RANCANG BANGUN APLIKASI SISTEM SIDANG SKRIPSI MENGGUNAKAN METODE OBJECT ORIENTED

Slamet Arif Billah
Selamet Hariadi
Jurusan Teknik Informatika, Fakultas Sains dan Teknologi Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang.

Abstrak - *Object Oriented Programming (OOP) adalah perangkat lunak yang dihasilkan dari pemodelan menggunakan UML yang berisi analisis dan perancangan perangkat lunak yang merupakan perpaduan dari beberapa metode yang telah ada sebelumnya. Penelitian ini berisi masalah pada sistem yang telah ada, yaitu kurang terintegrasinya data terkait sidang skripsi, pelaporan yang masih dikerjakan secara manual, dan databasing yang kurang sistematis. Solusi dari permasalahan yang ada adalah dengan implementasi OOP sebagai metodologi dalam analisis, perancangan dan pemrograman sistem sidang skripsi mulai dari pembimbingan skripsi, pelaksanaan sidang, penilaian hasil sidang, sampai pada pelaporan-pelaporan. Tujuan penelitian ini adalah untuk menghasilkan implementasi OOP pada aplikasi sistem sidang skripsi yang diharapkan menjadi solusi dari keterbatasan-keterbatasan sistem yang ada. Simpulan dari penelitian ini adalah bahwa aplikasi sidang skripsi yang menggunakan metode OOP memiliki banyak keunggulan, diantaranya adalah sifat reusability program, maintenance yang tidak rumit, dan mudah untuk dikembangkan.*

Kata Kunci: *Object Oriented Programming*,UML, sidang skripsi

A. PENDAHULUAN

Kepesatan teknologi yang telah sampai pada era kepadatan informasi menuntut setiap pihak pemilik informasi untuk mengelola informasi mereka secara integratif dan sistematis. Sehingga data menjadi aman, mudak dipakai dan dapat diolah secara efektif dan efisien. Untuk menghasilkan pengelolaan data pada sistem yang mudah dan handal, maka diperlukan suatu aplikasi yang menjadi penghubung sekaligus pengatur pengelolaan data dan informasi tersebut.

Aplikasi yang dihasilkan menggunakan metodologi yang bersifat *object oriented* memiliki banyak keuntungan dan kehandalan dibandingkan menggunakan pendekatan lainnya, sehingga pembangunan aplikasi sidang skripsi di sini menggunakan *object oriented* sebagai metodologi analisis, perancangan dan pemrogramannya.

B. TINJAUAN PUSTAKA

1. Pendekatan Berorientasi Objek

Berorientasi objek atau *object oriented* merupakan paradigma baru dalam rekayasa perangkat lunak yang memandang sistem sebagai kumpulan objek-objek diskrit yang saling berinteraksi. Yang dimaksud berorientasi objek adalah bahwa mengorganisasikan perangkat lunak sebagai kumpulan objek-objek yang diskrit yang bekerja sama antara informasi atau struktur data dan perlilaku (*behaviour*) yang mengaturnya [4].

Orientasi objek adalah pergeseran paradigma, menghujam sampai ke proses mental bukan sekadar berubah bahasa pemrograman. Orientasi objek adalah cara pandang bukan sekedar algoritma yang diterpkan pada bahasa berorientasi objek. Cara pandang ini harus dibiasakan dari level mikro sampai level tertinggi.

Cara berpikir berorientasi objek adalah segala sesuatu dipandang sebagai objek. Paradigam rekayasa perangkat lunak berorientasi objek berbasis pada konsep berorientasi objek.

Awalnya pengembangan aplikasi berorientasi objek berfokus pada gagasan objek/kelas dan pewarisan, sedangkan saat ini telah berfokus pada komponen. Gagasannya yang saat ini diusung adalah membangun aplikasi menggunakan komponen-komponen yang telah tersedia (dibuat sendiri, spesifik, atau komersial) yang handal dan dapat dipelihara. Coad-Yourdon mendefinisikan pendekatan berorientai objek dengan persamaan berikut:

Berorientasi objek = objek +
klasifikasi +
pewarisan +
komunikasi

Persamaan tersebut adalah persamaan untuk menggambarkan orientasi objek yang paling sederhana. Bila mengikuti Booch maka persamaan akan meliputi tujuh komponen, yaitu:

Berorientasi objek = abstraksi (objek) +
pengkapsulan +
modularitas +
hirarki +
[*typing* +
konkurensi +
persistensi]

Pendekatan berorientasi objek mempunyai keunggulan sebagai berikut:

- Pendekatan objek menuntun penggunaan ulang (*reuse*) komponene-komponen program sebelumnya. Penggunaan kembali menuntun pengembangan perangkat lunak yang lebih cepat dan berkualitas lebih tinggi
- Perangkat lunak yang dikembangkan dengan berorientasi objek mempermudah pemeliharaan karena strukturnya secara *inheren* sudah *decouple*.
- Sistem berorientasi objek lebih mudah diadaptasi dan diskala menjadi sistem lebih besar karena sistem-sistem lebih besar dibuat dengan merakit subsistem-subsitem yang dapat diguna ulang

2. Analisis Berorientasi Objek

Sebutan lengkap analisi adalah analisis kebutuhan perangkat lunak (*software requirement*). Analisis kebutuhan adalah mendaftar apa-apa yang harus dipenuhi oleh sistem perangkat lunak, bukan mengenai bagaimana sistem perangkat lunak melakukannya.

