E-ISSN 2338 - 8641

ISSN 2303 - 3878

TEMATIKA

Jurnal Penelitian Teknik Informatika dan Sistem Informasi

Volume 8 Nomor 2, September 2020

FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ATMA JAYA MAKASSAR

TEMATIKA

Jurnal Penelitian Teknik Informatika dan Sistem Informasi

Alamat Redaksi:
Fakultas Teknologi Informasi
Universitas Atma Jaya Makassar
Jl. Tanjung Alang No. 23 Makassar 90224
Telp: (0411) 871038 / 871733 Fax: (0411) 870294
http://tematika.uajm.ac.id
tematika@fti.uajm.ac.id

Jurnal TEMATIKA merupakan jurnal ilmiah sebagai media pengembangan ilmu pada bidang Teknik Informatika dan Sistem informasi yang diterbitkan oleh Fakultas Teknologi Informasi Universitas Atma Jaya Makassar.

Jurnal TEMATIKA diterbitkan 2 (dua) kali dalam 1 (satu) tahun pada bulan **Maret** dan **September**

ISSN 2338 - 8641 ISSN 2303 - 3878

TEMATIKA

Jurnal Penelitian Teknik Informatika dan Sistem Informasi

Volume 8 Nomor 2, September 2020

DAFTAR ISI

PENINGKATAN MODULARITAS SISTEM TATA KELOLA PERKULIAHAN DENGAN PENDEKATAN MODEL VIEW CONTROLLER

Adi Chandra Syarif
Prodi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Atma Jaya Makassar
Alamat e-mail: adi_syarif@lecturer.uajm.ac.id

ABSTRACT

Based on PP No. 4 year 2014, college is a unit of organized higher education. In Providence, the College has the autonomy to manage their own institution comprising the system setting, planning, supervision, monitoring, and evaluation to achieve the goals of higher education. Atma Jaya University Makassar (UAJM) as one of the higher education has been harnessing information and communication technology infrastructure such as SIAMIK and other supporting systems. The Lecture Process Management System is a support system that has been developed. This system is part of the supervision system of the lecture at Atma Jaya University Makassar. Based on the results of the simulation test, it was found that the service interface of the Lecture Management System tablet PC was still difficult to use due to the coding structure carried out and the involvement of different teams. This study aims to update the data structure and interface of the java platform for the services of the Lecture System on the tablet PC system, update the audio and video data acquisition modules and engineer interaction services on the lecture digital attendance module. The development method used is the waterfall design approach with the UML method. System reengineering methods are used to understand the system and concept of the Model-View-Controller architecture that encapsulates data along with processing (model), isolates it from the manipulation process (controller) and display (view) to be represented on a user interface. The results of the research are that the system that has been reengineered and integrated with SIAMIK can help lecturers and chairman of the study program in planning, implementing and supervising student academic activities at FTI-UAJM.

Keywords: *Information Service, Model-View-Controller, Re-Engineering, System Governance.*

1. PENDAHULUAN

Berdasarkan PP No. 4 tahun 2014, Perguruan Tinggi adalah satuan pendidikan yang menyelenggarakan Pendidikan Tinggi. Penyelenggaraan Pendidikan Tinggi dilaksanakan oleh Perguruan Tinggi yang memiliki otonomi untuk mengelola sendiri lembaganya sebagai pusat penyelenggaraan Tridharma Perguruan Tinggi. Penyelenggaraan Pendidikan Tinggi yang lebih dikenal dengan istilah tata kelola pendidikan tinggi (*Higher Education Governance*) adalah sebuah sistem pengaturan, perencanaan, pengawasan, pemantauan, dan evaluasi untuk mencapai tujuan pendidikan tinggi, dimana telah diatur dalam keputusan penetapan Standar Nasional Pendidikan Tinggi. Salah satu standar utama yang ditetapkan adalah penerapan teknologi informasi dan komunikasi dalam tata kelola pendidikan tinggi.

Sistem Tata Kelola Perkuliahan (STKP) merupakan sistem penunjang yang telah dikembangkan. Sistem ini merupakan suatu model sistem terintegrasi yang meliputi pengaturan, perencanaan, pengawasan, pemantauan, dan evaluasi terhadap berjalannya proses akademik terutama kegiatan perkuliahan [1]. Hasil observasi pada sistem ini, menunjukan bahwa sistem ini telah mampu meng-integrasikan fungsi perencanaan, pelaksan aan,pengelolaan dan juga pengawasan proses akademik melalui layanan yang disediakan. Adapun pada tahap pengujian yang telah dilakukan sistem ini, saat diimplementasikan ke tablet, sistem ini mengalami beberapa kekurangan dan ketidaksesuaian dengan kondisi pada tahap pengimplementasiannya. Kekurangan meliputi *GUI* pada Selain itu, diketahui bahwa SIAMIK selalu berkembang dan Sistem Tata Kelola Proses Perkuliahan yang memiliki ketergantungan pada SIAMIK

mengakibatkan sistem harus diperbaharui agar bisa dintegrasikan dengan SIAMIK, dan serta pada pengembangan yang dilakukan oleh beberapa tim yang berbeda mengakibatkan struktur pemrograman yang tidak rapi atau teratur

Berdasarkan permasalahan yang telah diuraikan, pada penelitian ini dilakukan pemuktahiran modul java dan struktur data pada Sistem Tata Kelola Perkuliahan pada perangkat *tablet PC*. Elmblad [2] mengatakan pemuktahiran atau pembaruan perangkat lunak, yang kadang-kadang disebut perangkat lunak *patch*, adalah pengunduhan gratis untuk aplikasi, sistem operasi, atau *software suite* yang menyediakan perbaikan untuk fitur yang tidak bekerja sebagaimana dimaksud atau menambahkan peningkatan kecil perangkat lunak dan kompatibilitas Pembaruan kecil dan bertahap ini meningkatkan pengoperasian perangkat lunak.

Pemuktahiran modul java dan strukur data pada sistem ini merupakan penambahan model, adapun yang dimaksud ialah menerapkan konsep arsitektur *Model-View-Controller*. Pada pemuktahiran ini, dilakukan melalui rekayasa ulang sistem.

Tujuan dari rekayasa ulang sistem ialah untuk memahami sistem seperti spesifikasi, desain (arsitektur, komponen, dan interface). Kemudian peneliti menerapkan konsep arsitektur *Model-View-Controller* (*MVC*) yang meng-enkapsulasi data bersama dengan pemrosesan (*model*), mengisolasi dari proses manipulasi (*controller*) dan tampilan (*view*) untuk direpresentasikan pada sebuah user *interface* [3]. Peneliti menggunakan arsitektur *MVC* untuk me-muktakhirkan modul java dana struktur data pada sistem ini. Dengan menggunakan arsitektur *MVC* sistem memiliki struktur pemograman yang lebih baik, karena konsep *MVC* memisahkan memisahkan antara desain, data, dan proses. Sehingga dalam mengatasi bug, error, dan maintenance menjadi lebih mudah untuk dilakukan.

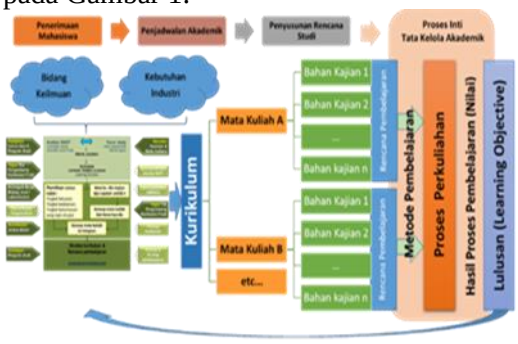
Istilah struktur data yang digunakan dalam penelitian ini adalah hubungan komposisi antara satu objek dari suatu class data dengan objek-objek lain yang menjadi penyusunnya. Selain itu, dilakukan pemuktahiran pada modul akuisisi data audio dan video serta pengembangan layanan interaksi modul absensi digital perkuliahan pada Sistem Tata Kelola Perkuliahan pada Universitas Atma Jaya Makassar. Dalam penelitian yang dilakukan, peneliti menggunakan struktur data Sistem Tata Kelola Perkuliahan sebelumnya yang disesuaikan dengan struktur data SIAMIK terbaru yaitu SIAMIK versi 5.0.

