Cloud-computing, auto-scaling, sistem prediksi, simpel moving average (SMA)*

1. Pendahuluan

Layanan *cloud computing* yang bersifat *on-demand service* dan *rapid-elasticy* banyak dipilih sebagai layanan yang memberikan kemudahan pada jaringan internet saat ini terutama dalam penyimpanan data dan komputasi. Untuk mendukung layanan *on-demand* dan *rapid-elasticy* dibutuhkan metode *auto-scaling*, yaitu metode pengelolaan *resources* komputasi secara otomatis yang dilakukan oleh sistem. Kemampuan otomatisasi ini menjadikan *auto-scaling* bersifat *scalable* dan *self-adaptive*. Salah satu implementasi *auto-scaling* yaitu dengan menerapkan konsep prediksi atau mengacu kepada kondisi tertentu yang sudah ditentukan sebelumnya.

Metode *auto-scaling* yang *scalable* dan *self-adaptive* dapat menjaga efisiensi kebutuhan *server cloud* karena metode ini akan menyesuaikan kebutuhan pengguna terhadap *resources cloud* yang digunakan. Contoh kasusnya adalah ketika lebih dari 100 layanan baru dijalankan maka kapasitas memori, utilitas *processor* dan *resource* yang lain harus ditingkatkan agar layanan tetap berjalan dengan optimal. Permasalahan yang muncul adalah diperlukan suatu metode prediksi yang tepat agar kebutuhan *resource* dari layanan dapat dipenuhi dengan tepat. *Resource* yang diberikan tidak boleh kurang atau berlebihan, karena akan menurunkan kualitas layanan maupun performansi sistem.

Model prediksi *auto-scaling* yang digunakan pada penelitian ini adalah metode *Simple Moving Average* (SMA). Pada metode SMA, prediksi dilakukan adalah dengan membuat pola prediksi dari data masa lalu berdasarkan data *time series* atau keterurutan waktu kedatangan data. Data keterurutan waktu kedatangan sering memunculkan ketidakberaturan yang berpengaruh terhadap beragamnya prediksi. Pendekatan untuk meminimalisasi efek ketidakberaturan ini adalah dengan menggunakan rataan dari beberapa nilai yang sedang diamati.

Pada penelitian ini dirancang sistem *cloud computing* dengan framework *opensource* OpenStack untuk mengimplementasikan *auto-scaling* berbasis metode SMA. Perancangan sistem *cloud computing* yang dibangun pada penelitian ini meliputi kebutuhan infrastruktur untuk pembangunan server, model topologi jaringan dan skenario pengujian untuk mendapatkan hasil pengujian yang selanjutnya akan digunakan sebagai dasar untuk analisis performasi metode SMA berdasarkan parameter pengujian seperti *mean time to failure (MTTF), mean time between failure (MTBF), mean time to repair (MTTR), Availability Operational, Down Time* dan *Up Time* serta ketepatan *predictive System.*

Paper ini disusun menjadi tiga bab utama, yaitu pendahuluan, pembahasan dan kesimpulan. Bab pendahuluan menjelaskan latar belakang dan tujuan penelitian. Bab pembahasan dibagi menjadi lima bagian, yaitu: pembahasan singkat tentang teknologi *cloud computing*, pembahasan tentang mekanisme *auto-scaling*

dan metode penyusunnya, pembahasan metode SMA sebagai salah satu metode prediksi *auto-scaling*, pembahasan mengenai *framework cloud computing* OpenStack dan yang terakhir adalah perancangan sistem *cloud computing* untuk mendukung implementasi metode prediksi *auto-scaling* dengan SMA. Bab terakhir adalah bab kesimpulan tentang perancangan sistem *cloud computing* serta penelitian selanjutnya yang akan dilakukan.