Analisis berorientasi objek mendefinisikan semua kelas yang relevan terhadap masalah beserta operasi-operasi dan atribut-atribut yang diasosiaksikan dengan kelas itu, keterhubungan di kelas-kelas dan perilaku yang dimilikinya. Sasaran analisis berorientasi objek adalah untuk mengembangkan model yang mendeskripsikan perangkat lunak yang memenuhi sekelompok kebutuhan yang didefinisikan pemesan.

Analisis berorientasi objek menggunakan sejumlah pemodelan untuk memenuhi sasaran. Model analisis akan mengekspresikan informasi, perilaku, dan fungsi di dalam konteks model objek.

Analisis menghasilkan pandangan logik (*logical view*) terhadap sistem untuk mendeskripsikan keberadaan dan maksud abstraksi dan mekanisme kunci yang membentuk ruang masalah atau yang mendefinisikan arsitektur sistem. Selama analisis berorientasi objek, kita harus menangani pertanyaan berikut:

a. Apa perilaku-perilaku sistem yang diinginkan?

b. Apa peran dan tanggungjawab masing-masing kelas objek agar menyediakan perilaku-perilaku yang ada di dalamnya?

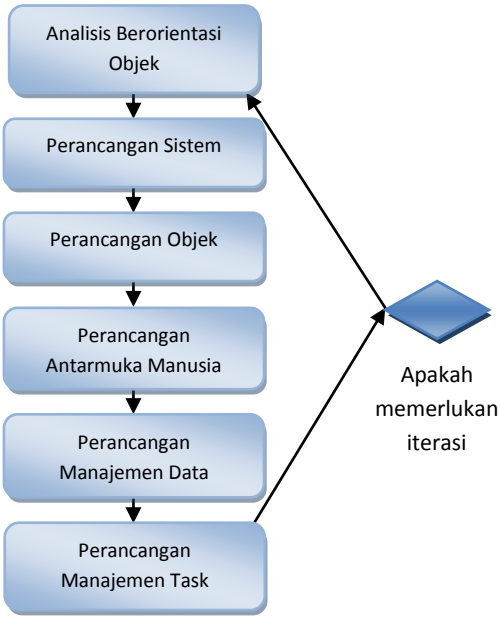
3. Perancangan Berorientasi Objek

Perancangan merupakan penghubung antara spesifikasi kebutuhan dan implementasi. Perancangan merupakan rekayasa representasi yang berarti terhadap sesuatu yang hendak dibangun. Hasil perancangan harus dapat ditelusuri

sampai ke spesifikasi kebutuhan dan dapat diukur kualitasnya berdasar kriteria-kriteria rancangan yang bagus. Perancangan menekankan pada solusi logik mengenai cara sistem sistem memenuhi kebutuhan.

Perancangan berorientasi objek mengidentifikasi dan mendefinisikan kelas-kelas dan objek-objek yang secara langsung merefleksikan domain masalah dan tanggungjawab sistem di dalamnya.

Berikut adalah tahap-tahap perancangan yang diadapasi dari metode *Coad-Yourdon* dapat digambarkan sebagai berikut:



Gambar 1. Tahap-tahap Perancangan Berorientasi Objek

4. Pemrograman Berorientasi Objek

Booch menyatakan: *Pemrograman berorientasi objek adalah metode implementasi dimana program diorganisasikan sebagai kumpulan objek yang bekerja sama, masing-masing objek merepresentasikan instan dari kelas, dan kelas-kelas itu anggota suatu hirarki kelas-kelas yang disatukan lewat keterhubungan pewarisan.*

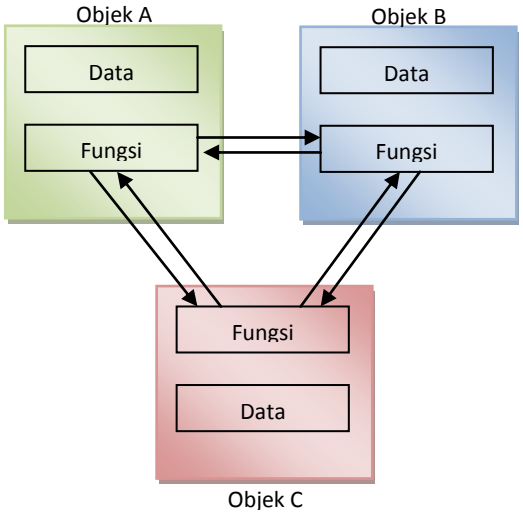
Tiga aspek penting dalam pemrograman berorientasi objek:

a. Menggunakan objek-objek bukan algoritma-algoritma sebagai blok-blok bangunan logik dasar (hirarki "*part of*").
b. Masing-masing objek adalah instan suatu kelas
c. Kelas-kelas saling berhubungan lewat keterhubungan pewarisan ("*is a*")

Bahasa pemrograman berorientasi objek umumnya sama dalam mendukung konsep-konsep dasar pendekatan berorientasi objek berikut [3]:

a. Objek, kombinasi data dan operasi
b. *Polymorphism* saat jalan
c. Pewarisan

Tujuan akhir dari mempelajari pemrograman berorientasi objek adalah kemampuan untuk membangun program menggunakan konsep berorientasi objek. Sifat utama dari konsep berorientasi objek adalah *reusable* atau guna ulang, yaiut penggunaan program oleh program lain. Dengan menggunakan sifat ini proses pembangunan program menjadi lebih efektif karena tidak terjadi pengulangan penulisan kode program [1].