2. LANDASAN TEORI

2.1 Sistem Tata Kelola Universitas

Sistem tata kelola pendidikan tinggi merupakan suatu model sistem pengaturan dan perencanaan penerapan kurikulum pada pelaksanaan proses akademik meliputi pencapaian learning objective (LO) pada setiap mata kuliah melalui pemantauan pelaksanaan rencana pembelajaran serta penilaian pada proses belajar mengajar [1].

Pada artikel jurnal Pengembangan Prototipe Model Sistem Tata Kelola dan Pengawasan Proses Akademik pada Universitas Atma Jaya Makassar, Daniel Mohammad Rosyid seorang Guru Besar ITS mengatakan bahwa tata kelola pendidikan sering mendorong praktik dan budaya layanan pendidikan menjadi kurikulum yang terlaksana, yang bisa bertentangan dengan tujuan kurikulum yang direncanakan yang tampak bagus di atas kertas namun dipandang "berat konten miskin proses" [1]. Oleh karena itu, sebuah tata kelola pendidikan yang efektif haruslah berorientasi terhadap perbaikan dan peningkatan pada proses bukan hanya pada perbaikan konten maupun hasil dari proses yang selama ini dilaksanakan akademik seperti yang terlihat pada Gambar 1.



Gambar 1 Tata Kelola Proses Akademik (Syarif et al., 2015)

Pemenfaatan tata kelola teknologi informasi adalah sebuah hal yang serius (*critical*) dalam operasional suatu organisasi. Penerapan TI didalam organisasi dapat

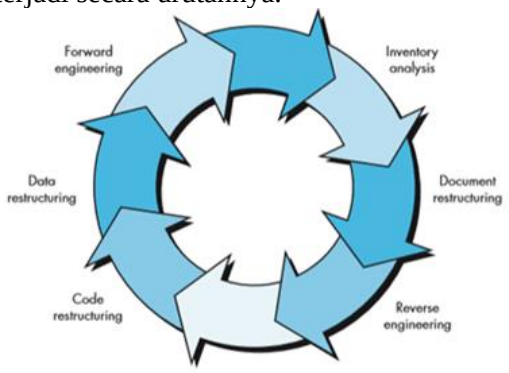
dilakukan dengan baik apabila ditunjang dengan suatu tata kelola TI mulai dari perencanaan sampai implementasinya.

2.2 Rekaysa Ulang Sistem

Rekayasa perangkat lunak ialah reorganisasi dan memodifikasi sistem perangkat lunak yang ada untuk membuatnya lebih mudah dipelihara. Rekayasa ulang sistem sering disebut renovasi yang dimana proses pengujian dan perubahan sistem dilakukan untuk disusun kembali menjadi sistem dengan bentuk baru dan implementasi baru. Pressman dan Maxim [4] menganalogikan rekayasa ulang sistem ini seperti membangun sebuah rumah. Sistem informasi yaitu suatu sistem yang menyediakan informasi untuk manajemen dalan mengambil keputusan dan juga untuk menjalankan operasional perusahaan dimana sistem tersebut merupakan kombinasi dari orang-orang, teknologi informasi dan prosedur-prosedur yang terorganisasi.

a. Sebelum mulai membangun kembali, kita harus memeriksa terlebih dahulu. Hal ini dilakukan untuk menentukan mana yang perlu dibagun kembali. Dengan tujuan unutk membuat sebuat daftar kriteria sehingga pemerikasaan rumah menjadi lebih sistematis.
b. Sebelum meruntuhkan dan mebangun kembali seluruh rumah, pastikan bahwa struktur rumah tidak kuat. Jika rumah dari sudut bangunya kuat, maka kita mungkin dapat "merombak" tanpa membangun ulang rumah tersebut (dengan biaya yang jauh lebih rendah dan dalam waktu yang lebih cepat).
c. Sebelum mulai membangun kembali rumah tersebut, pastikan memahami bagaimana bangunan rumah asli ini dibangun. Memahami kabel, pipa, dan struktur di dalam rumah. Ketika melakukan hal ini, maka pemahaman mengenai rumah ini akan memjadi lebih baik ketika kita akan memulai proses kontruksi.
d. Jika ingin membangun kembali, ingatlah untuk hanya menggunakan bahan-bahan yang tahan lama dan paling moderen. Ini mungkin membutuhkan biaya yang cukup mahal, tetapi ini akan membantu untuk terhindar dari pemeliharaan atau maintenance yang lebih mahal dan memakan waktu yang lebih lama.
e. Jika memutuskan untuk membangun kembali, disiplinlah mengenai poin yang telah dipaparkan sebelumnya. Gunakanlah hal-hal yang akan menghasilkan kualitas yang tinggi untuk hari ini dan dimasa depan.

Meskipun prinsip-prinsip ini berfokus pada pembangunan kembali rumah, prinsip ini juga berlaku sama baiknya dengan rekayasa ulang sistem dan aplikasi berbasis komputer. Untuk menerapkan prinsip ini, kita dapat mengunakan model proses rekayasa perangkat lunak yang mendefenisikan 6 kegiatan yang ditunjukkan pada Gambar 3. Secara umum, kegiatan ini berlangsung dalam urutan linear, tapi dapat pula tidak terjadi secara urutannya.



Gambar 2. Model Proses Rekayasa Ulang Sistem (Pressman, Maxim et al., 2015)

Model pola rekayasa ulang sistem seperti yang digambarkan pada gambar 2. Me-rupakan model siklus. Ini berarti bahwa setiap aktivitas disajikan sebagai bagian dari model pola yang mungkin dapat dikunjungi kembali. Untuk setiap siklus tertentu, proses dapat berhenti setelah kegiatan salah satu komponennya.

1. *Inventory Analysis*

Sistem dari suatu organisasi harus memiliki persediaan/invetarisasi untuk semua aplikasi maupun sistem. Bentuk dari persedian ini dapat berupa model spreadsheet yang berisi semua informasi yang memberikan penjelesan rinci (misalnya ukuran, usia, kekritisan bisnis) dari setiap aplikasi yang aktif dengan memilah informasi sesuai dengan kekritisan bisnis, umur, ukuran, pemeliharaan saat ini dan kriteria, calon sistem yang akan direkayasa ulang akan muncul. Sumber-sumber

informasi dapat dialokasikan untuk aplikasi yang akan direkayasa ulang.

Catatan penting bahwa setiap persedian/ inventory dapat dikunjungi kembali pada siklus yang tetap. Status sistem dapat berubah fungsinya seiring dengan berjalannya waktu dan hasilnya, prioritas rekayasa ulang akan bergeser.

2. *Document Restructuring*

Dokumentasi yang kurang kuat merupakan *trademark* dari banyak warisan atau versi dari sistem. Ada beberapa opsi yang dapat dilakukan untuk mengatasi hal ini.

a. Membuat dokumentasi untuk sebuah sistem akan terlalu menghabiskan banyak waktu. Tidak mungkin untuk membuat kembali dokumentasi untuk ratusan program dari sistem tersebut.
b. Dokumentasi harus diperbaharui tetapi setiap organisasi memiliki keterbatasan sumber informasi. Kita tidak perlu mengulang dokumentasi sepenuhnya. Sebaliknya, bagian-bagian dari sistem yang saat ini tengah mengalami perubahan akan didokumentasikan seluruhnya. Seiring waktu, kumpulan dari dokumentasi yang berguna dan relevan tentunya akan berkembang.
c. Sistem merupakan kritikal bisnis yang telah dipaparkan di atas merupakan pilihan yang dapat diambil. Sistem pada organisasi kita harus didokumentasikan kembali sepenuhnya. Bahkan didalam kasus ini, pendekatan yang baik ini adalah mengupas dokumentasi ke dalam sebuah sifat dasar minimum.

