Terbit online pada laman web jurnal : **http://jurnal.iaii.or.id**

JURNAL RESTI

(Rekayasa Sistem dan Teknologi Informasi)

Vol. 1 No. 3 (2017) 183 – 190 ISSN Media Elektronik : 2580-0760

Pengembangan Aplikasi Antarmuka Layanan *Big Data Analysis*

Gede Karya[a], Veronica S. Moertini[b]

[ab]Program Studi Teknik Informatika, Fakultas Teknologi Informasi dan Sains, Universitas Katolik Parahyangan,

[a]gkarya@unpar.ac.id, [b]moertini@unpar.ac.id

Abstract

In the 2016 Higher Competitive Grants Research (Hibah Bersaing Dikti), we have been successfully developed models, infrastructure and modules of Hadoop-based big data analysis application. It has also successfully developed a virtual private network (VPN) network that allows integration and access to the infrastructure from outside the FTIS Computer Lab. Infrastructure and application modules of analysis are then wanted to be presented as services to small and medium enterprises (SMEs) in Indonesia. This research aims to develop application of big data analysis service interface integrated with Hadoop-Cluster. The research begins with finding appropriate methods and techniques for scheduling jobs, calling for ready-made Java Map-Reduce (MR) application modules, and techniques for tunneling input / output and meta-data construction of service request (input) and service output. The above methods and techniques are then developed into a web-based service application, as well as an executable module that runs on Java and J2EE based programming environment and can access Hadoop-Cluster in the FTIS Computer Lab. The resulting application can be accessed by the public through the site http://bigdata.unpar.ac.id. Based on the test results, the application has functioned well in accordance with the specifications and can be used to perform big data analysis.

Keywords: web based service, big data analysis, Hadoop, J2EE

Abstrak

Pada penelitian Hibah Bersaing Dikti tahun 2016 telah berhasil dikembangkan model, infrastruktur dan modul-modul aplikasi *big data analysis* berbasis Hadoop. Selain itu juga telah berhasil dikembangkan jaringan *virtual private network* (VPN) yang memungkinkan integrasi dan akses infrastruktur tersebut dari luar Lab Komputer FTIS. Infrastruktur dan modul aplikasi analisis tersebut selanjutnya ingin dipresentasikan sebagai layanan kepada usaha kecil dan menengah (UKM) di Indonesia. Penelitian ini bertujuan untuk mengembangkan aplikasi antarmuka layanan *big data analysis* yang terintegrasi dengan Hadoop-Cluster. Penelitian diawali dengan mencari metode dan teknik yang tepat untuk menjadwalkan *job*, memanggil (*call*) modul aplikasi Java Map-Reduce (MR) yang sudah jadi, dan teknik untuk *input/ output tunneling* serta konstruksi *meta data* permintaan layanan (*input*) dan hasil layanan (*output*). Metode dan teknik di atas, kemudian dikembangkan menjadi aplikasi layanan berbasis web, serta modul eksekutor yang berjalan di atas lingkungan pemrograman berbasis *Java 2 Enterprise Edition* (J2EE) dan dapat mengakses Hadoop-Cluster di Lab Komputer FTIS. Aplikasi yang dihasilkan dapat diakses oleh publik melalui situs http://bigdata.unpar.ac.id. Berdasarkan hasil pengujian, aplikasi telah berfungsi dengan baik sesuai dengan spesifikasi dan dapat digunakan untuk melakukan *big data analysis*.

Kata kunci: layanan berbasis web, *big data analysis*, Hadoop, J2EE

1. Pendahuluan

Pada penelitian Hibah Bersaing Dikti tahun 2016 [1] telah berhasil dikembangkan model, infrastruktur dan modul-modul aplikasi *big data analysis* (BDA) berbasis Hadoop. Modul-modul aplikasi dikembangkan berbasis Java dalam bentuk modul-modul aplikasi Map-Reduce (MR). Infrastruktur BDA telah dikembangkan dalam bentuk Hadoop-Cluster dengan kapasitas sampai 20 *node* di Lab Komputer FTIS (LabKom-FTIS). Selain itu juga telah berhasil dikembangkan jaringan *virtual private network* (VPN) yang memungkinkan integrasi dan akses infrastruktur tersebut dari luar LabKom-FTIS. Infrastruktur dan modul aplikasi analisis tersebut selanjutnya ingin dipresentasikan sebagai layanan kepada usaha kecil dan menengah (UKM) di Indonesia. Masalah utama yang dihadapi adalah: (1) Ketersediaan LabKom-FTIS dalam menyediakan akses BDA pada jam 19.00 – 06.00 WIB, memanfaatkan *idle time* di mana LabKom-FTIS tidak digunakan untuk kegiatan perkuliahan mahasiswa. Sementara ada harapan untuk menyediakan antarmuka layanan *full time* kepada UKM; (2) Modul-modul

aplikasi MR dieksekusi secara *batch* pada infrastruktur Hadoop-Cluster. Jika akses melalui web oleh UKM maka eksekusi harus didelegasikan kepada suatu agen pada saat Hadoop-Cluster tersedia.

Oleh karena itu, penyediaan presentasi layanan BDA, selanjutnya disebut LBDA, kepada UKM memerlukan antarmuka aplikasi yang bertugas untuk menerima permintaan layanan, menjadwalkan permintaan tersebut (dalam rentang layanan LabKomp-FTIS), mengeksekusi permintaan, dan mengelola hasil eksekusi untuk selanjutnya diberikan kembali kepada UKM yang meminta layanan tersebut.

Penelitian ini bertujuan untuk menyiapkan metode dan teknik serta aplikasi sebagai komponen utama dari antarmuka LBDA. Lebih rinci tujuan dirumuskan sebagai berikut: (1) menemukan metode dan teknik untuk menjadwalkan *job* yang sesuai; (2) menemukan metode dan teknik untuk mengeksekusi *job* MR dari aplikasi berbasis J2EE; (3) menemukan metode dan teknik untuk *input/ output tunneling job* MR dari aplikasi berbasis J2EE; (4) mengembangkan aplikasi berbasis web J2EE untuk mengimplementasikan ketiga metode dan teknik di atas.