Gambar 2. Pengorganisasian data serta fungsi pada OOP

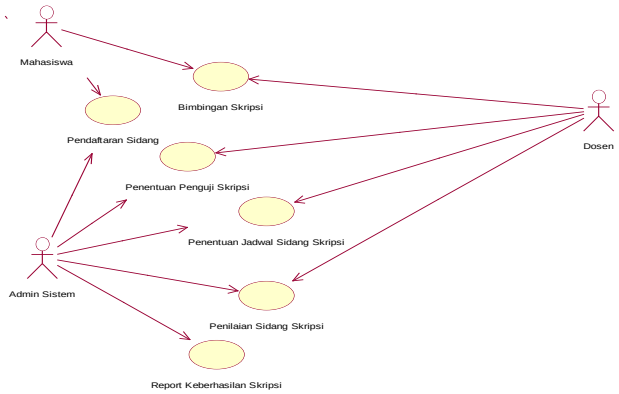
C. PEMBAHASAN

1. Analisis Sistem Alur Skripsi

Pembangunan sistem pada evaluasi keberhasilan skripsi dimulai dari penentuan masalah yang untuk ditemukan solusi jawabannya. Hal pokok yang ada di dalam evaluasi keberhasilan skripsi adalah sebagai berikut:

a. Ujian tugas akhir dilaksanakan setelah mahasiswa dinyatakan lulus dalam ujian komprehensif dan sudah menyelesaikan penulisan tugas akhir.
b. Ujian akhir dilaksanakan di hadapan majelis penguji yang terdiri dari penguji utama, ketua penguji dan sekretaris/pembimbing.
c. Sistem mencatat proses bimbingan skripsi.
d. Sistem dapat memberikan laporan tentang mahasiswa yang menyelesaikan skripsi dalam waktu 1 tahun.

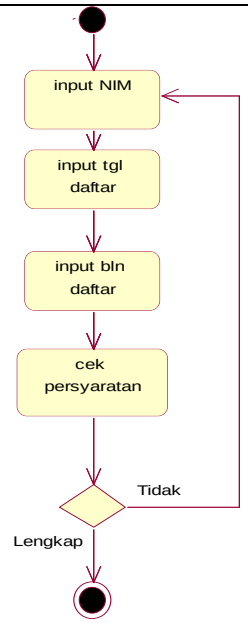
Untuk menjawab serta memecahkan hal tersebut maka dengan penggunaan UML sebagai jalan analisis pemecahan masalah, maka pada perancangan awal ditemukan perancangan *use case diagram* sebagai berikut:



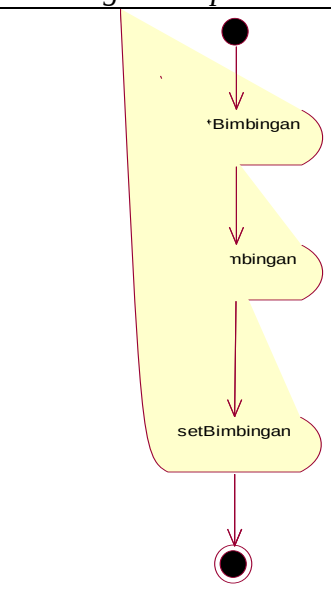
Gambar 3. ***Use case diagram***

Dari setiap *use case* tersebut dirancang flow *event*, *activity diagram*, hingga mempunyai class tersendiri untuk mempermudah pembacaan sistem melalui *use case* tersebut, serta *statechart*-nya. Pada fase analisis ini setiap *use case* akan dijelaskan lebih lanjut dengan menggunakan *flow of event* seperti yang dapat dilihat pada tabel-tabel berikut.

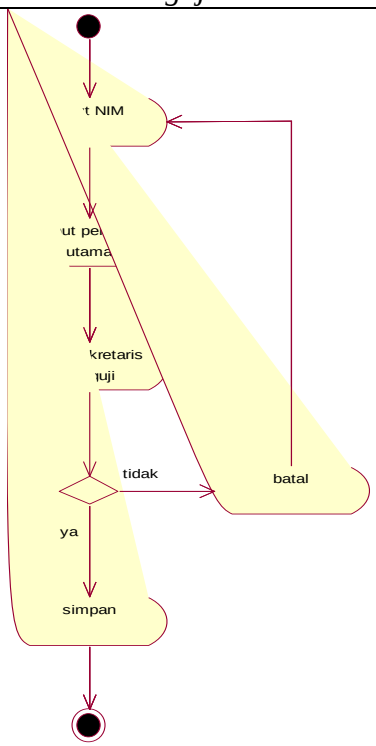
Tabel 1. *Flow of event* dari *use case* Daftar Sidang

1.0 *Flow of event dari use case Daftar Sidang*	*Activity Diagram use case Daftar Sidang*
1.1 *Preconditions* Mahasiswa telah selesai melakukan pembimbingan dan mengajukan sidang skripsi dengan mengumpulkan persyaratan-persyaratan yang telah ditentukan.	
1.2 *Main Flow* *Use case* ini dimulai ketika mahasiswa membawa pesyaratan-persyaratan yang diperlukan untuk bisa mengajukan sidang skripsi. Kasus yang muncul adalah E1: mahasiswa tidak melengkapi persyaratan	
1.3 *Subflows* Apabila proses pendaftaran sesuai ketentuan maka dapat segera ditentukan jadwal sidang skripsi.	
1.4 *Alternative Flows* E1 : bila mahasiswa tidak melengkapi persyaratan maka pengajuan sidang skripsi tidak diterima sampai mahasiswa tersebut melengkapi semua persyaratan.	