Setiap pilihan yang telah dipaparkan di atas merupakan pilahan yang dapat diambil. Sistem pada organisasi kita harus memilih salah satu yang paling tepat untuk setiap kasus.

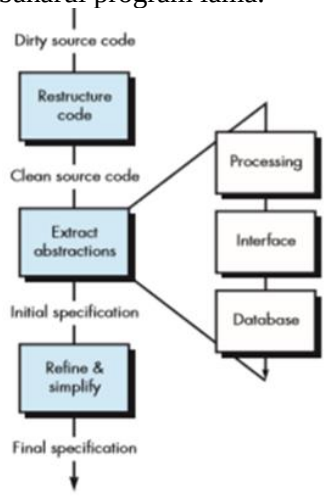
3. *Reverse Engineering*

Reverse Engineering adalah proses menganalisis program dalam upaya untuk menciptakan representasi program pada tingkat abstraksi yang lebih tinggi dari pada *source code*. *Reverse engineering* dapat mengekstrak informasi desain dari *source code*, tetapi tingkat *abstraksi*/pemisahan, kelengkapan dokumentasi, memilki ketergantungan dari alat dan seorang analis manusia bekerja bersama, dan arah dari proses sangat bervariasi.

Pada level abstraksi/pemisahan dari proses reverse engineering dan tools yang digunakan untuk memberikan pengaruh pengalaman dari informasi desain yang diekstrak dari *source code*. Idealnya, tingkat dari abstraksi harus memiliki level yang sangat tinggi. Proses *reverse engineering* harus mampu menurunkan gambaran dari cara kerja desain (pada level terendah dari abstraksi), informasi mengenai program dan struktur data (pada level yang lebih tinggi dari abstraksi), model objek data dan/ atau kontrol alur model (pada level yang relatif lebih tinggi dari abstraksi), model relasi entitas (pada level tinggi dari abstraksi) dalam peningkatan level abstraksi, kita akan diberikan informasi mengenai program yang lebih mudah dipahami.

Kelengkapan dari proses *reverse engineering* menunjuk kepada detail level yang disajikan pada sebuah level abstraksi.

Jika keterarahan proses *reverse engineering* adalah satu jalur, semua informasi yang diekstrak dari kode sumber/*source code* yang disajikan untuk rekayasa sistem dapat digunakan selama ada aktivitas maintenance, jika keterarahan proses ini dua jalur, maka informasi akan diterima oleh *tools* rekayasa ulang yang ada pada proses rekontruksi kembali atau memperbaharui program lama.



Gambar 3. Proses *Reverse Engineering*
(Pressman, Maxim et al., 2015)

4. *Code Restructuring*

Code restructuring dilakukan untuk menghasilkan desain yang menghasilkan fungsi yang sama tetapi dengan kualitas yang lebih tinggi dari program aslinya.tujuan utama dari tahap ini ialah agar mendapatkan desain *prosedural* yang sesuai dengan filosofi pemrograman terstruktur atau objek.

5. *Data Restructuring*

Tahap ini dilakukan jika pada proses *reverse engineering* yang disebut *analysis of source code* sudah dilakukan. Semua pernyataan bahasa pemograman yang mengandung defenisi data, deskripsi data, *I/O* dan deksripsi antar muka dievaluasi. Tujuannya untuk mengestrak data dan objek untu mendapatkan informasi tentang aliran data dan juga untuk memahami struktur data yang ada. Kegiatan ini biasa di disebut analisa data.

Setelah analisa data selesai, proses mendasain kembali data akan dilakukan. Sebuah standarisasi *record* data akan menjelaskan defenisi data untuk mendapatkan konsistensi di antara nama item data atau format record fisik didalam struktur data atau format *file* yang ada didalam bentuk pasling sederhana. Selain bentuk desain ulang, di sebut juga dengan data name rationalization, memastikan bahwa semua ketentuan penamaan data sesuai dengan standar lokal dan nama-nama lain dari data tersebut akan di singkirkan sebagain aliran data melalui sistem.

Ketika strukturisasi ulang berpindah melewati proses standarisasi dan rasionalisasi, modifikasi data untuk struktur data yang akan dibuat agar desain data lebih efektif. Mungkin ini berarti mengubah dari satu format *file* ke format *file* yang lain, ata di dalam beberapa kasus, mengubah dari satu tipe database ke tipe database lain.

6. *Forward Engineering*

Dalam dunia yang ideal, aplikasi akan dibangun kembali menggunakan "mesin rekayasa ulang otomatis." Program lama akan dimasukkan ke dalam mesin, dianalisis, direstrukturisasi, dan kemudian diregenerasi dalam bentuk yang menunjukkan aspek terbaik dari kualitas sistem. Dalam jangka pendek, tidak mungkin mesin semacam itu akan muncul, tetapi vendor telah memperkenalkan tools/alat yang menyediakan sub rangkaian yang di batasi oleh kemampuan ini yang mengatasi domain aplikasi tertentu (misalnya, aplikasi yang diimplementasikan menggunakan sistem basis data spesifik). Yang lebih penting lagi, alat reengineering ini menjadi semakin lebih canggih.

Forward engineering tidak hanya memulihkan informasi desain dari sistem yang ada tetapi menggunakan informasi ini untuk mengubah atau menyusun kembali sistem yang ada dalam upaya untuk meningkatkan kualitasnya secara keseluruhan. Dalam banyak kasus, sistem yang direkayasa ulang untuk membuat ulang fungsi sistem yang ada dan juga menambah fungsi baru dan/ atau meningkatkan kinerja secara keseluruhan [4].

2.3 Software Update

Elmblad (2019) pada artikel "The Difference Between Software Updates and Upgrades"mengatakan Software update atau pemuktahiran perangkat lunak adalah pembaruan perangkat lunak, yang kadang-kadang disebut perangkat lunak patch, adalah pengunduhan gratis untuk aplikasi, sistem operasi, atau software suite yang menyediakan perbaikan untuk fitur yang tidak bekerja sebagaimana dimaksud atau menambahkan peningkatan kecil perangkat lunak dan kompatibilitas. Pentingnya pembaruan pe-rangkat lunak. Pembaruan perangkat lunak memainkan peran penting yang sering terkait dengan penyelesaian atau pencegahan masalah:

a. Lindungi dari risiko keamanan yang baru ditemukan.
b. Perkenalkan fitur baru dalam perangkat lunak Anda.
c. Meningkatkan laju penipisan baterai atau kecepatan kinerja.
d. Perpanjang masa pakai peralatan Anda dengan memungkinkan produktivitas maksimumnya.
e. Perbaiki bug dalam perangkat lunak dan tingkatkan fungsionalitas.

2.4 *Model-View-Controller*

Dayat dan Angriani [3] mengatakan *Model-View-Controller* adalah sebuah konsep yang diperkenalkan oleh penemu Smalltalk (Trygve Reenskaug) untuk meng-enkapsulasi data bersama dengan pemrosesan (*model*), mengisolasi dari proses manipulasi (*controller*) dan tampilan (*view*) untuk direpresentasikan pada sebuah user interface. *MVC* mengikuti pendekatan yang paling umum dari Layering. Layering hanyalah sebuah logika yang membagi kode kita ke dalam fungsi di kelas yang berbeda. Pendekatan ini mudah dikenal dan yang paling banyak diterima. Keuntungan utama dalam pendekatan ini adalah penggunaan

ulang (*reusability*) kode. Definisi teknis dari arsitektur MVC dibagi menjadi tiga lapisan :

a. *Model*, digunakan untuk mengelola informasi dan memberitahu pengamat ketika ada perubahan informasi. Hanya model yang mengandung data dan fungsi yang berhubungan dengan pemrosesan data. Sebuah model meringkas lebih dari sekedar data dan fungsi yang beroperasi di dalamnya. Pendekatan model yang digunakan untuk komputer model atau abstraksi dari beberapa proses dunia nyata. Hal ini tidak hanya menangkap keadaan proses atau sistem, tetapi bagaimana sistem bekerja. Sebagai contoh, programmer dapat menentukan model yang menjembatani komputasi back-end dengan front-end GUI (graphical user interface).
b. *View*, bertanggung jawab untuk pemetaan grafis ke sebuah perangkat. View biasanya memiliki hubungan one to one dengan sebuah permukaan layar dan tahu bagaimana untuk membuatnya. View melekat pada model dan me-render isinya ke permukaan layar. Selain itu, ketika model berubah, view secara otomatis menggambar ulang bagian layar yang terkena perubahan untuk menunjukkan perubahan tersebut. Terdapat kemungkinan beberapa view pada model yang sama dan masing-masing view tersebut dapat merender isi model untuk permukaan tampilan yang berbeda.
c. *Controller*, menerima input dari pengguna dan menginstruksikan model dan view untuk melakukan aksi berdasarkan masukan tersebut. Sehingga, controller bertanggung jawab untuk pemetaan aksi pengguna akhir terhadap respon aplikasi. Sebagai contoh, ketika pengguna mengklik tombol atau memilih item menu, controller bertanggung jawab untuk menentukan bagaimana aplikasi seharusnya merespon.