2. Pembahasan

2.1. Cloud Computing

Cloud computing dapat didefinisikan sebagai sebuah cara baru komputasi yang menerapkan konsep *scalability* dan *virtualization* untuk penggunaan *resource* komputasi pada jaringan internet. Dengan menggunakan layanan *cloud computing*, pengguna yang menggunakan berbagai macam *device* seperti PC/Laptop, *smartphone* dan PDA dapat mengakses program, media penyimpanan dan *application-development platforms* di atas jaringan internet, melalui *service* yang diberikan penyedia layanan *cloud computing*. Keuntungan dari teknologi *cloud computing* diantaranya adalah:

1. *Cost saving* bagi pengguna, karena pengguna tidak perlu membeli atau menyediakan server untuk komputasi.
2. *High availability*, *resource* selalu tersedia dimanapun dan kapanpun pengguna membutuhkan.
3. *Easy scalability*, pengguna dapat mengatur jumlah *resource* yang dibutuhkan sesuai kebutuhan pengguna.

Berdasarkan layanan yang diberikan kepada pengguna, teknologi *cloud computing* dapat dibedakan menjadi beberapa layer arsitektur service. Layer atas adalah layanan *cloud computing* yang disebut dengan *Software as a Service* (SaaS), layanan ini memungkinkan pengguna untuk melakukan *running* aplikasi secara *remote* terhadap *cloud*. Layer layanan berikutnya adalah *Infrastructure as a Service* (Iaas) yang menyediakan penggunaan computing *resource* sebagai *service cloud*, termasuk virtualisasi komputer yang menjamin *processing power* dan *bandwidth* untuk *storage* atau media penyimpanan dan internet akses.

Layer *Platform as a Service* (PaaS) memiliki layanan yang hampir serupa dengan IaaS, perbedaannya adalah PaaS menyediakan layanan *operating system* dan beberapa *service* yang diperlukan untuk menjalankan beberapa aplikasi. Sederhananya PaaS adalah IaaS dengan fitur untuk *custom software stack* yang dapat digunakan untuk menjalankan suatu aplikasi. *Layer* layanan yang terakhir adalah *data Storage as a Service* (dSaaS) yang memungkinkan penyimpanan data bagi pengguna termasuk kebutuhan *bandwidth* bagi media penyimpanannya.

Untuk kemampuan akses layanan, teknologi *cloud computing* dibedakan menjadi beberapa jenis yaitu *public cloud*, *private cloud*, serta *hybrid cloud*. Pada *public cloud* atau eksternal *cloud*, *resource* untuk komputing secara dinamis dikelola di atas jaringan internet melalui *web application* atau *web services* dari suatu off-site pihak ketiga yang memberikan layanan. *Public cloud* dijalankan oleh pihak ketiga dan aplikasi dari pengguna yang berbeda-beda yang saling bercampur bersama pada *cloud server*, sistem penyimpanan maupun jaringan.

Private cloud atau internal cloud merupakan layanan cloud computing yang dijalankan pada *private networks*, *private cloud* ini dibangun untuk *exclusive client* yang memungkinkan untuk melakukan full-control terhadap data, aplikasi, *security* maupun *quality of service*. *Private cloud* ini dapat dibangun dan dikelola oleh pemilik perusahaan, organisasi atau kelompok IT, atau bahkan *cloud provider* itu sendiri.

Hybrid cloud mengkombinasikan model *multiple public* dan *private cloud*, yang dapat mengakomodir bagaimana distribusi aplikasi diantara sebuah *public cloud* dengan *private cloud* [2].

2.2. Auto-scaling

Auto-scaling merupakan suatu teknik yang digunakan untuk mengelola *resource* komputasi secara otomatis, sehingga administrator tidak perlu melakukan penambahan ataupun pengurangan *resource* secara *manual* ketika menjalankan suatu layanan *cloud* [4]. Penerapan *auto-scaling* ini akan meningkatkan efektivitas dan efisiensi *resource* komputing *cloud* maupun mengurangi beban administrator *cloud.*

Sistem *auto-scaling* dapat dijalankan dengan berbagai *rule* sehingga proses *scaling*-nya dapat dijalankan, ada dua jenis *rule* yang digunakan dalam *auto-scaling* ini yaitu *predictably* (prediksi) dan *dynamically* [14]. Metode prediksi menggunakan aturan dengan melakukan prediksi terhadap layanan yang dijalankan, dengan melakukan prediksi akan diketahui kapan kira-kira terjadi lonjakan trafik ataupun penurunan trafik yang kemudian akan dilakukan *scaling* yang disesuaikan dengan trafik yang dibutuhkan, misalnya suatu ketika trafik naik secara drastis maka CPU *utilization* dapat ditingkatkan menjadi 50% dari normal, *Memory usage* ditingkatkan, atau bahkan penambahan *resource* dari server lainnya dan seterusnya. Sistem prediksi sangat efektif jika *predictive system* yang digunakan tepat, namun sebaliknya, sistem prediksi ini dapat memperburuk penggunaan *resource* jika *predictive system* yang digunakan tidak tepat.