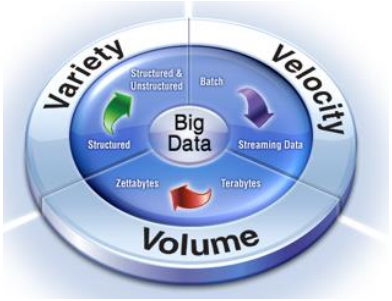
Atas dasar tujuan tersebut maka rumusan masalah dalam penelitian ini diformulasikan dalam bentuk 3 pertanyaan berikut: (1) metode dan teknik penjadwalan *job* apa yang cocok untuk kasus di atas? (2) metode dan teknik apa yang cocok untuk mengeksekusi (*call*) job MR pada lingkungan J2EE? (3) metode dan teknik apa yang cocok untuk *input/output tunneling job* MR pada lingkungan J2EE?

2. Tinjauan Pustaka/Penelitian Sebelumnya

Pada bagian ini dijelaskan tentang tinjauan pustaka, yang mencakup: *big data* dan *big data analysis*, J2EE. Kemudian dibahas penelitian sebelumnya yang berhubungan dengan penelitian ini.

2.1 *Big Data* dan *Big Data Analysis*

Big data didefinisikan sebagai data yang memiliki karakteristik 3V [2], yaitu: *Volume* yang besar, *Velocity* yang cepat dan *Variety* yang banyak, baik sumber maupun formatnya.

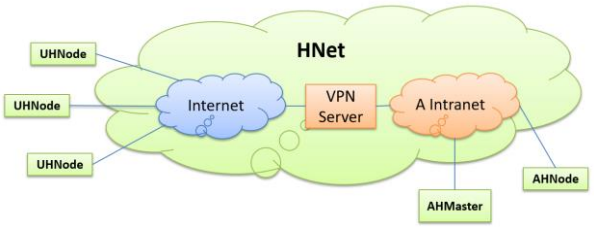


Gambar 1 Karakteristik *Big Data*

Big data memiliki *value* berupa pola/ pengetahuan bagi organisasi yang dapat diekstrak melalui proses *big data analysis*. Proses BDA termasuk dalam konteks *knowledge data discovery* [3]. Untuk mengelola *big data* tidak dapat menggunakan teknologi pemrosesan konvensional. Oleh karena itu, telah banyak dikembangkan teknologi untuk memproses data dengan karakteristik *big data* tersebut. Salah satu teknologi yang secara *defacto* digunakan saat ini untuk menangani *big data* adalah Hadoop [4]. Hadoop merupakan *framework* pengelolaan *big data* yang menyediakan *storage* (*Hadoop Distributed File System* – HDFS) dan pemrosesan terdistribusi (Map-Reduce - MR). Dengan menggunakan atau memanfaatkan *platform* Hadoop, kita dapat mengembangkan aplikasi untuk pengelolaan dan BDA berbasis MR. Selain itu, kita juga dapat memanfaatkan aplikasi-aplikasi yang telah di buat di atas platform Hadoop yang disebut sebagai Hadoop Ecosystem, seperti: Hadoop database (Hbase), Hive (*data warehouse*) dan yang lainnya.

2.2 Penelitian Terkait Yang Telah Dilaksanakan

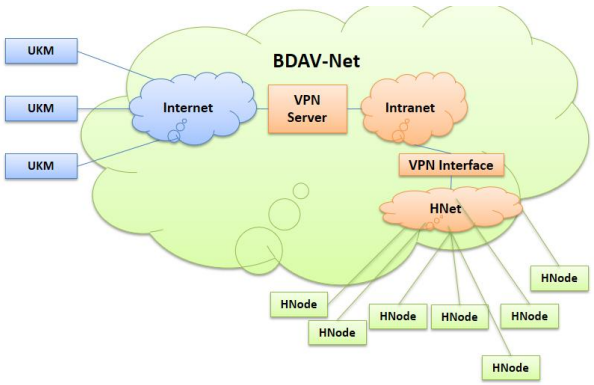
Pada [1] telah dihasilkan infrastruktur operasional *big data* seperti pada Gambar 2. Model ini dikembangkan dari hasil penelitian yang untuk Pembukuan UMK yang dipublikasikan pada [6]. Model ini juga telah dipublikasikan pada [7].



Gambar 2 Arsitektur Infrastruktur Operasional Big Data [1]

Pada Gambar 2, **HNet** adalah jaringan virtual Hadoop yang dibangun bersama-sama oleh UKM, misalnya melalui paguyuban/ asosiasi terkait. **HNet** memanfaatkan jaringan Internet dan intranet penyelenggara (misalnya asosiasi) **A Intranet**. Sebagai infrastruktur dasar asosiasi A menyediakan server *dedicated* **AHMaster** sebagai Hadoop master (berisi master node HDFS, HMaster HBase, Zookeeper, Hive dan sejenisnya), dan beberapa Hadoop *slave* **AHNode** (Data Node, HRegionServer HBase, Map Reduce Job Tracker, dan sejenisnya). Sedangkan untuk setiap UKM yang tergabung menyediakan server *slave* **UHNode** dengan isi sama dengan **AHNode** ditambah dengan aplikasi untuk mengaksesnya dari masing-masing UKM (HDFS Client, HBase CLI, Hive client, dan sejenisnya. Arsitektur ini telah diimplementasikan dan diujicoba di LabKom-FTIS Unpar.

Selain itu juga telah dikembangkan arsitektur infrastruktur untuk BDA untuk UKM, seperti pada Gambar 3.