3. METODOLIGI PENELITIAN

3.1 Rancangan Dan Cara Kerja

Pada penelitian ini penulis menggunakan pendekatan perancangan waterfall dengan metode UML. Pendekatan Waterfall me-rupakan suatu proses pengembangan software yang bersifat sekuensial atau berurutan, dimana setiap tahapan yang dikerjakan secara berurut menurun. Fase-fase dalam waterfall Model [4]:

1. Communication (*Project Initiation and Requirements Gathering*)

Pada fase pertama penulis menentukan kebutuhan yang digunakan dalam mengembangkan layanan sistem tata kelola perkuliahan dengan komponen modul peralatan STKP.

2. *Planning* (*Estimating, Scheduling, Tracking*)

Di tahap ini peneliti menentukan data atau fitur yang digunakan oleh sistem berdasarkan hasil pada fase pertama.

3. *Modeling* (Analysis and Design)

Di tahap ini peneliti merancang desain pengembangan Sistem Tata kelola Perkuliahan perangkat tablet PC berdasarkan hasil dari fase Planning yang

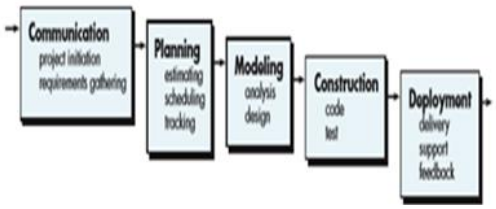
telah dilakukan seperti merancang use case diagram, class diagram, dan sequence diagram pada sistem tata kelola perkuliahan.

4. *Construction* (*Code* and *Test*)

Di tahap ini hasil dari tahap modeling yang telah dilakukan, dilanjutkan dengan peng-kodean sistem tata kelola perkuliahan FTI-UAJM dengan menggunakan bahasa pemograman java, kemudian dilakukan pengujian blackbox yang berfokus pada hasil eksekusi melalui data uji dan memeriksa fungsional dari sistem yang dikembangkan.

5. *Deployment*(*Delivery*, *Support*, *Feedback*)

Di tahap ini merupakan tahap implementasi, perbaikan, evaluasi, dan pengembangan berdasarkan umpan balik yang diberikan agar sistem dapat tetap berjalan dan berkembang sesuai dengan fungsinya di FTI-UAJM.



Gambar 4. Model Perancangan *Waterfall* (Pressman, Maxim et al., 2015)

3.2 Metode Pengumpulan Data

1. Studi Literatur

Studi literatur dilakukan dengan mempelajari maupun menelaah isi literatur dari referensi penelitian Rekayasa Ulang dan Integrasi Sistem Digital Faculty dengan Sistem Informasi Akademik Universitas Atma Jaya Makassar [5].

2. Wawancara

Wawancara yang dilakukan dalam penelitian ini kepada dosen Fakultas Teknologi Informasi yang berjumlah 4 orang dan Dekan Fakultas Teknologi Informasi. Lokasi wawancara bertempat di gedung FTI pada jam 12.00 – 13.00 AM.

3.3 Analisa Data

Teknik analisis data yang digunakan dalam kegiatan penelitian ini adalah analisa data kualitatif. Analisis data kualitatif diolah dari hasil pengumpulan data wawancara baik berupa lisan maupun tulisan tentang kinerja proses akademik program studi.

4. HASIL DAN PEMBAHASAN

4.1 Penelitian Sebelum

Sistem Tata Kelola Perkuliahan(STKP) terbagi menjadi dua yaitu sistem yang diperuntukan untuk perangkat tablet PC dan aplikasi web. Sistem Tata Kelola Perkuliahan yang berbasis aplikasi web digunakan untuk mengelola data mata kuliah seperti daftar hadir, nilai mahasiswa, data rencana pembelajaran, berita acara perkuliahan, dan lain sebagainya. Sedangkan pada Sistem Tata Kelola yang diperuntukan untuk perangkat tablet PC digunakan untuk membantu dosen dalam kegiatan mengajar di kelas dalam hal pengabsensian, bahan materi ajar, jadwal mengajar dan lain sebagainya pada setiap pertemuan dilakukan.

Penelitian ini, merupakan penelitian yang telah dikembangkan sebelumnya dengan judul Penerapan Digital Faculty pada Fakultas Teknologi Informasi Universitas Atma Jaya Makassar [6]. Hasil dari penelian yang dialakukan ialah suatu Sistem Digital Faculty yang dapat mengakomodasi perencanaan, pelaksanaan, penilaian serta pengawasan proses perkuliahan di FTI-UAJM, dan dilanjutkan oleh The [5] dengan judul penelitian Rekayasa Ulang dan Integrasi Sistem Digital Faculty dengan Sistem Informasi Akademik Universitas Atma Jaya Makassar, penelitian yang dilakukan bertujuan untuk meningkatkan diseminasi pada sistem Digital Faculty melalui proses rekayasa ulang dan integrasi sistem ini dengan Sistem Informasi Akademik (SIAMIK).

Penelitian yang dilakukan merupakan penelitian lanjutan yang dilakukan The [5] dengan judul Rekayasa Ulang dan Integrasi Sistem Digital Faculty dengan Sistem Informasi Akademik Universitas Atma Jaya Makassar. Sistem ini dikembang-kan menggunakan pemrograman Java dan perancangannya berbasis *objek* (OOP). Sistem ini memiliki *package* atau *folder* yang berjumlah sebelas *package* yang dimana tiap *package* tersebut memiliki fungsinya tersendiri. Sedangkan struktur data pada sistem ini tergabung dengan *method bisnis logic* dan pemprosessan tampilan.

Rekayasa ulang dilakukan untuk menyusun ulang struktur kelas dari Sistem Tata Kelola Perkuliahan pada perangkat tablet PC, sehingga memiliki struktur kelas yang lebih rapi menggunakan arsitektur *model-view-controller* (MVC).

Tahap pertama yang dilakukan ialah melakukan rekayasa ulang yang bertujuan untuk mendapatkan informasi mengenai struktur sistem sebelumnya. Informasi yang dimaksud ialah pemahaman tentang alur data dari sistem tersebut, inisialisasi dan deklarasi pada sebuah variabel maupun dari objek suatu kelas, logika proses dari suatu kelas dan struktur organisasi folder sistem. Informasi tersebut digunakan sebagai daftar kriteria untuk menetukan kelas yang perlu direkayasa ulang. Setelah melakukan rekayasa ulang pada struktur data dan kelas pada sistem, kelas tersebut dibangun ulang dengan menerapkan konsep *model-view-controller*.

4.2 Rekayasa Ulang Sistem Tata kelola Perkuliahan

4.2.1 *Inventory Analysis*

Tahap ini, peneliti mengumpulkan informasi sistem secara lebih rinci, baik ukuran, usia, kriteria, dan juga kekritisan bisnis sistem.