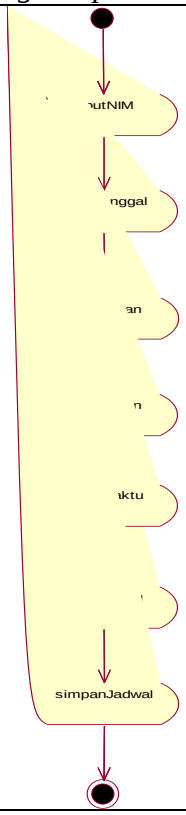
Tabel 2. *Flow of event* dari *use case* Bimbingan Skripsi

***Use case* Bimbingan Skripsi**	
1.0 Flow of event dari use case Bimbingan Skripsi	*Activity Diagram use case Bimbingan Skripsi*
1.1 *Preconditions* Mahasiswa telah melakukan ujian komprehensif dan telah dinyatakan lulus. Data judul skripsi telah ada dalam database beserta dosen pembimbing satu dan dosen pembimbing dua.	
1.2 *Main Flow* *Use case* ini dimulai ketika mahasiswa mulai melakukan proses pembimbingan. Tahapan pembimbingan adalah per-bab skripsi dan terdapat dua kemungkinan. Yaitu bab tersebut di ACC atau memerlukan revisi. Apabila di ACC maka dapat melanjutkan pembimbingan pada bab selanjutnya.	
1.3 *Subflows* Apabila proses pembimbingan lancar dan sampai pada selesainya pembuatan skripsi, maka ujian skripsi dapat segera dilaksanakan.	
1.4 *Alternative Flows*	

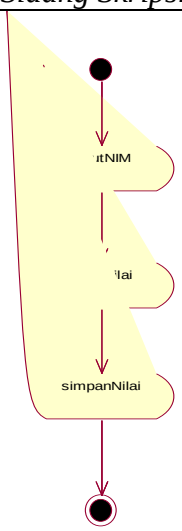
Tabel 3. *Flow of event* dari *use case* Penentuan Penguji

***Use case* Penentuan Penguji**	
1.0 Flow of event dari use case Penentuan Penguji	*Activity Diagram use case Penentuan Penguji*
1.1 *Preconditions* Sistem Yang berhubungan dengan ujian komprehensif telah diselesaikan, serta adanya data mahasiswa yang melakukan pembimbingan skripsi sehingga data mahasiswa yang akan melakukan ujian skripsi telah tersedia.	
1.2 *Main Flow* *Use case* ini dimulai dari admin memasukkan data mahasiswa ddari input NIM-nya terlebih dahulu, kemudian akan tampil data mahasiswa yang mengambil skripsi yang dimaksud berserta pembimbingnya. selanjutnya admin memasukkan input penguji utama dengan menginputkan id dari penguji yang kemudian akan muncul data penguji, lalu input sekretaris penguji yang selanjutnya melakukan penyimpanan untuk mengakhiri proses.	
1.3 *Subflows* Admin sistem dapat melakukan mengedit ulang penguji dan mahasiswa sebelum keluar atau menyimpan proses ke *database*.	
1.4 *Alternative Flows*	

Tabel 4. *Flow of event* dari *use case* Penentuan Jadwal Sidang Skripsi

***Use case* Penentuan Jadwal Sidang Skripsi**	
1.0 *Flow of event dari use case Penentuan Jadwal Sidang Skripsi*	*Activity Diagram use case Penentuan Jadwal Sidang Skripsi*
1.1 *Preconditions* Mahasiswa telah melakukan pembimbingan dan semua bab telah di ACC oleh dosen pembimbing masing-masing.	
1.2 *Main Flow* *Use case* ini dimulai ketika mahasiswa mengajukan ujian skripsi beserta persyaratannya kepada pihak jurusan. Kemudian pihak jurusan akan segera menetukan jadwal sidang skripsi mahasiswa tersebut.	
1.3 *Subflows* Apabila penjadwalan lancar maka akan diatur pelaksanaan sidang skripsi di jurusan dengan segala keperluannya, seperti pengumuman dan pemberitahuan pada dosen penguji.	
1.4 *Alternative Flows*	

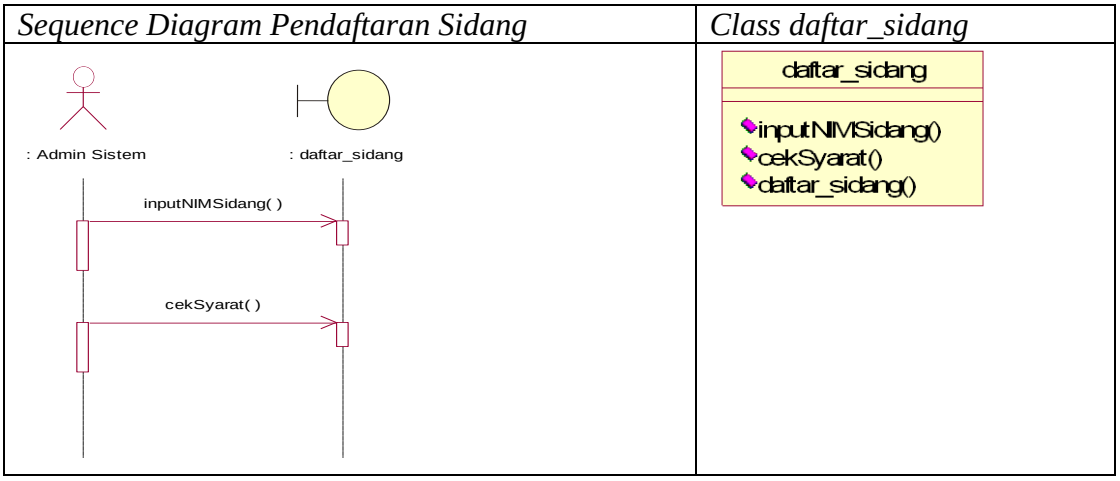
Tabel 5. *Flow of event* dari *use case* Penilaian Sidang Skripsi