Gambar 3 Arsitektur Infrastruktur untuk BDA [1]

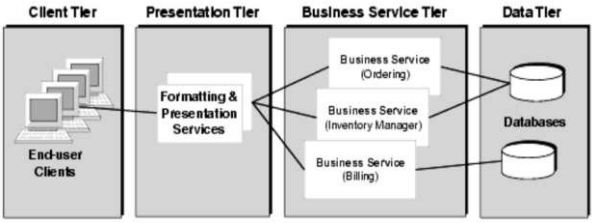
Pada Gambar 3, dapat dilihat bahwa jaringan virtual *Big Data Analytical Virtual Network* (**BDAV-Net**). Pada jaringan tersebut terkoneksi node **UKM** (server UKM untuk keperluan analisis data/ operasional) melalui internet. VPN Server merupakan server yang disediakan untuk melayani koneksi VPN dari internet ke jaringan Hadoop (**HNet**). Intranet merupakan jaringan internal suatu organisasi, misalkan: Unpar. HNet merupakan jaringan *cluster* Hadoop. Pada HNet terkoneksi node-node Hadoop (**HNode**), yang menjalankan layanan HDFS, Map Reduce (Yarn), HBase, Hive, Sqoop, dan ekosistem Hadoop yang disediakan. Dalam hal ini contoh HNet adalah jaringan LabKom-FTIS, di mana HNode adalah komputer-komputer yang ada di LabKom-FTIS yang digunakan untuk layanan di luar jam kerja. **VPN Interface** merupakan node komputer sebagai *end point* koneksi ke VPN Server.

Pada saat layanan Hadoop-Cluster (**HNet**) dihidupkan, maka akan terhubung ke VPN Server melalui VPN Interface. Dengan demikian UKM yang ingin memanfaatkan layanan ini dapat mengakses HNet melalui koneksi ke VPN Server, sehingga aksesnya tetap *dedicated* dan aman melalui jaringan publik (internet) dalam bentuk **BDAV-Net**. Model arsitektur ini sudah diimplementasikan dan diujicoba di LabKom-FTIS.

Pada infrastruktur Gambar 3 juga telah dikembangkan dan diuji modul-modul aplikasi Java MR untuk BDA, antara lain: (1) Modul-modul program pada Sistem BI, seperti Modul *Streaming* Twitter, Modul *Instagram Crawler,* Modul Counter Objek Wisata (untuk analisis data Twitter & Instagram dengan teknik statistic sederhana) dan Modul Analisis Log Web dari Apache Web Server; (2) Modul program untuk pengelompokan *big data*, seperti: modul Enhanced k-Means Paralel Berbasis MapReduce. Hasil pengembangan aplikasi tersebut telah dipublikasikan pada [8] dan [9]. Aplikasi-aplikasi inilah yang nantinya menjadi bagian dari layanan BDA yang dipresentasikan kepada UKM.

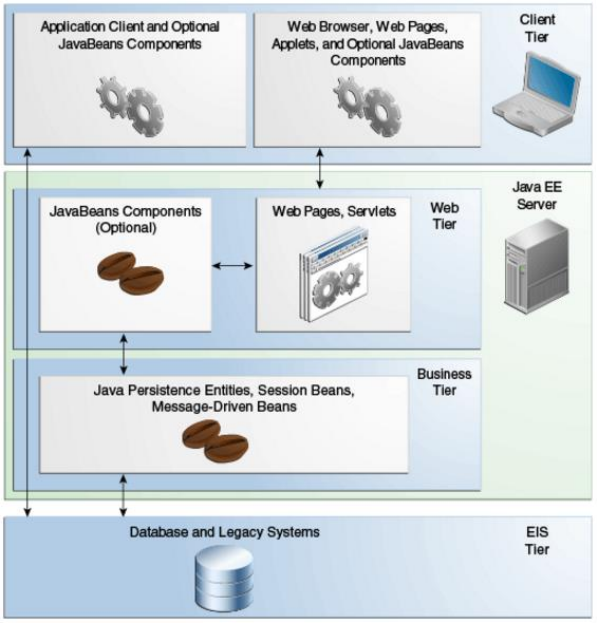
2.3 Lingkungan *Java 2 Enterprice Edition* (J2EE)

J2EE merupakan platform Java untuk tingkat enterprise. Platform ini cocok untuk mengimplementasikan aplikasi berbasis web dan *multi tier architecture*. Arsitektur J2EE dapat dilihat pada Gambar 4.[5]



Gambar 4 Arsitektur Multi Tier J2EE [5]

Komponen J2EE terbagi dalam 4 *tier*, yaitu: *client tier, presentation tier, business service tier* dan *data tier*. Dari sisi teknologi implementasi, masing-masing *tier* dapat dilihat pada Gambar 5.



Gambar 5 Arsitektur Implementasi J2EE [5]

Pada Gambar 5, dapat dilihat bahwa pengembangan aplikasi web dilakukan pada *Web Tier*, berupa *web page* dan *servlet*. Selain itu juga baik *web page* maupun *servlet* dapat mengakses komponen-komponen *Java Bean* dalam bentuk kelas-kelas Java yang telah dibuat, dan dapat juga mengakses kelas-kelas Java pada *Business Tier*. Kelas-kelas Java pada *Business Tier* dapat digunakan untuk mengakses sistem lain, termasuk basis data menggunakan *Java Data Base Connectivity* (JDBC).

2.2 *Job Scheduler*

Job scheduler merupakan aplikasi sistem/ utilitas untuk menjadwalkan eksekusi suatu aplikasi pada waktu tertentu. Elemen waktu yang dimaksud adalah pada 1 waktu tertentu, atau perulangan setiap periode waktu tertentu, atau perulangan setiap menit, jam atau hari tertentu dalam suatu periode jam, hari, minggu, bulan, tahun. Ada beberapa alternatif penggunaan job scheduler, yaitu: (1) level sistem operasi, yang disediakan oleh sistem operasi, seperti: CRON pada sistem operasi Linux, AT atau *Task Scheduler* pada sistem operasi Windows. (2) level platform, seperti pada J2EE ada aplikasi Quartz yang dapat digunakan untuk melakukan penjadwalan. (3) level aplikasi, yaitu dengan membuat sendiri aplikasi Java yang bertugas sebagai *scheduler*.