1. Ukuran : 46.7 MB.
2. Usia : 3 tahun 7 bulan.
3. Kriteria

Sistem Tata Kelola Perkuliahan terbagi menjadi dua sistem yaitu:

a. Sistem Tata Kelola Perkuliahan perangkat tablet PC : 10.7 MB.
   Sistem ini digunakan untuk membantu proses perkuliahan yang berlangsung di kelas.
b. Sistem Tata Kelola Perkuliahan aplikasi web : 35.5 MB.
   Sistem ini digunakan unutk membantu civitas akademika untuk membantu mengelola data-data perkuliahan.

4. Kekritisan bisnis

Sistem ini pada dasarnya harus diimplementasikan mengingat perannya dalam mengakomodasi, pelaksanaan, penilaian serta pengawasan proses perkuliahan di FTI-UAJM. Namun berdasarkan pengujian yang dilakukan diketahui sistem ini mengalami kendala yaitu struktur pemograman yang tidak rapi, struktur data SIAMIK yang selalu berubah, dan fitur absensi ketika di-implemtasikan pada *tablet*.

4.2.2 *Document Restructuring*

Peneliti menggunkanan opsi kedua yaitu sistem yang saat ini tengah mengalami perubahan didokumentasikan seluruhnya. Terdapat dua dokumentasi dari penelitian yang dilakukan yaitu Penerapan Digital Faculty pada Fakultas Teknologi Informasi Universitas Atma Jaya Makassar [5]. Dokumentasi kedua berasal dari hasil penelitian yang dilakukan oleh Permatasari [6] yaitu Rekayasa Ulang Dan Integrasi Sistem Digital Faculty Dengan Sistem Informasi Akademik Universitas Atma Jaya Makassar. Pada tahapan ini peneliti tidak men-dokumentasikan bagian-bagian sistem yang mengalami perubahan, namun kita mengumpulkan data-data maupun dokumen mengenai sistem *Digital Faculty* agar pada tahapan perancangan peneliti dapat menentukan bagian dari sistem yang akan di rekayasa ulang dan menerapkan konep arsitektur *model-view-controller*.

Hasil dari tahapan *document restructuring* ini dapat diketahui bahwa sistem ini berfungsi untuk membantu proses perkuliahan yang meliputi tahap perencanaan, pelaksanaan, penilaiaan, dan juga pengawasan perkuliahan. Sistem ini dalam perancangannya menggunakan pendekatan berbasis *Object Oriented Programing* (OOP) serta untuk mendukung fungsi pegawasan, sistem ini menggunakan perangkat IP Camera yang berada pada setiap kelas untuk membantu proses pengawasan perkuliahan di kelas.

4.2.3 *Reverse Engineering*

Tahap reverse engineering, peneliti mengevaluasi sistem dan serta menggunakan source code untuk membangun spesifikasi yang digunakan untuk menerapkan konsep arsitektur model-view-controller pada modul yang terdapat pada sistem, dan juga struktur data atau database program yang digunakan. Pada penelitian yang dilakukan, sistem ini menggunakan bahasa pemrograman Java sehingga memiliki beberapa kelas pada setiap layanan yang disediakan. Adapun proses evaluasi ini terbagi menjadi menjadi:

1) Modul layanan java

Strukturisasi pada modul layanan java dilakukan dengan membuat suatu kelas yang dapat menyediakan lanyan yang disediakan oleh kelas yang distrukturisasi. Pada Gambar 5, merupakan kelas yang berkerja untuk menangani *event*/peristiwa saat user memilih salah satu lanyanan yang di-sediakan sistem.

```
public class MainView {
    protected static JFrame frame;
    protected static Menu menu;
    protected static JPanel view = new JPanel();
    private JPanel view3;
    private boolean Am_I_In_FullScreen = false;
    private int PrevX,PrevY,PrevWidth,PrevHeight;
    private int p, pert, jum;
    private String bag,kd,nok,kelas,kls[],kodemk[],nokmk[],nmdos;
    private GetJadwal gj;
    private GetDetailSemTahun gdst = new GetDetailSemTahun();
public MainView(String nama){
        frame = new JFrame("Digital Faculty FTI UAJM");
        nmdos = nama;
        FullScreen(frame);
        initComponents();}
        protected void initComponents(){
        menu = new Menu(nmdos);
public void actionPerformed(ActionEvent e) {
            setTampilan(new TempAbsen());} });
        menu.btnrp.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e) {
                setTampilan(new TempRP());}
      });
  menu.btnrp.addActionListener(new ActionListener(){
           @Override
           public void actionPerformed(ActionEvent e) {
               setTampilan(new TempRP());}
      });
  menu.btndh.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e) {
                setTampilan(new TempAbsen());}
        });
  menu.btnbap.addActionListener(new ActionListener(){
           @Override
           public void actionPerformed(ActionEvent e) {
               setTampilan(new TempBAP());}
      });
  menu.btnmateri.addActionListener(new ActionListener(){
           @Override
           public void actionPerformed(ActionEvent e) {
               setTampilan(new TempMateri());}
      });
```

Gambar 5. Kelas MainView

Pada Gambar 5, ketika user memilih salah satu menu sebagai contoh menu Rencana Pembelajaran maka pada fungsi *menu.btnrp.addActionListener* akan melakukan proses pembuat tampilan interface dari menu rencana pembelajaran.

Fungsi *addActionListener* adalah fungsi yang bertindak sebagai pendengar dari *ActionEvent* dari sebuah objek (tombol rencana pembelajaran). Berikut beberapa modul layanan yang distrukturisasi.

a. Modul Layanan Pengisian Daftar Hadir

Pada layanan ini terdapat dua kelas yang berkerja untuk menyediakan layanannya yaitu kelas *TempAbsen* dan kelas *Absensi*. Pada Gambar 5, kita melihat bahwa kelas *TempAbsen* yang pertama kali berkerja kemudian dilanjutkan ke kelas *Absensi*, tapi pada kelas *TempAbsen* proses yang dilakukan adalah inisialisasi dan pemanggilan ke kelas *GetJadwal* untuk mendapatkan data nomor kurikulum, kode matakuliah, kelas, dan ruang. Kemudian data tersebut dikirim ke kelas Absensi yang digunakan untuk melakukan *query* ke database dan membuat tampilannya.

b. Modul Layanan Rencana Pembelajaran

Layanan pada modul ini berfungsi untuk melihat rencana pembelajaran yang telah disusun. Modul layanan ini memiliki dua kelas yang berkerja untuk menyediakan layanannya yaitu kelas *TemRP* dan *ViewRp*. Dalam pengkodean dilakukan terdapat proses yang dinama kelas TempRP dan *ViewRP* melakukan inisialisasi dan pemangilan objek yang berulang pada method konstruktornya. *Method konstruktor* adalah method yang pertamakali saat sebuah objek pertama kali diciptakan.

c. Modul Layanan Isi Berita Acara Perkuliahan

Modul ini tediri dari dua kelas dua kelas yang berkerja untuk menyediakan layanan nya yaitu kelas *TempBAP* dan kelas *BAP*. Pada Gambar 5, kelas *TempBAP* yang pertama yang dikerjakan ketika memilih tombol menu Isi Berita Acara Perkuliahan kemudian di-lanjutkan menuju kelas *BAP* yang secara umum kelas *BAP* merupakan kelas utama. Hal ini dikarnakan proses yang dilakukan di pada kelas *BAP* adalah pembuatan tampilan dan melakukan query ke database. Kelas *TempBAP* hanya mengirimkan data yang digunakan untuk melengkapi *query* yang terjadi di kelas *BAP*.

d. Modul Lihat Jadwal Perkuliahan

Pada modul ini, berfungsi untuk menampilkan seluruh jadwal perkuliahan yang terdaftar selama satu minggu.

e. Modul Layanan Lihat Daftar Hadir

Pada layanan ini terdapat dua kelas yang berkerja untuk menyediakan layanannya yaitu kelas kelas TempVabsensi dan kelas