***Use case* Penilaian Sidang Skripsi**	
1.0 *Flow of event dari use case Penilaian Sidang Skripsi*	*Activity Diagram use case Penilaian Sidang Skripsi*
1.1 *Preconditions* mahasiswa telah selesai melakukan sidang skripsi di jurusan yang dihadiri oleh semua dosen penguji dan mahasiswa tersebut.	
1.2 *Main Flow* *Use case* ini dimulai ketika penguji telah mempunyai nilai untuk mahasiswa yang telah selesai melaksanakan sidang skripsi, kemudian penguji akan memberikan penilaian untuk kemudian semua nilai penguji akan diakumulasikan untuk dihasilkan nilai akhir skripsi mahasiswa terkait.	
1.3 *Subflows* Apabila proses penilaian sidang skripsi berjalan sesuai aturan, maka mahasiswa terkait akan bisa segera melaksanakan yudisium sarjana.	
1.4 *Alternative Flows*	

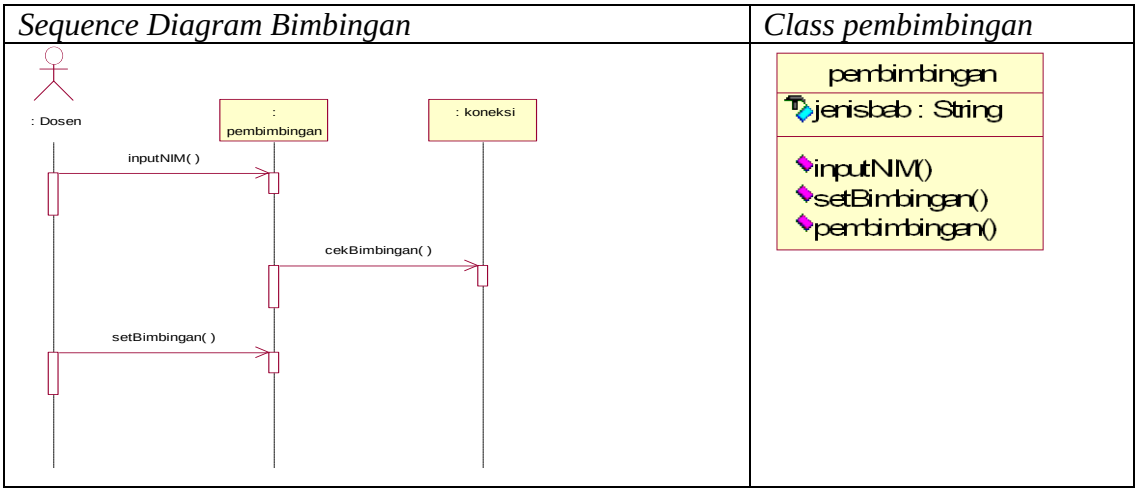
2. **Perancangan Sistem Sidang Skripsi**

Untuk efisiensi pada perancangan serta implementasi program nantinya dari setiap *use case* memiliki *class* tersendiri sebagi form tampilan pada pengguna sistem, sedangkan untuk koneksi akan memiliki class tersendiri sebagai class induk.

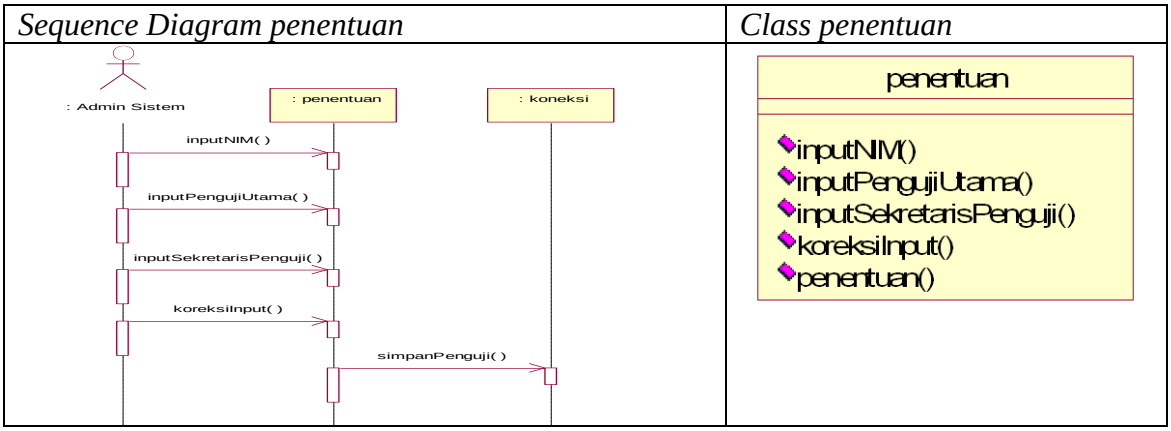
Dari analisis dari *use case* diagram di atas, maka akan dilanjutkan pada *sequence diagram* untuk perancangan alur *programming* di dalam *class* nantinya. Untuk gambaran *sequence diagram* serta *class* yang dihasilkannya adalah sebagai berikut:



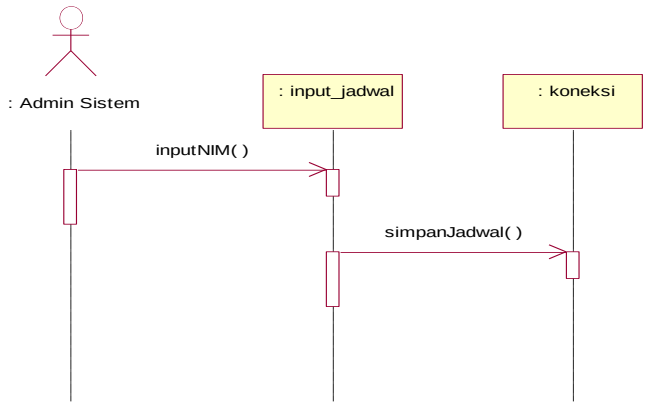
Gambar 4. *Sequence Diagram* dari *use case* Pendaftaran Sidang



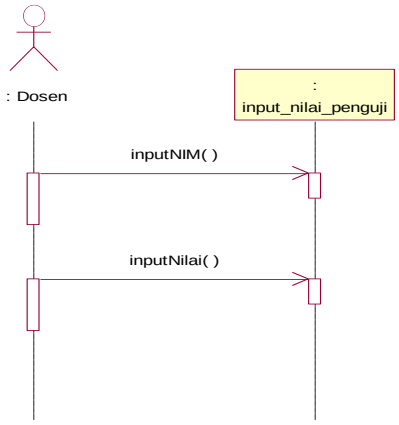
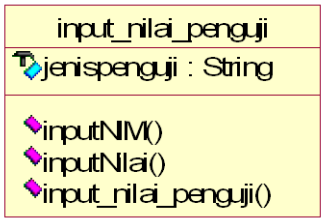
Gambar 5. *Sequence Diagram* Bimbingan Skripsi



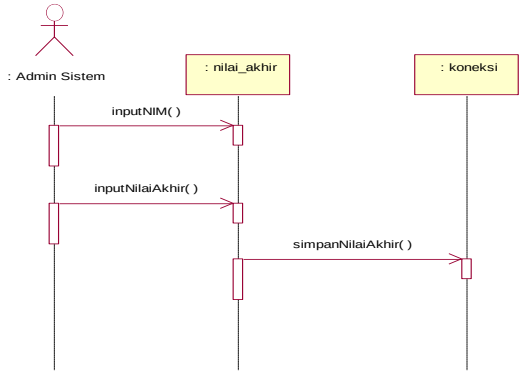
Gambar 6. *Sequence Diagram* Penentuan Penguji Skripsi

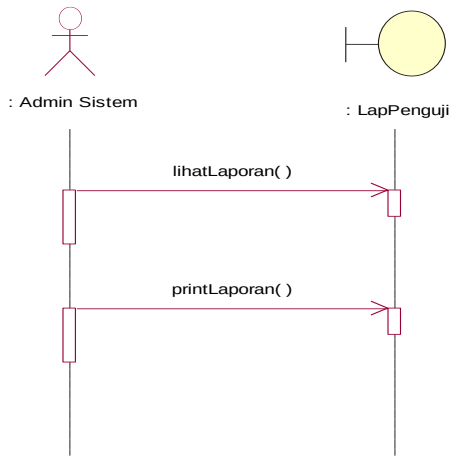
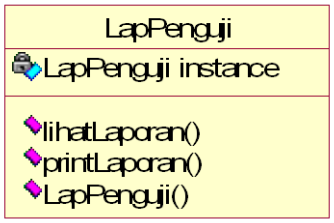
Sequence Diagram input_jadwal	*Class input_jadwal*
	