3. Metodologi Penelitian

Berdasarkan uraian di atas, penelitian ini diharapkan berkontribusi dalam menyediakan metode, teknik dan aplikasi dalam mengembangkan antarmuka aplikasi layanan *big data analysis*. Antarmuka diberikan terhadap infrastruktur seperti pada Gambar 2 dan 3 dan aplikasi-aplikasi yang telah dikembangkan pada [1]. Penelitian ini dilaksanakan dengan metodologi sebagai berikut: (1) studi pustaka dan eksplorasi tentang: hasil-hasil penelitian tahun 2016, kajian lingkungan pengembangan *Java 2 Enterprise Edition* (J2EE), khususnya yang berhubungan dengan pengembangan aplikasi web, *web service* dan penjadwalan pada lingkungan J2EE; (2) mengusulkan metode dan teknik serta algoritma untuk penjadwalan, eksekusi dan *input/output tunneling* sesuai permasalahan yang ditangani; (3) mengembangkan model arsitektur aplikasi layanan BDA yang terintegrasi dengan Hadoop-Cluster; (4) mengembangkan aplikasi sesuai arsitektur yang telah didefinisikan dalam bentuk aplikasi berbasis web pada lingkungan J2EE dan aplikasi manajemen eksekutor berbasis J2SE; (5) melaksanakan pengujian dan eksperimen, dengan beberapa kasus *job* sederhana sesuai dengan hasil penelitian [1].

4. Hasil dan Pembahasan

Hasil dari penelitian ini dikelompokkan menjadi 5 topik, yaitu: (1) metode dan teknik penjadwalan *job*; (2) metode dan teknik eksekusi dan *input/ output tunneling job*; (3) arsitektur aplikasi LBDA yang terintegrasi dengan Hadoop-Cluster; (4) *web site* LBDA dan contoh hasil eksperimen menggunakan LBDA.

4.1 Metode dan Teknik Penjadwalan Job

Komponen penjadwalan sangat penting pada LBDA yang dikembangkan, karena beberapa alasan:

1. Hadoop-Cluster sebagai infrastruktur eksekutor utama yang menjalankan layanan ini tidak tersedia sepanjang waktu, melainkan hanya tersedia pada jam 19.00 wib sampai dengan 06.00 wib. Hal ini terjadi karena memanfaatkan jadwal kosong dari LabKom-FTIS pada malam hari.
2. Aplikasi layanan dapat diakses secara *online* sepanjang waktu.
3. Oleh karena itu, pesanan analisis dapat dibuat kapan saja, namun eksekusinya tetap mengikuti jadwal ketersediaan infrastruktur eksekutor. Oleh karena itu, diperlukan agen eksekutor dan penjadwalan *job*.

Metode penjadwalan yang diusulkan adalah *first in first serve* (FIFO) dengan agen eksekutor tunggal. Agen eksekutor tunggal digunakan dengan pertimbangan agar setiap *job* dapat menggunakan Hadoop-Cluster secara ekslusif, sehingga dapat diukur performansinya dengan jelas (waktu *start-time* dan *finish-time* jelas). Namun demikian, untuk menghindari *job* yang berjalan tidak normal (menggantung) sehingga tidak selesai-selesai, maka diberi waktu maksimum eksekusi *job* berupa parameter *execTimeOut* dalam satuan *micro second*. Selain itu, untuk setiap penyelesaian job, Hadoop-Cluster diberikan kesempatan untuk memulihkan diri (*flushing file*, dan *background processes* lainnya) dengan memberikan jeda waktu selama 10 detik (10.000 *micro second*) sebelum mengeksekusi *job* selanjutnya. Agar mendapatkan jaminan bahwa *job* yang diambil dapat dieksekusi, maka pengecekan terhadap status Hadoop-Cluster selalu dilakukan sebelum mengambil *job* dari local Hbase server. Algoritma penjadwalan job dapat dilihat pada Gambar 6.

Algoritma 1 Penjadwalan Job

```
// input: hdfsURL, hbaseServer, hbasePort, execTimeOut
01 hdfsURL = Params(1); // read hdfsURL from input params
02 hbaseServer = Params(2); // read hbaseServer host from input params
03 hbasePort = Params(3); // read hbasePort from input params
04 execTimeOut  = Params(4) // read execTimeOut from input params
05 While (job=GetIdleJob() ) do // ambil job dari antrean
06      ExecuteJob(job, execTimeOut); // eksekusi job
07      Idle (10000); // idle time 10 second
08 End while;
```

Gambar 6 Algoritma Penjadwalan Job

Pada Gambar 6, Algoritma 1 dieksekusi sebagai *daemon* (*service*) pada sistem operasi Linux. Dengan demikian akan terus bekerja berulang sepanjang waktu.

4.2 Metode dan Teknik Eksekusi dan *Input/ Output Tunneling Job*

Eksekusi MR *job* dilakukan dengan metode execute *command line* menggunakan method *Runtime.getRuntime().exec(command)* pada Java *Application Programming Interface* (API). Setiap

eksekutor adalah agen. Parameter *input* dan *output* di-*tunnel* sebagai parameter *command line*. Dengan demikian algoritma untuk eksekusi job *(ExecuteJob(Job, execTimeOut))* dapat dilihat pada Gambar 7.

Algoritma 2 Eksekusi Job

```
// Parameter masukan adalah job, dan execTimeOut
01 GetJobAtributes(job); // Baca atribut job dari basis data
02 If (job.state==idle)  // Cek sekali lagi apakah job.state=idle
// Eksekusi job
03      process = Exec(job.command, job.inputParams, job.outputParams);
04      job.SetState(running); // update state=running pada table Job
// tunggu proses sampai selesai, atau execTimeOut tercapai
05      success = Wait (process, execTimeOut);
// ambil hasil proses (std_error, std_out dan file sesuai parameter output)
06      If (success)
07          job.SetState(finish); // update state pada table Job
// Ambil hasil dari HDFS file path job.outputParams,
// lalu pindahkan ke tabel JobResult
08          job.SaveResult(job.outputParams);
09      Else
10          job.SetState(error); // update state=error pada table Job
11      End If;
// Ambil pesan dari std_error, dan simpan pada tabel JobResult
12      job.SaveError();
// Ambil pesan dari std_out, dan simpan pada tabel JobResult
13      job.SaveOutput();
14 End if;
```