Dynamically melakukan proses *scaling* secara dinamis tidak tergantung dari trafik yang berjalan, metode ini akan melakukan *scaling* untuk layanan-layanan tertentu yang telah diatur oleh administrator, misalnya ketika

sistem menjalankan layanan dari Amazon EC2 maka CPU *utilization* ditingkatkan beberapa persen, memory ditingkatkan menjadi beberapa gigabytes, port-port tertentu dibuka, dan sebagainya. Metode ini akan baik jika memang layanan tersebut ketika dijalankan membutuhkan *resource* yang sesuai, namun jika layanan tersebut dijalankan sedangkan pemakaian layanan sangat sedikit maka hal tersebut dapat mengurangi efisiensi penggunaan *resource* dan menambah biaya dari *resource* yang berjalan.

2.3. Metode Simple Moving Average (SMA)

SMA merupakan salah satu jenis metode prediksi berdasarkan *time series* atau keturutan waktu kuantitatif dalam teori peramalan. Metode *simple moving average* menggunakan nilai di masa lalu yang digunakan sebagai acuan dalam melakukan prediksi di masa depan. Secara umum tujuan dari jenis peramalan *time series* adalah menemukan pola dalam deret historis dari suatu data dan mengeskploitasinya untuk dijadikan pola di masa depan.

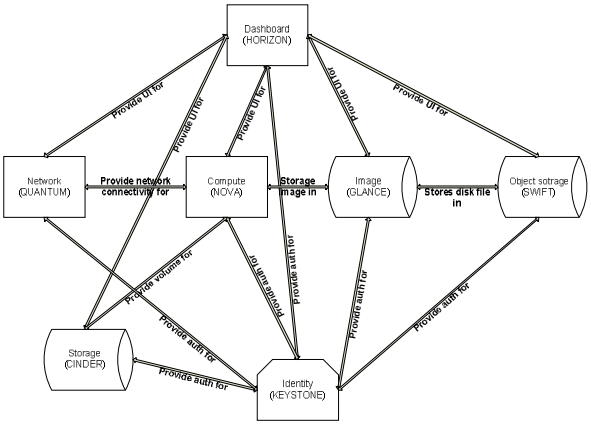
Data *time series* seringkali mengandung ketidakteraturan yang akan menyebabkan prediksi yang beragam. Untuk menghilangkan efek yang tidak diinginkan dari ketidak-teraturan ini, metode SMA mengambil beberapa nilai yang sedang diamati, memberikan rataan, dan menggunakannya untuk memprediksi nilai untuk periode waktu yang akan datang [15]. Semakin banyak jumlah pengamatan yang dilakukan, maka pengaruh metode *moving average* akan lebih baik. Meningkatkan jumlah observasi akan menghasilkan nilai peramalan yang lebih baik karena metode ini cenderung meminimalkan efek-efek pergerakan yang tidak biasa yang muncul pada data [10].

Metode SMA dapat dirumuskan sebagai berikut [15]:

$$A_t = \frac{D_t + D_{t-1} + D_{t-2} + \cdots + D_{t-N+1}}{N} \qquad (1)$$

Dengan:
N adalah total jumlah periode rataan.
A_t adalah prediksi pada periode $t + 1$.