Gambar 7. *Sequence Diagram* Penentuan Jadwal Sidang Skripsi

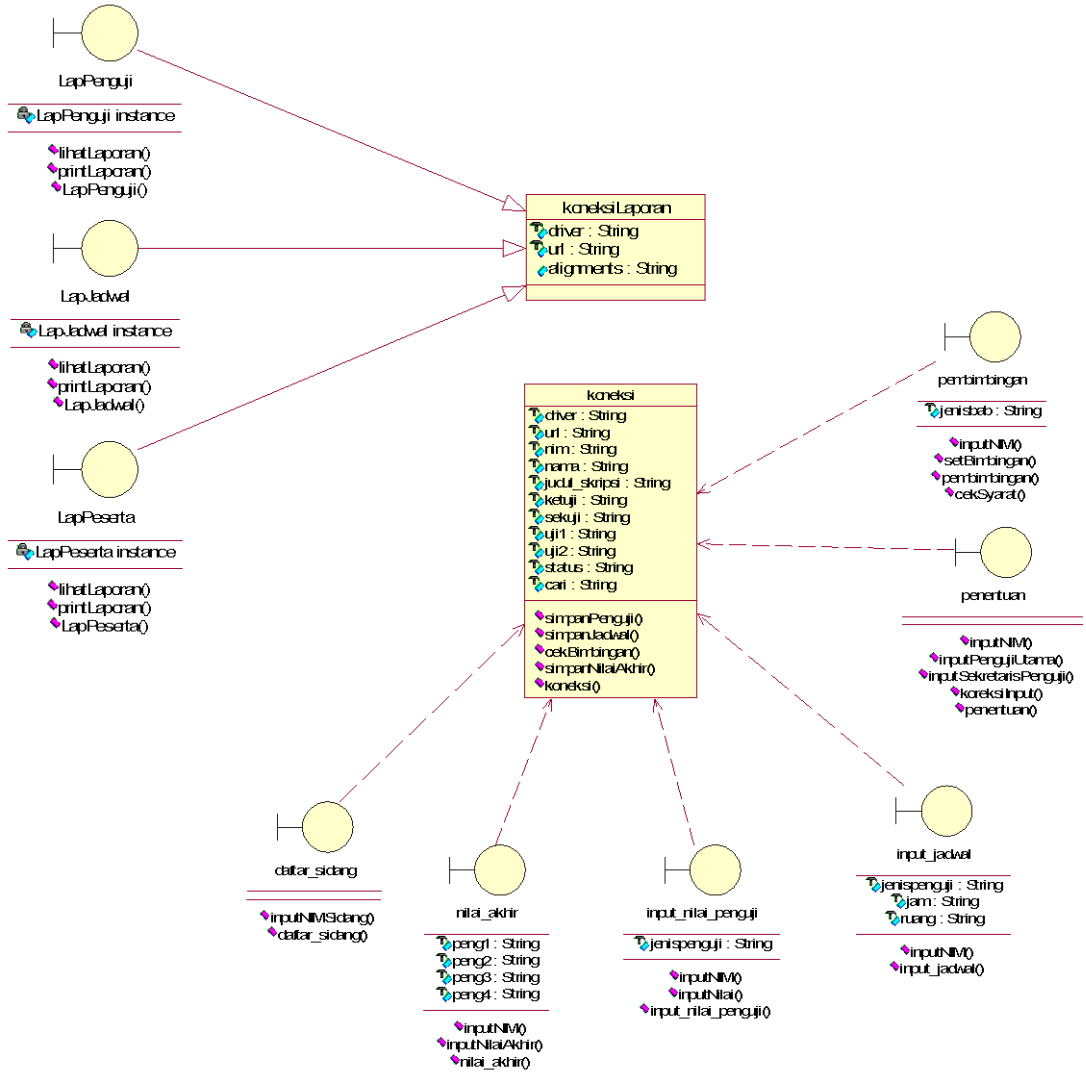
Sequence Diagram inputNilaiPenguji	*Class input_nilai_penguji*
	

Gambar 8. *Sequence Diagram* Input Nilai Penguji

Sequence Diagram inputNilaiAkhir	*Class nilai_akhir*
	

Sequence Diagram report-penguji	*Class LapPenguji*
	

Gambar 10. *Sequence Diagram* Laporan Penguji



Gambar 11. *Class Diagram*

Pada *report* atau laporan ini ada 3 *class* yakni LapPenguji, LapJadwal dan LapPeserta. *Method* yang ada dalam *class* hampir sama, namun *query* yang dipanggil

di setiap *class* yang berbeda. Ketiga *class* ini mempunyai *class* induk koneksiLaporan.

Pada *Class Diagram* di atas menunjukkan adanya relasi antar *class*. *Class* koneksi dan *class* koneksiLaporan dipergunakan sebagai *class* induk. *Class* koneksiLaporan merupakan *class* induk dari *class* yang berhubungan dengan laporan yang akan dicetak dalam sistem ini. Sedang *class* koneksi merupakan *class* induk dari *class* yang merupakan *class* input data dan pemrosesan data dalam sistem ini. Dengan adanya *class* induk ini, penggunaan atribut serta *method* dalam *class diagram* akan lebih efisien sehingga tak menulis atau mendeklarasikan lagi di tiap *class*.

3. Implementasi Pemrograman pada Sistem Sidang Skripsi

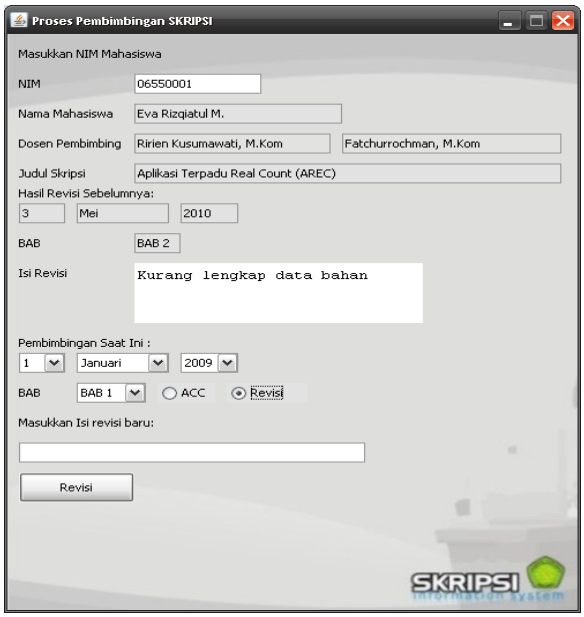
Pada implementasi pemrograman dari analisis dan perancangan ini pertama kali tampilan utama hasil program ini adalah sebagi berikut:

Gambar 12. Tampilan Utama

Pada tampilan di atas terdapat 3 menu utama dan 1 menu tambahan. Menu utama tampilan dia tas adalah Pra-Ujian, Proses Skripsi dan juga Laporan; Sedangkan menu tambahan adalah menu *Help* yang berisi petunjuk penggunaan dan menu *About* yang berisi informasi pembangun aplikasi. Pada menu Pra-Ujian terdapat 4 sub-menu, yakni :

1. Pembimbingan Skripsi

Menu ini berasal dari class pembimbingan, yang digunakan sebagai *form* pembimbingan dari tiap pembimbing agar lebih mudah melakukakn pemantauan pada mahasiswa bimbingannya. Tampilannya adalah sebagai berikut:



Gambar 13. Tampilan Pembimbingan Skripsi

Pada gambar, pertama kali proses dimulai dengan memasukkan NIM mahasiswa yang kemudian program akan mengeluarkan data mahasiswa yang melakukan proses bimbingan sebelumnya sesuai dengan data NIM yang dimasukkan.