Gambar 7 Algoritma Eksekusi Job

Perhatikan kembali Algoritma 2 pada Gambar 7, *input tunneling* dilakukan dengan membaca parameter input melalui method *GetJobAttribute(job),* kemudian mengirimkan input tersebut kepada proses dengan pemanggilan *Exec(job.command, job.inputParams, job.outputParam).* Sedangkan pengambilan hasil (*output tunneling*) dilakukan melalui method *job.SaveResult(),* yang men-*download* hasil HDFS dengan file *path=job.outputParams* kemudian menyimpannya ke tabel *JobResult* pada *field result*. Untuk me-*record* semua pesan yang tampil di layar (`std_out`) dan pesan error (`std_error`), maka digunakan method *job.SaveOutput()* dan *job.SaveError().* Dengan demikian setiap *job* yang dieksekusi akan ter-*record* seluruh *output*-nya.

Khusus untuk *job.inputParams* dapat berupa parameter *input* program, atau *file path* pada HDFS sebagai file *input*. Sedangkan *job.outputParams* adalah nama file hasil proses pada HDFS.

Berikut ini adalah contoh kasus *big data* klasifikasi. Jika dijalankan pada terminal Linux, maka perintahnya sebagai berikut:

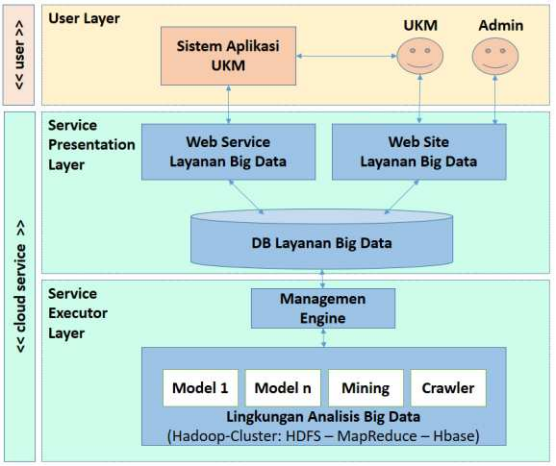
```
> Hadoop jar testing.jar
   /klasifikasi/testing/inp1
   /klasifikasi/testing/out1
   /klasifikasi/training/out1
   application.properties meta.info
```

Penjelasan dari *command line* di atas adalah sebagai berikut: hadoop jar testing.jar untuk menjalankan file map reduce testing.jar menggunakan hadoop; /klasifikasi/testing/inp1 adalah HDFS *file path* tempat file input yang diproses oleh modul testing.jar; /klasifikasi/testing/out1 adalah HDFS *file path* tempat menyimpan hasil dari proses klasifikasi; /klasifikasi/training/out1 adalah HDFS *file path* tempat menyimpan hasil pembelajaran dari modul training.jar yang digunakan sebagai basis pengetahuan pada modul testing.jar ini; application.properties adalah *local file path* tempat propertis dari aplikasi testing.jar; dan meta.info adalah *local file path* tempat meta informasi yang digunakan oleh aplikasi testing.jar.

Command line di atas disimpan pada job sebagai berikut: *job.command* ← hadoop jar /home/hduser/app /klasifikasi/testing.jar; *job.inputParams* ← /klasifikasi/ testing/inp1 /klasifikasi/testing/out1 /klasifikasi/training/out1 /home/hduser/app/klasifikasi/application.properties /home/ hduser/app/klasifikasi/meta.info; dan *job.outputParams* ← /klasifikasi/testing/out1.

4.3 Arsitektur Aplikasi LBDA Terintegrasi

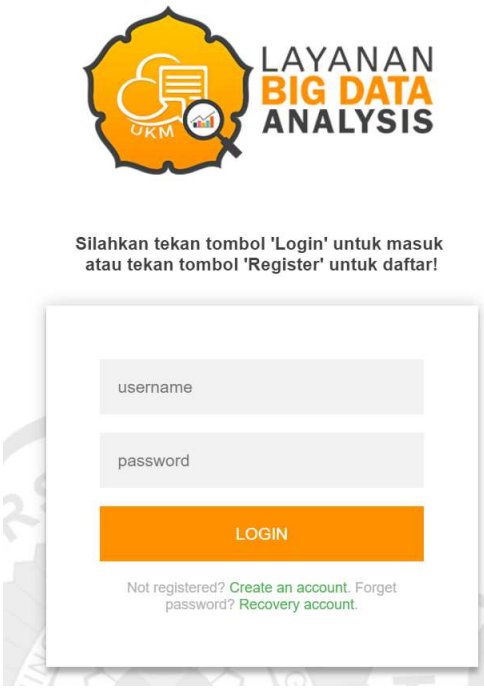
Arsitektur aplikasi LBDA yang terintegrasi dapat dilihat pada Gambar 8. Arsitektur LBDA tediri atas 2 sisi, yiatu sisi *User* dan sisi *Cloud Service*. Pada sisi user, terdiri atas 1 *layer*, yaitu User *layer* merupakan pihak yang berperan sebagai pengguna yang akan mengkonsumsi atau mempekerjakan layanan yang disediakan. Sedangkan di sisi *Cloud Service* terdiri atas 2 *layer*, yaitu: (1) *Service Presentation Layer*, yang bertugas menyediakan antarmuka layanan kepada pengguna, dan menyimpan *job order* pada basis data Layanan Big Data. Antarmuka disediakan dalam 2 mode, yaitu: (1.a) *web service* untuk melayani aplikasi lain, dan (1.b) *web site* untuk melayani *end user*; (2) *Service Executor Layer* bertugas untuk mengeksekusi *job* dengan mempekerjakan aplikasi MR yang telah tersedia pada lingkungan BDA pada Hadoop-Cluster.