2.4. Open Stack



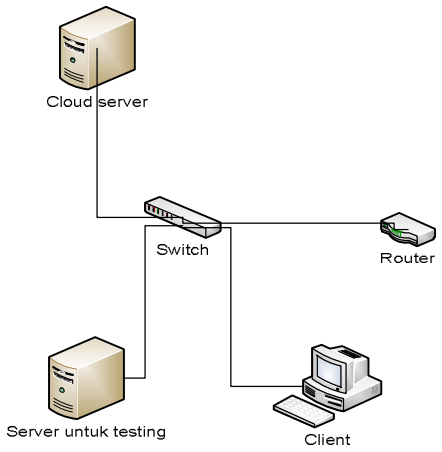
Gambar 1. *Arsitektur Openstack [16]*

OpenStack merupakan salah satu *framework* yang bersifat *opensource* untuk membangun *public* dan *private* cloud computing. Seperti yang ditunjukkan pada Gambar 1 arsitektur OpenStack disusun dari tujuh komponen yang memiliki fungsionalitas yang berbeda-beda, ketujuh komponen tersebut adalah :

- **Dashboard (horizon)**
  Merupakan *interface* untuk akses ke semua *service* OpenStack secara *web based.*
- **Compute (nova)**
  Komponen yang memungkinkan *virtual machine* berjalan di atas sistem, nova mampu menjalankan beberapa jenis *virtual machine* seperti KVM, Xen, LXC, Hyper-V, ESX. Selain itu nova menangani kontrol *services* seperti *schedulling,* API calls, dan sebagainya.
- **Block storage (cinder)**
  Komponen yang menyediakan *block* penyimpanan untuk *virtual machine.*
- **Networking (quantum)**
  Komponen yang menyediakan layanan untuk koneksi jaringan diantara perangkat yang dikelola oleh service OpenStack.
- **Image service (glance)**
  Komponen untuk penyimpanan virtual disk image yang biasanya diakses oleh nova.
- **Object store (swift)** :
  Komponen untuk penyimpanan file namun tidak mount directories seperti *fileserver*.
- **Identity service (keystone)**
  Komponen untuk mengatur autentikasi dan otorisasi unutk semua OpenStack *service* maupun mengelola user.

2.5. Perancangan Sistem Cloud Computing

Untuk mengimplementasikan *auto-scaling* menggunakan metode prediksi SMA perlu dirancang sistem *cloud computing* yang mendukung. Framework OpenStack dipilih untuk mengimplementasikan *cloud computing* karena selain *open source*, *framework* ini juga memiliki banyak dukungan untuk pengembangannya, baik dari komunitas maupun dukungan infrastruktur.



Gambar 2. *Topologi jaringan cloud computing*

Arsitektur topologi *cloud computing* yang akan digunakan ditunjukkan pada Gambar 2. Satu *server* digunakan untuk membangun sistem *cloud, server testing* untuk membangun *web server* dan *video streaming server* untuk melakukan pengujian dan satu unit klien (untuk kebutuhan analisis dampak jumlah pengguna, klien baru akan dibuat secara virtual). Ketiga komponen terhubung dengan router untuk dihubungkan dengan jaringan internet. Untuk spesifikasi masing-masing komponen ditunjukkan pada Tabel 1.

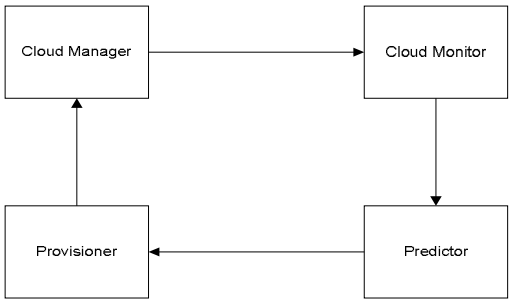
Untuk model rancangan model *auto-scaling* yang dibangun untuk implementasi SMA ditunjukkan pada Gambar 3. *Predictor* akan melakukan prediksi terhadap *resource* yang harus digunakan dalam melayani suatu permintaan dari pengguna. Pada sistem akan didefinisikan *treshold* yang digunakan sebagai batasan untuk proses scaling, *treshold* atas didefinisikan untuk melakukan *scale up* dan *treshold* bawah didefinisikan untuk melakukan *scale down*. Ketika suatu data atau informasi yang dikirim melewati batas *treshold* maka *triger* akan aktif untuk melakukan proses *scaling*, script yang berisikan prediksi untuk pengalokasian *resource* akan diaktifkan yang dikirimkan dari provisioner kepada *cloud* manager sehingga *cloud* manager akan membuat alokasi *resource* sesuai dengan apa yang diminta oleh *provisioner*, akhirnya *cloud* manager mengalokasikan *resource* untuk layanan yang dijalankan.