ViewAbsensi. Pada Gambar 6, dapat dilihat bahwa pada pada *method* konstruktornya melakukan inisialisasi dan pemangilan objek yang berulang.

```
public ViewAbsensi(){
gj = new GetJadwal();
kd = gj.getKode();
nok = gj.getNokur();
jumbag = getBagian();
initComponents();
aksesData();
}
public ViewAbsensi(String kode,
String nokur, String kelas){
gj = new GetJadwal();
kd = kode;
nok = nokur;
kls = kelas;
jumbag = getBagian();
initComponents();
aksesData();
}
```

```
public TempVAbsensi(){
gj = new GetJadwal();
kode = gj.getKode();
kd = kode.split(",");
nokur = gj.getNokur();
nok = nokur.split(",");
kelas = gj.getKelas();
kls = kelas.split(",");
initComponents();
}
```

Gambar 6 *Kelas TempVabsensi dan kelas ViewAbsensi*

f. Modul Layanan Lihat Berita Acara Perkuliahan

```
public TempVBAP(){
gj = new GetJadwal();
kode = gj.getKode();
kd = kode.split(",");
nokur = gj.getNokur();
nok = nokur.split(",");
kelas = gj.getKelas();
kls = kelas.split(",");
initComponents();}
```

```
public ViewBAP(){
gj = new GetJadwal();
kd = gj.getKode();
nok = gj.getNokur();
initComponents();
aksesData();
}
```

Gambar 7 *Kelas TempVBap dan kelas ViewBap*

Pada Gambar 7, terlihat bahwa pada *method konstruktor* dari kedua kelas tersebut melakukan inisialisasi dan pemangilan objek yang berulang pada objek *GetJadwal*.

Berdasarkan hasil evaluasi dari setiap modul layanan yang disediakan sistem diketahui bahwa kesimpulan bahwa dalam pengkodean yang dilakukan pada pengembagan sebelumnya bahwa kelas-kelas yang terdapat pada sistem terdapat beberapa kelemahan dalam pengembangan nya. Salah satu kelemahan yang paling sering dijumpai ialah inisialisasi dan pemanggilan objek yang berulang. Maka pada kelas-kelas yang terdapat pada setiap modul layana, akan distrukturisasi ulang berdasarkan hasil yang didapat pada tahapan ini.

Peneliti juga melakukan distrukturisasi pada kelas *GetJadwal*, *GetaMAC, GetDetailSemTahun* dan *GetDetailMK*. Berdasarkan hasil observasi sistem, ketiga kelas tersebut merupakan kelas yang berkerja ketika dosen yang sudah terdaftar dan masuk kedalam sistem. Awalnya, sistem akan mencocokan *username* dan *password* dengan data yang ada pada *database*. Jika cocok, maka dicocokkan kembali dengan jadwal yang tersedia. Kemudian, sistem akan

menerima *ip address* dari kelas yang digunakan dari kelas *GetaMAC*. Pada kelas *GetaMAC* setelah menemukan ruang, maka *ip address* dari ruang yang telah ditetapkan sistem untuk *ip camera* pada kelas tersebut, selanjutnya akan dilakukan pengujian koneksi terhadap *ip camera*. Jika dapat terhubung, maka selanjutnya sistem akan mengecek jadwal perkuliahan pada jam tersebut. Jika jadwal tersedia untuk dosen pada jam tersebut, maka status pertemuan dan input absensi dosen yang terkait dengan mata kuliah yang tersedia pada jam tersebut akan bertambah dan akan masuk ke dalam beranda utama. Proses pengecekan jadwal kuliah terjadi pada *sourcode* kelas *getjadwal* kemudian dilanjutkan ke kelas *GetDetailMK* yang berfungsi untuk mendapatkan data matakuliah berdasarkan kode matakuliah dan nomor kurikulum yang berasal dari kelas *getjadwal*.

2) Modul Perekaman Video STKP

Pada penelitan sebelumnya yang dilakukan oleh Loe (2014), ketika dosen akan masuk ke dalam halaman utama maka akan ada window/jendela perekaman pada sistem ini yang akan berhenti merekam jika jendela tersebut ditutup secara manual. Hal ini kurang efektif maka pengembangan dilakukan lagi pada modul ini yang menggunakan mekanisme dimana window/ jendela perekaman yang akan menutup jendela perekaman secara otomatis saat dosen telah logout dari sistem (The, 2017). Pengembangan yang dilakukan pada kasus ini, peneliti akan menerapkan suatu modul yang akan merekam proses perkuliahan tanpa adanya window/jendela baru yang muncul.modul yang digunakan pada video perekaman ini yaitu *opencv-343.jar*.

4.2.4 *Code Restructuring*

Tahap *code restructuring*, peneliti melakukan *code restructuring* yaitu mengubah kode berdasarkan kekurangan-kekurangan dan spesifikasi yang telah dijelaskan pada subbab sebelumnya. Adapun juga, pada proses code restructuring peneliti menerapkan konsep arsitektur *model-view-controller* pada setiap modul yang terdapat pada sistem. Hasil *code restructuring* dan penerapan konsep arsitektur *model-view-controller* yang telah dilakukan dijabarkan sebagi berikut.

1) *Model*

Pada bagian model terdapat kelas-kelas yang mengandung data dan fungsi yang berhubungan dengan pemrosesan data. Biasanya terdiri *property* (*atribut*) dan method getter-setter terhadap atribut-atributnya.

```
public class Model_Absensi {
    public Model_DOsen ModelDosen;
    private int status = 0;
    private int bagian;
    private int jumbag;
    private int pertf;
    private int row;
    private String sesif = "data_sesi";
    private absensiDOA d = new absensiDOADB();

    public Model_Absensi() {
        ModelDosen = new Model_Dosen();
    }

    public int getPertf() {
        return pertf;
    }

    public int getRow() {
        return row;
    }

    public void setRow(int row) {
        this.row = row;
    }

    public int getbagian() {
        int bag = 1;
        int p = Integer.parseInt(ModelDosen.getPertemuan());
        pertf = p % 16;
        if (pertf == 0) {
            pertf = 16;
        }

        if (p > 16) {
            bag = 2;
        }
        if (p > 32) {
            bag = 3;
        }
        return bag;
    }
```

Gambar 8 Kelas *Model_Absensi*

2) *View*

Terdiri dari kelas-kelas yang berfungsi untuk menampilkan informasi dari *Model*. Informasi tersebut diproses dengan cara kelas pada view mengambil model yang namanya sesuai dengan *controller*, kemudian mengeksekusi metode dalam model dan memberikan hasilnya ke sebuah variabel untuk ditampilkan.

```
public class View_Frame_Absensi extends JPanel {
  private Labels kdmklbl, nmmklbl, harilbl, jamlbl, ruanglbl,
  dosenlbl, semtalbl, kelaslbl;
  public MultiLineTextArea dosen, kdmk, nmmk, hari, jam, ruang, semta,
  Kelas;
  private JPanel panel1, panel2, panel3, panel4, panelutama,
  panelkeyboard, panelbtn;
  private MenuButton home, save;
  public ImageIcon iconSlc = new
  ImageIcon(getClass().getResource("/images/selected.png"));
  public ImageIcon iconUnSlc = new
  ImageIcon(getClass().getResource("/images/unselected.png"));
  private Controller_Absensi ctrlAbsensi;
  private Model_Absensi mdlAbsensi;
public View_Frame_Absensi() throws ClassNotFoundException,
SQLException {
  mdlAbsensi = new Model_Absensi();
  ctrlAbsensi = new Controller_Absensi(this);
  ctrlAbsensi.setModel(mdlAbsensi);
  initComponents();
  ctrlAbsensi.aksesData();
public void initComponents() throws SQLException {
  kdmklbl = new Labels("  Kode Mata Kuliah", 0);
  nmmklbl = new Labels("  Nama Mata Kuliah", 0);
  harilbl = new Labels(" Hari", 0);
  jamlbl = new Labels("  Pukul", 0);
  ruanglbl = new Labels("  Ruang", 0);
  kelaslbl = new Labels("  Kelas", 0);
  dosenlbl = new Labels("  Dosen", 0);
  semtalbl = new Labels("  Semester/Tahun", 0);
```

Gambar 9 Kelas *View_Frame_Absensi*

3) *Controller*

Terdiri dari kelas-kelas bekerja berdasarkan task apa yang telah diminta dan berdasarkan task tersebut maka *controller*