Gambar 8 Arsitektur Aplikasi LBDA

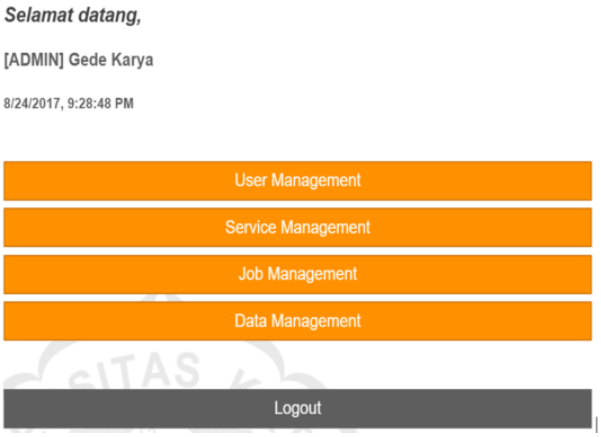
4.4 *Web Site* LBDA

Web Site LBDA (Web-LBDA) melayani *user* UKM dan *user* Admin. Validasi *user* menggunakan mekanisme LogIn menggunakan userId dan passWord. Untuk mengkhiri *session* menggunakan LogOut (lihat Gambar 9).



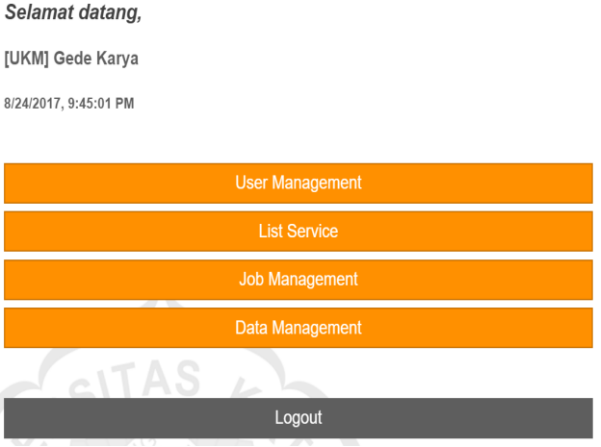
Gambar 9 Halaman Awal Web-LBDA

User Admin merupakan pengelola situs Web-LBDA. Fitur Web-LBDA yang disediakan untuk Admin (Gambar 10), antara lain: (1) *User Management*, berupa fasilitas untuk mengelola *user* aplikasi UKM dan *end user* UKM, antar lain: menampilkan daftar *user*, menampilkan detail *user*, aktifasi dan deaktifasi user; (2) *Service Management,* berupa fasilitas untuk mengelola layanan yang disediakan, antara lain: menampilkan daftar layanan, memasukkan layanan baru, aktifasi dan deaktifasi layanan baru, memonitor status layanan (berjalan atau tidak); (3) *Job Management,* berupa fasilitas untuk mengelola *job* yang dipesan oleh *user*. Fitur-fitur yang disediakan antara lain: menampilkan daftar job order (dengan filter semua, per *user*, per layanan, per status), memonitor status setiap *job* (*idle, running, finish, error*); (4) *Data Management,* berupa fasilitas untuk mengelola data yang disimpan pada HDFS yang dapat diakses oleh user. Fitur-fitur yang disediakan antara lain: menampilkan daftar data (dengan filter semua, per user, per status), memonitor status setiap data (HDFS, web, URL link), menambahkan data yang sudah tersedia di HDFS yang dapat digunakan oleh semua user.



Gambar 10 Halaman Web-LBDA untuk Admin

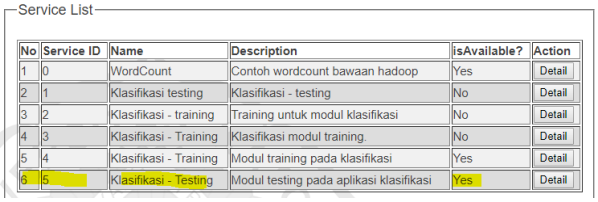
Web-LBDA juga menyediakan layanan untuk user UKM (lihat Gambar 11), antara lain: (1) *User Management,* untuk mengelola data user. Fitur yang disediakan antara lain: (a) *Registration,* mendaftarkan diri dengan mengisi form pendaftaran. Pendaftaran dikonfirmasi via email untuk proses aktivasi mandiri. Masa retensi 1 hari; (b) *Forget Password*, dengan memasukkan alamat email, maka urlLink ganti password tanpa otentifikasi akan dikirimkan, dengan masa retensi selama 1 hari; (c) *Update Profile,* untuk mengupdate deskripsi, email dan urlLink; (d) *Change Password,* untuk mengganti password; (e) *Unregister,* mengundurkan diri dari keanggotaan, dengan konsekuensi penonatifan, kemudian penghapusan semua data yang dimiliki. Sebelum efektif dikonfirmasi dulu via email, baik ke user yang bersangkutan maupun Admin. Masa retensi untuk unregister adalah 1 minggu;



Gambar 11 Halaman Web-LBDA untuk UKM

(2) *List Service,* untuk menampilkan daftar layanan yang disediakan oleh Web-LBDA (lihat Gambar 12). Untuk setiap layanan dapat dilihat propertinya lebih detail; (3) *Data Management,* untuk mengelola data setiap user. Fitur-fitur yang disediakan antara lain: (a) *List Data,* untuk menampilkan daftar data yang dimiliki, beserta statusnya, apakah sudah ada di HDFS