Layanan yang berjalan di *cloud* akan selalu dimonitor oleh *cloud* monitor sehingga dapat diketahui apakah perlu melakukan *scale up* atau *scale down*. *Treshold* atas pada penelitian ini dapat didefinisikan sebagai besar total alokasi memori current/saat itu dikurangi dengan jumlah rata-rata memori yang dikurangi oleh *service*s yang berjalan, sedangkan untuk *treshold* bawah pada sistem yang dibangun ini didefinisikan dengan 15% dari total penggunaan memory yang ada.

Setelah pembangunan sistem, disusun skenario pengujian untuk meneliti pengaruh dari *auto-scaling* dengan *predictive system* pada *cloud computing* yang dibangun dengan beberapa parameter dan data seperti *uptime, downtime*, MTTR, MTTF dan MTBF. Performansi *auto-scaling* akan diketahui setelah melakukan pengujian terhadap sistem dan seberapa tepat prediksi dari *simple moving average* yang diimplementasikan untuk menjalankan *auto-scaling*.

Tabel 1. *Spesifikasi Komponen Infrastruktur*

Komponen	Server Cloud	Server Testing	Client
Processor	Intel i7	Core2Duo	Core2Duo
Memory	4GB	2GB	1GB
Hardisk	50 GB	50 GB	50 GB
Aplikasi	OpenStack	Web Server dan Video Server	Menyesuaikan



Gambar 3. *Model auto-scaling untuk SMA*

Rencana skenario pengujian untuk menganalisis performasi dari metode SMA adalah sebagai berikut:

- **Skenario 1: Pengujian untuk menganalisis permintaan layanan.**
  Permintaan layanan dibedakan menjadi dua yaitu layanan yang bersifat stabil dan layanan yang bersifat fluktuatif. Pada layanan stabil, *web server* yang melayani aktifitasnya cenderung sama setiap waktunya. Durasi pengujian dilakukan dalam selang waktu 1 minggu. Pada permintaan layanan yang bersifat fluktuatif *video streaming server* yang dikondisikan penggunaannya tidak tentu setiap waktunya. Durasi pengujian dilakukan dalam selang waktu 1-2 minggu.
- **Skenario 2: Pengujian untuk menganalisis pengaruh jumlah klien.**
  Dari pengujian di skenario 1 dan skenario 2, masing-masing diimplementasikan dengan klien tunggal dan multi klien (2-10 klien), dari implementasi tersebut dianalisis seberapa besar pengaruhnya terhadap *auto-scaling*.

Dari pengujian sistem akan didapatkan data-data untuk penelitian selanjutnya yaitu analisis performasi *auto-scaling* dengan metode prediksi SMA. Data yang akan diambil dari pengujian yang dilakukan adalah *uptime* dan *downtime server*, *Mean Time To Repair* (MTTR) yang menggambarkan waktu rata-rata dari server untuk dapat melakukan *recovery* layanan setelah mengalami *failure* atau *down*, *Mean Time To Failure* (MTTF) menggambarkan jumlah waktu rata-rata suatu *server* mengalami *failure*, *Mean Time Between Failure* (MTBF) yang menjelaskan jumlah rata-rata waktu dari *server failure* yang pertama dengan *server failure* yang selanjutnya, dan yang terakhir adalah *Availability Operational* yaitu lama operasional yang dapat dijalankan oleh server. Serta seberapa tepat (optimal) penggunaan *resource* yang dialokasikan dengan metode SMA pada *auto-scaling cloud computing* dengan melihat utilitas masing-masing *resource* yang dialokasikan.

3. Kesimpulan

Penerapan metode prediksi yang tepat akan meningkatkan performansi *auto-scaling* pada *cloud computing*. SMA merupakan metode prediksi yang memanfaatkan keteraturan waktu permintaan layanan. Dengan perancangan implementasi *cloud computing* dengan *framework* OpenStack diharapkan data-data yang diperlukan dapat dikumpulkan untuk dapat dilakukan analisis lebih lanjut terhadap performasi *auto-scaling cloud computing*. Penelitian selanjutnya yang akan dilakukan adalah mengimplementasikan rancangan arsitektur yang diajukan pada penelitian ini, kemudian akan dilakukan pengambilan data dengan skenario pengujian yang sudah dipaparkan.