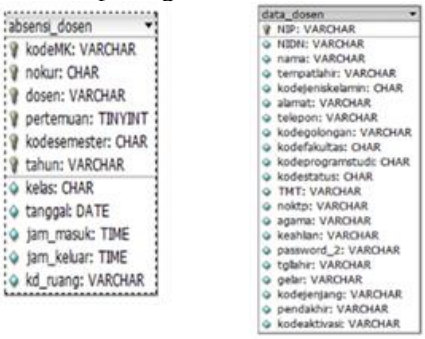
mengambil data dari *model* dan mengirimkan data dari *model* tersebut ke *view*. *Controller* berkerja berdasarkan inputan user dan biasa disebut dengan nama task. Jadi kelas-kelas pada controller yang bertugas unutk mengendalikan alur program berikut gambar kelas *controller*.

```
public class Controller_Absensi {
    private Model_Absensi mdlAbsensi;
    private View_Frame_Absensi frm;

    public KetHadir[] dftr;
    public KetAbsen keterangan[];
    private String stb[], kehadiran[];

    public JButton[] BtHdr;
    public JButton[] BtThdr;
    public ImageIcon iconSlc = new
ImageIcon(getClass().getResource("/images/selected.png"));
    public ImageIcon iconUnSlc = new
ImageIcon(getClass().getResource("/images/unselected.png"));

    private static int status = 0;

    public Controller_Absensi(View_Frame_Absensi frm) {
        this.frm = frm;
    }

    public void setModel(Model_Absensi mdlAbsensi) {
        this.mdlAbsensi = mdlAbsensi;
    }

    public ArrayList<String> getJumlahMhs() {
        ArrayList<String> dataMhs = new ArrayList<>();
        int pnjgdata = 0;
        dataMhs = mdlAbsensi.getdaftarMhs();
        pnjgdata = dataMhs.size() / 2;

        System.out.println("getdaftarMhs panjang data = "+ pnjgdata);
        for (int i = 0; i < dataMhs.size(); i++) {
            System.out.println("mhs ke-"+i+" = "+dataMhs.get(i));

        }
        mdlAbsensi.setRow(pnjgdata);

        return dataMhs;
    }
```

Gambar 10 Kelas *controller_Absensi*

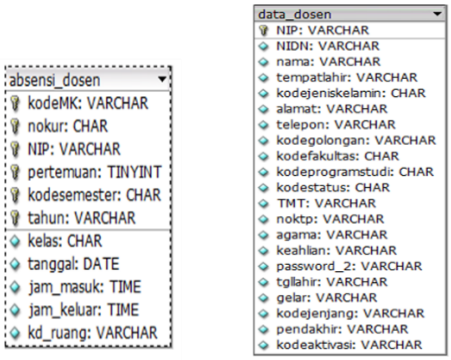
4.2.5 *Data Restructuring*

Pada tahap ini, peneliti melakukan data name rationalization, yaitu proses dimana kita memastikan bahwa semua ketentuan penamaan data sesuai dengan standar lokal dan nama-nama lain dari data tersebut akan disingkirkan sebagai aliran data suatu sistem. Menetapkan konsistensi di antara nama item data atau format record fisik di dalam struktur data ataupun format file yang ada di dalam kebentuk yang paling sederhana agar aliran data suatu sistem dapat terjaga. Salah satu contoh dari restukturisasi data pada sistem ini dapat dilihat pada gambar 11.



Gambar 11 Perbandingan Tabel Absensi Dosen dan Data Dosen

Gambar 11, menunjukkan perbandingan tabel absensi dosen dan jadwal tambahan yang digunakan pada sistem ini. Tabel yang berada di kiri merupakan tabel absensi dosen dan tabel yang berada di kanan merupakan tabel data dosen. Dapat terlihat perbedaan penamaan field pada kedua tabel ini. Tabel absensi dosen menggunakan nama dosen pada fieldnya, sedangkan tabel jadwal tambahan menggunakan nama nip. Oleh karena itu, kita harus menyamakan field tersebut, sehingga field kedua tabel ini akan disamakan menjadi NIP seperti yang ditunjukkan pada Gambar 12.



Gambar 12 Tabel Absensi Dosen dan Data DosenSetelah Strukturisasi Data

4.2.6 *Forward Engineering*

Pada tahap ini, informasi spesifikasi yang telah didapat digunakan untuk mengubah atau menyusun kembali sistem yang ada dalam upaya untuk meningkatkan kualitasnya secara keseluruhan (Pressman dan Maxim, 2015). Penyususnan kembali sistem dilakukan menggunakan model perancangan waterfall yang terdiri sebagai berikut.

A. *Communication*

Penelitian yang dilakukan telah memilki *prototype* dari sistem yang telah dikembangkan sebelumnya. Untuk menganalisa kebutuhan dari Sistem Sistem Tata Kelola Perkuliahan ini, peneliti melakukan metode pengumpulan data melalui studi literatur dari penelitian sebelumnya yang dilakukan oleh The (2017). Dan juga melakukan observasi di FTI-UAJM. Hasil dari metode pengumpulan data ini, diketahui bahwa terdapat beberapa kesalah/masalah.

Pada Sistem Tata Kelola Perkuliahan perangkat tablet PC yang telah melalui pengujian implementasi diketaui bahwa terdapat beberapa kendala yaitu kemudahan pemakai hal ini diakibatkan sistem akan diimplementasikan pada tablet, struktur

pemograman yang tidak rapi atau teratur akibat keterlibatan dari tim yang berbeda, struktur data SIAMIK yang selalu berubah, fitur absensi ketika diimplemtasikan pada tablet, serta telah diperbaharui database yang digunakan pada SIAMIK sehingga perolehan data menjadi tidak sinkron antara database Sistem Tata Kelola Perkuliahan perangkat tablet PC dengan database SIAMIK yang baru.

B. *Planning*

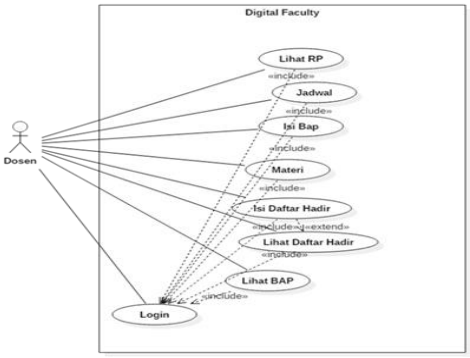
Berdasarkan hasil studi literatur dan rekayasa ulang sistem pada fase sebelumnya, diperoleh beberapa spesifikasi pada sistem sebelumnya seperti alur data. Berikut ini contoh alur data yang berkerja secara umum dimulai saat user atau dosen login ke sistem dengan memasukan username dan password, jika data username dan password terdaftar di database user dapat mengakses layanan yang disediakan sistem pada kelas login yang bertugas untuk menangani proses login. Pada kelas login tersebut terdapat variabel gj dan gdst yang merupakan objek dari kelas GetJadwal dan GetDetailSemTahun. Kedua objek tersebut memiliki fungsi atau method yang bertugas untuk mendapatkan data-data yang berkaitan dengan proses perkuliahan seperti jam mulai perkuliahan, nama matakuliah, kode matakuliah, dan nomor kurikulum. Berdasarkan pembahasan sebelumnya mengenai proses login, jika data *username* dan *password* benar dan terdaftar dalam sistem user dapat mengakses layanan yang disediakan sistem tapi jika user atau dosen tersebut tidak memiliki jadwal mengajar maka user tidak dapat masuk dan mengakses sistem.

Berdasarkan pembahasan yang telah diuraikan, pada tahap ini peneliti menentukan fitur dan data yang akan digunakan sebagai berikut:

1. Sistem yang dikembangkan menggunakan fitur yang ada sebelumnya tapi untuk beberapa layanan seperti modul layanan absensi, modul perekaman.perkuliahan, dan berita acara perkuliahan. Peneliti melakukan rekayasa ulang interaksi mudul layanan absensi dan modul perekaman perkuliahan.
2. Mengintegrasikan struktur data STKP dengan SIAMIK.