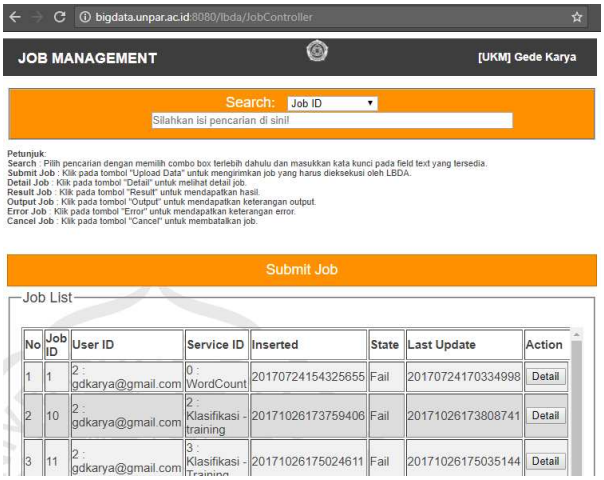
atau masih berada pada link eksternal atau tersimpan pada situs web; (b) *Upload Data*, untuk memasukkan data yang akan dianalisis. Upload data hanya dapat dilakukan pada saat Hadoop-Cluster online; (c) *Delete Data,* untuk menghapus data dari HDFS. Untuk data yang telah ada di HDFS, maka akan diberlakukan sebagai job yang akan dieksekusi pada saat Hadoop Cluster tersedia; (d) *Share Data,* untuk menshare data yang telah tersimpan di HDFS kepada user lain. Jika hal ini dilakukan, maka data yang di-share juga akan muncul pada List Data user penerima;


Service List

No	Service ID	Name	Description	isAvailable?	Action
1	0	WordCount	Contoh wordcount bawaan hadoop	Yes	Detail
2	1	Klasifikasi testing	Klasifikasi - testing	No	Detail
3	2	Klasifikasi - training	Training untuk modul klasifikasi	No	Detail
4	3	Klasifikasi - Training	Klasifikasi modul training.	No	Detail
5	4	Klasifikasi - Training	Modul training pada klasifikasi	Yes	Detail
6	5	Klasifikasi - Testing	Modul testing pada aplikasi klasifikasi	Yes	Detail

Gambar 12 Halaman Web-LBDA fitur Service List

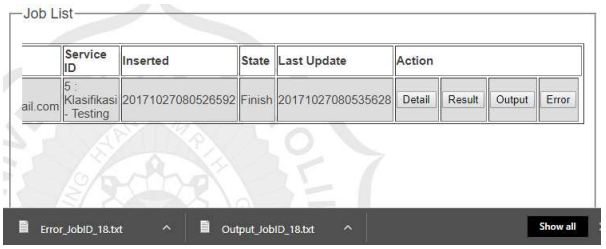
(4) *Job Management,* untuk mengelola job setiap user (lihat Gambar 13). Fitur-fitur yang disediakan antara lain: (a) *submit Job Order*, untuk mengirimkan job order yang harus dieksekusi oleh LBDA. Job order yang dapat di-*submit* sesuai dengan daftar layanan yang disediakan oleh Web-LBDA; (b) *Job Order Status Monitor*, untuk meminta status terkini dari Job Order yang telah di-submit. Job Order yang dapat dimintakan statusnya hanya Job Order yang di-submit oleh user UKM yang meminta. Status Job Order adalah *idle* (belum dieksekusi), *running* (sedang dieksekusi), *finish* (sudah selesai dieksekusi, sehingga ada hasilnya), *error* (sudah dieksekusi namun terhenti karena ada masalah/ *error*, keterangan *error* dapat diambil);



No	Job ID	User ID	Service ID	Inserted	State	Last Update	Action
1	1	2 : gdkarya@gmail.com	0 : WordCount	20170724154325655	Fail	20170724170334998	Detail
2	10	2 : gdkarya@gmail.com	2 : Klasifikasi - training	20171026173759406	Fail	20171026173808741	Detail
3	11	2 : gdkarya@gmail.com	3 : Klasifikasi - Training	20171026175024611	Fail	20171026175035144	Detail

Gambar 13 Halaman Web-LBDA *Job Management*

(c) *Get Job Result*, (lihat Gambar 14) untuk mengakses/ men-*download* hasil eksekusi dari *job* yang telah berstatus finish. Hanya *job* yang dimiliki oleh User UKM tertentu saja yang dapat di ambil hasilnya; (d) *Cancel Job Order*, untuk membatalkan eksekusi atas Job Order tertentu. Job yang dapat dibatalkan hanya yang berstatus: *idle* saja.


Job List

	Service ID	Inserted	State	Last Update	Action			
ail.com	5 : Klasifikasi - Testing	20171027080526592	Finish	20171027080535628	Detail	Result	Output	Error

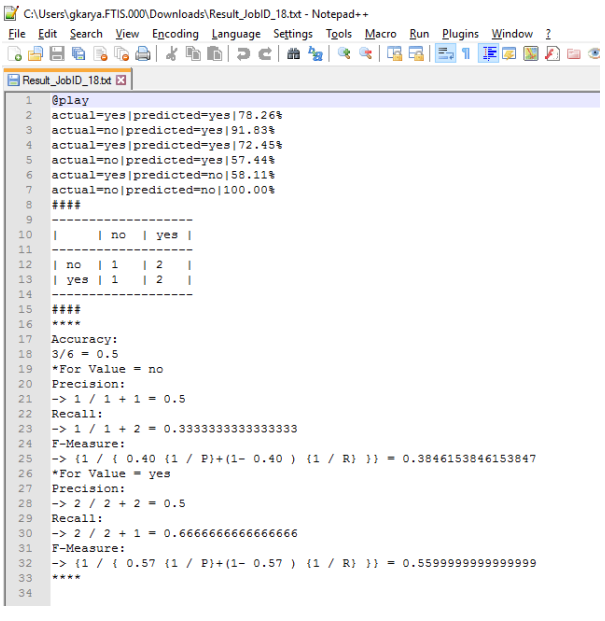
Error_JobID_18.txt Output_JobID_18.txt Show all

Gambar 14 Halaman Web-LBDA fitur Job Result Monitor

Pada Gambar 14 dapat dilihat bahwa, untuk setiap *job* yang didaftarkan dapat dipantau *service* yang dijalankan (ServiceID), statusnya (State), kapak didaftarkan (Inserted), dan kapan status terakhir di-*update* (Last Update). Selain itu juga dapat dilihat informasi detail dari *job* ini (Detail) beserta hasil eksekusinya. Pada Gambar 14 tersebut status job adalah Finish berarti sudah selesai dikerjakan dengan baik (berhasil). Oleh karena itu, jika kita ingin melihat hasilnya dapat meng-klik tombol Result. Jika ingin melihat hasil capture layar output dapat meng-klik tombol Output. Sedangkan jika ingin melihat adanya pesan *error* selama proses eksekusi, dapat meng-klik tombol Error.