Hasil data yang didapatkan akan dianalisis, untuk melihat performasi dari metode SMA terhadap *auto-scaling* yang diterapkan pada *cloud-computing* dengan beberapa parameter seperti MTTR, MTTF, MTTB dan *Availability Operational*.

Daftar Pustaka

[1] Anthony T. Velte, Toby J. Velte, Ph.D., Robert Elsenpeter,2010, Cloud Computing A Practical Approach, New York, McGraw-Hill Companies.

[2] Borko Furht , Armando Escalante,2010, Handbook of Cloud Computing, London, Springer.

[3] Di Niu, Hong Xu, Baochun Li Shuqiao Zhao.(2012). "Qualiti-Assured Cloud Bandwidth Auto-Scaling for Video-on-Demand Appalications". University of Toronto

[4] Filippo Lorenzo Ferraris, Davide Franceschelli, Mario Pio Gioiosa.(2012)." Evaluating the Auto Scaling Performance of Flexiscale and Amazon EC2 Clouds". Politecnico di Milano

[5] Gregor von Laszewski, Javier Diaz, Fugang Wang, Geoffrey C. Fox. (2012). "Comparison of Multiple Cloud Frameworks". Indiana University, Bloomington

[6] Jackson, Kevin, 2012, OpenStack Cloud Computing Cookbook, Birmingham, PACKT Publishing

[7] L´ılia Rodrigues Sampaio, Raquel Vigolvino Lopes.2012. "Towards practical auto scaling of user facing applications". Universidade Federal de Campina Grande, Paraiba

[8] Maarif, M.S. 2003. Teknik-Teknik Kuantitatif Untuk Manajemen. Grasindo: Jakarta

[9] Ming Mao, Marty Humphrey.(2011)." Auto-Scaling to Minimize Cost and Meet Application Deadlines in Cloud Workflows". University of Virginia

[10] Nilabja Roy, Abhishek Dubey, Aniruddha Gokhale.(2011)." Efficient Autoscaling in the Cloud using Predictive Models for Workload Forecasting". Vanderbilt University

[11] Openstack, Openstack Compute Administration Guide.(2013) [Online] tersedia di : http://www.docs.openstack.org/trunk/openstack-compute/admin/content/index.html

[12] Rajkummer Buyya, James Broberg, Andrzej Goscinski,2011, Cloud Computing Priciples and Paradigms, United State, Wiley.

[13] Santiko Bayu. (2012) ."Auto-scaling Cloud Computing Untuk Layanan VoIP Pada IP Multimedia Subsystem". Institut Teknologi Telkom, Bandung

[14] Tania Lorido-Botrano, Jose Miguel-Alonso, Jose A.Lozano. (2012) ." Auto-scaling Techniques for Elastic Applications in Cloud Environments". University of the Basque Country

[15] University of Guelph. 2013. Forecasting. http://www.uoguelph.ca/~dsparlin/forecast.htm#CAUSAL FORECASTING METHODS

[16] OpenStack Folsom Architecture Cloud Computing. 2012. http://ken.pepple.info/openstack/2012/09/25/openstack-folsom-architecture/

Biodata Penulis

Praditya Wahyu W, memperoleh gelar Ahli Madya (A.Md), pada Jurusan Teknik Informatika Institut Teknologi Telkom, lulus tahun 2012. Saat ini penulis sedang melanjutkan pendidikan lanjutan untuk mendapatkan gelar Sarjana Teknik di Universitas Telkom.

Novian Anggis Suwastika, memperoleh gelar Sarjana Teknik (S.T), Jurusan Teknik Informatika Institut Teknologi Telkom, lulus tahun 2009. Memperoleh gelar Magister Teknik (M.T) Program Pasca Sarjana Magister Informatika dari Institut Teknologi Bandung, lulus tahun 2012. Saat ini penulis menjadi Dosen Fakultas Teknik di Universitas Telkom Bandung.

ISSN : 2302-3805