C. *Modeling*

Sistem ini diperuntukan untuk absensi dan membantu kegiatan mengajar di dalam kelas serta dikembangkan berbasis objek. Pada tahap ini peneliti menganalisa sistem dengan sebelumnya. Spesifikasi-spesifikasi yang telah didapatkan dari tahap *reverse engineering* digunakan untuk merancang desain sistem yang dilakukan sebagai berikut



Gambar 13 Use Case Diagram Sistem Tata Kelola Perkuliahan tablet PC

D. *Contruction*

Rekayasa ulang kelas berdasarkan hasil fase modeling dan penyesuaian struktur data Sistem Tata Kelola Perkuliahan ke SIAMIK versi terbaru telah dilakukan. Berikut hasil rekayasa ulang STKP yaitu.

Gambar 14 Perbandingan Struktur folder Sistem Tata Kelola Perkuliahan sebelum dan sesudah rekayasa

Pada Gambar 14, gambar sebelah kiri menunjukan perubahan struktur folder dari sistem, package di susun berdasarkan penerapan konsep model-view-controller. Package yang mengalami perubahan ialah Package Menu dan Package Get. Pada Package Menu, kelas yang terdapat didalamnya distrukturisasi berdasarkan hasil

evaluasi pada tahap reverse engineeringg. Adapun selain struktur folder, modul yang berkerja untuk menyediakan layanan juga ikut mengalami perubahan dalam memproses layanannya. Berikut ini modul-modul yang mengalami perubahan yaitu:

1. Menu Isi Daftar Hadir.
2. Menu Rencana Pembelajaran.
3. Menu Isi Berita Acara Perkuliahan.
4. Menu Materi.
5. Menu Lihat Berita Acara Perkuliahan.

Selain perubahan pada modul sistem ini, modul perekaman yang digunakan juga berubah. Pada penelitian sebelum, modul yang digunakan untuk merekam yaitu menggunkan vlc. Sedangkan pada penelitian ini men-ggunakan modul opencv. Peneliti me-nggunakn modul opencv dikarenakan modul tersebut memiliki banyak fitur yang bermanfaat, apabila sistem ini di-implementasikan di *tablet pc* untuk peng-implementasiannya.

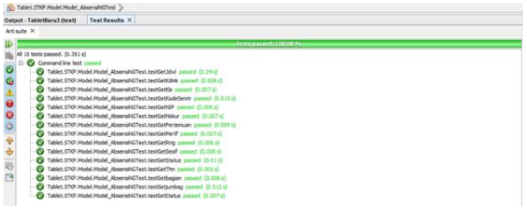
E. *Uji Kesahian*

Pengujian terhadap sistem dilakukan dengan menggunkan metode blackbox Testing dan Unit Testing. Blackbox testing berfungsi untuk menguji fungsi modul yang terdapat pada sistem, sedangkan Unit Testing fokus pada usaha verifikasi pada unit yang terkecil pada desain perangkat lunak (komponen atau modul perangkat lunak). Setiap unit perangkat lunak diuji agar dapat diperiksa apakah aliran masukan (input) dan keluaran (output) dari unit sudah sesuai dengan yang diinginkan.

1. Pengujian unit

Pengujian unit (*Unit Testing*), peneliti menggunakan *tools TestNg*. TestNg adalah kerangka kerja pengujian untuk bahasa pemrograman Java yang dibuat oleh Cédric Beust dan terinspirasi oleh JUnit dan Nunit. Pada Gambar 15 kelas Model_Absensi yang dibuat oleh TestNg untuk melakukan testing. anotasi @Test untuk memberi tahu kompiler bahwa method tersebut merupakan method yang digunakan untuk testing. Panggil method yang akan diuji pada method test tadi dan lakukanlah testing pada method tersebut. Lalu bandingkan output yang diharapkan dan output yang dihasilkan oleh method yang sedang diuji dengan fungsi *.assertEquals*(). Pengujian yang dilakukan pada folder Model Kelas Model_Absensi.

Gambar 15 Kelas Model_*AbsensiNGTest* Unit Testing



Gambar 16 hasil unit testing Kelas *Model*_AbsensiNGTest

2. Pengujian Blackbox

Pengujian dilakukan oleh 2 dosen Fakultas Teknologi Informasi UAJM. berfungsi untuk menguji fungsi modul yang terdapat pada sistem sistem yang telah dikembangkan telah memenuhi kebutuhan dari suatu organisasi, dalam hal ini FTI-UAJM dan telah sesuai dengan tujuan penelitian.

Uji kesahihan dan wawancara dilakukan terhadap dosen dan juga ketua prodi FTI-UAJM dan hasil yang didapatkan pada uji kesahihan ini yaitu sebagai berikut.

a. Pengisian Berita Acara Perkuliahan telah sesuai dengan prosedur yang telah diterapkan pada FTI-UAJM. .
b. Pada menu Lihat Berita Acara Perkuliahan lebih baik tambahkan keterangan/histori seperti tanggal pada menu berita acara perkuliahan.
c. Menu Materi lebih baik disusun menjadi berdasarkan grid dan tambahkan status keterangan pertemuan.
d. Perlu di perhatikan proses penginputan dan output yang dihasilkan dari sistem.
e. Untuk modul perekamnya kalau bisa, kamera akan berhenti merekam ketika akan memasuki sesi berikutnya.

5. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan oleh penulis, maka kesimpulan yang dapat diambil adalah

1. Penelitian ini telah menghasilkan sistem yang menerapkan konsep arsitektur

Model-View-Controller unutk Sistem Tata Kelola Perkuliahan pada perangkat tablet PC dimana arsitektur tersebut terbagi menjadi 3 lapisan yaitu model merupakan komponen yang hanya mengandung data dan fungsi yang berhubungan dengan pemrosesan data. View merupakan komponen yang bertanggung jawab untuk pemetaan grafis ke sebuah perangkat. Controller yang berfungsi untuk menerima input dari pengguna dan menginstruksikan model dan view untuk melakukan aksi berdasarkan masukan.

2. Rekayasa ulang pada strukur data dan modul layanan pada Sistem Tata Kelola Perkuliahan telah menghasilkan bahwa sistem dapat terintegrasi dengan SIAMIK serta layanan yang disediakan oleh sistem mampu berjalan dengan baik dan benar.

6. DAFTAR PUSTAKA

[1] Syarif, A.C, Gunawan, F.H dan Lisangan E.A. 2015. Pengembangan Prototipe Model Sistem Tata Kelola Dan Pengawasan Proses Akademik Pada Universitas Atma Jaya Makassar. Vol. 3, No.2

[2] Elmblad,Shelley. 2019.The Differe nce Between Software Updates and Upgrades.(Online)(https://www.The balance.com/what-is-a-software-update-vs-software-upgrade-1294256 diakses 15 Juli 2019)(1).

[3] Dayat, Abd. Rachman dan Angriani Liza. 2017. Pemanfaatan Model-View-Controller (MVC) Dalam Rancang Bangun Sistem Informasi Rakornas APTIKOM 2017. Seminar Nasional APTIKOM (SEMNASTIKOM)

[4] Pressman, Roger S., Maxim Bruce R., 2015. Software engineering: a practitioner's approach,eighth edition,Raghurinivasan.(Online)(https://downloadnema.com/wp-content/uploads/2017/02/Software%20Engineering%20A%20Practitioner%E2%80%99s%20Approach%20eighth%20edition-(www.downloanema.com).pdf di akses 22 Oktober 2015).(2)

[5] The, Jelita Permatasari Thehumury. 2017. *Rekayasa Ulang dan Integrasi Sistem Digital Faculty dengan Sistem Informasi Akademik Universitas Atma Jaya Makassar*. Skripsi program Sarjana. Makassar: Universitas Atma Jaya Makassar.

[6] Loe, I. 2014. *Penerapan Digital Faculty pada Fakultas Teknologi Informasi Universitas Atma Jaya Makassar*. Skripsi program Sarjana. Makassar: Universitas Atma Jaya Makassar.