Contoh Result dapat dilihat pada Gambar 15 dari hasil suatu percobaan (eksperimen).

Percobaan/ eksperimen BDA telah dilaksanakan sebagai uji coba pada LBDA. Pada Gambar 15 dapat dilihat salah satu contoh hasil eksperimen BDA yang telah dilakukan menggunakan LBDA tersebut.



Gambar 15 Contoh Hasil Eksperimen Menggunakan LBDA

Khusus untuk modul *web service* (WS) dan modul *engine management* (ME) tidak dijelaskan pada makalah ini. Penjelasan detailnya dapat dilihat di [10].

5. Kesimpulan

Berikut adalah kesimpulan dan pengembangan selanjutnya dari uraian di atas.

5.1 Kesimpulan

Berdasarkan hasil dan pembahasan pada bagian 4, dapat disimpulkan hal-hal sebagai berikut: (1) Metode yang diterapkan untuk penjadwalan *job* telah dirumuskan dalam bentuk algoritma penjadwalan dengan eksekutor tunggal. Penggunaan eksekutor tunggal didasari pertimbangan agar setiap *job* dapat mengakses Hadoop-Cluster secara eksklusif sehingga kinerjanya maksimal dan terukur baik. Metode ini diimplementasikan sebagai program aplikasi yang berbasis Java pada lingkungan J2SE; (2) Metode eksekusi *job* berupa aplikasi berbasis Java menggunakan MR framework dipilih menggunakan eksekusi *external shell command* dengan Java API method *Runtime.getRuntime().exec(command),* di mana *command* adalah *shell command* dengan *path* absolut yang diturunkan dari *command line* saat melaksanakan percobaan BDA menggunakan Hadoop. Untuk keamanan dan kelancaran, setiap eksekusi dibatasi waktu *execTimeOut* yang ditentukan sebagai argumen/ parameter eksekusi agen; (3) *Input tunneling* ditambahkan pada *command (shell command)* sebagai parameter input sehingga sangat fleksibel. Parameter input dapat berupa perluasan perintah (*extended command line*) atau berupa nama file (*path*) sebagai file input; (4) *Output tunneling* didefinisikan sebagai atribut dari *job* yang selanjutnya di-*download* sebagai *output* dari HDFS untuk disimpan pada atribut *job* pada aplikasi web. Selain hasil dari eksekusi *job*, *output* yang disimpan juga berupa *capture* layar *output* (*std_out*) dan pesan *error* selama eksekusi (*std_error*) sehingga dapat mencerminkan seluruh *output* dari eksekusi *job*; (5) Seluruh metode di atas telah diintegrasikan dalam bentuk arsitektur aplikasi LBDA, yang terdiri atas modul *Web Site* (Web), *Web Service* (WS) dan aplikasi *Management Engine* (ME) sebagai eksekutor. Aplikasi tersebut mengakses basis data Hbase lokal dan mengakses Hadoop-Cluster secara *remote*; (6) Modul-modul aplikasi pada arsitektur di atas sudah berhasil diimplementasikan pada lingkungan berbasis Java, J2SE untuk modul ME dan J2EE untuk modul Web dan WS. Web site dapat diakses pada situs http://bigdata.unpar.ac.id; (7) Pengujian telah berhasil dilakukan baik fungsional maupun eksperimen penggunaan LBDA yang terintegrasi dengan Hadoop-Cluster.

5.2 Pengembangan Selanjutnya

Penelitian selanjutnya dilakukan untuk memantau kinerja dari LBDA setelah digunakan oleh UKM. Potensi lain adalah: arsitektur LBDA dapat digunakan juga sebagai fasilitas BDA untuk para peneliti.

Ucapan Terima Kasih

Ucapan terima kasih disampaikan kepada Lembaga Penelitian dan Pengabdian kepada Masyarakat, Universitas Katolik Parahyangan (LPPM-UNPAR) atas dukungan dan pendanaan penelitian ini melalui hibah penelitian internal dengan nomor kontrak III/LPPM/2017-01/14-P.

6. Daftar Rujukan

[1] Veronica S. Moertini, Gede Karya, 2016. Pengembangan Model Sistem Manajemen dan Teknik Analisis Big Data Bagi Komunitas UKM Indonesia, Laporan Akhir Penelitian Hibah Bersaing Dikti.
[2] D. Laney, 2001. 3D Data Management: Controlling Data Volume, Velocity, and Variety.
[3] S. Fosso Wamba, S. Akter, A. Edwards, G. Chopin, and D. Gnanzou, 2015. How 'big data' can make big impact: Findings from a systematic review and a longitudinal case study, Int. J. Prod. Econ., Vol. 165, pp. 234–246
[4] White T., 2012. Hadoop: The Definition Guide, 3rd Edition. O'Reilly.
[5] Oracle, Java 2 Enterprise Edition, online https://docs.oracle.com/javaee, tanggal akses 10 Desember 2016.
[6] Karya G, Veronica S M, 2017. Eksplorasi Teknologi Big Data Hadoop Untuk Sistem Aplikasi Berbasis Komunitas Studi Kasus: Aplikasi Pembukuan UMK, Jurnal RESTI vol 1 no 2 pp 160-169.
[7] Karya G, Veronica S M, 2017. Model Arsitektur dan aplikasi Manajemen Operasional Big Data untuk UMKM, Prosiding Seminar Nasional Aptikom (Semnastikom), pp 1-6.
[8] Veronica S M, Liptia V, 2017, Parallel K-Means For Big Data: On Enhancing Its Cluster Metrics And Patterns, Journal of Theoretical and Applied Information Technology, Vol.95. No 8 pp 1844-1856.
[9] Veronica S M, Vincensius K, Jeane S, 2017. Mining Opinions From Big Data Of Indonesian Hotel Reviews, Journal of Theoretical and Applied Information Technology, Vol.95. No 14 pp 3251-3259.