Selanjutnya pembimbing akan melihat *history* bimbingan sebelumnya yang kemudian memeriksa apa yang mahasiswa bimbingannya konsultasikan, kemudian pembimbing akan memasukkan waktu bimbingan pada hari itu selanjutnya memeriksa bab yang dikonsultasikan apakah telah sesuai yang kemudian di-*acc* (dengan menekan *radiobutton* ACC) atau revisi (dengan menekan *radiobutton* Revisi) dengan memasukkan isi revisinya.

Gambar 14. Tampilan Pendaftaran Skripsi

2. Pendaftaran Sidang

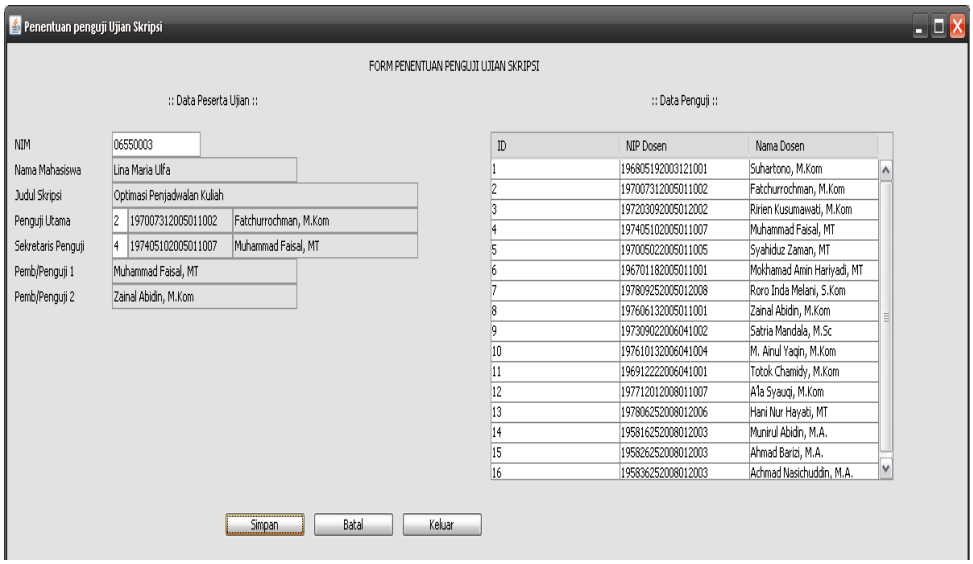
Pendaftaran Sidang adalah menu untuk memasukkan data mahasiswa yang mendaftar ujian sidang skripsi.

Pada *form* Pendaftaran sidang Skripsi ini dimulai dengan memasukkan NIM mahasiswa yang mendaftar. Kemudian akan ditampilkan data mahasiswa. Selanjutnya petugas memeriksa syarat yang disetorkan mahasiswa yang mendaftar.

Selanjutnya, jika telah dimaukkan semua dengan memeriksanya maka petugas dapat menyimpan dengan melakukan *update* data pada *database* sistem.

3. Penentuan Penguji Skripsi

Penentuan penguji skripsi *adalah* sub-menu untuk memasukkan data penguji skripsi mahasiswa yang akan melakukan ujian skripsi. Pertama kali dimulai dengan memasukkan NIM mahasiswa, kemudian program akan memunculkan data mahasiswa beserta pembimbing 1 dan pembimbing 2 yang akan masuk dalam penguji 1 dan penguji 2.

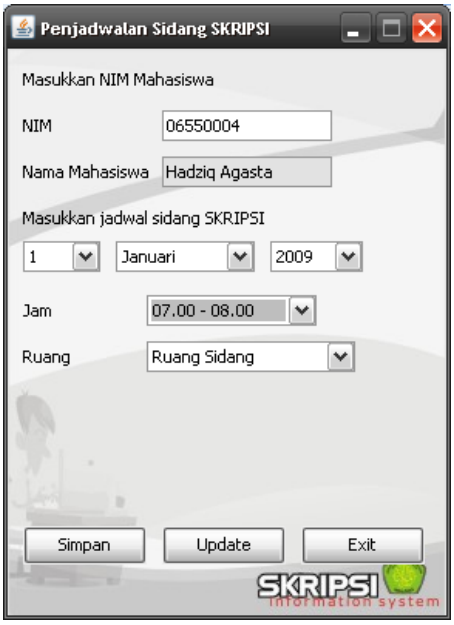


Gambar 15. Tampilan Penentuan Penguji Skripsi

Selanjutnya setelah tampil data mahasiswa, maka akan dibarengi dengan data dosen. Data dosen digunakan untuk membantu admin sistem melakukan masukan data hanya tinggal memasukkan id dosen sesuai tampilan pada tabel data dosen. Setelah awal memasukkan NIM, program akan langsung menunkukkan *cursor* pada ketua penguji, admin sistem selanjutnya hanya tinggal memasukkan id dosen lalu akan tampil data dosen sesuai dosen yang dipilih. Setelah itu memilih sekretaris penguji juga sama. Jika data telah benar, maka bisa langsung disimpan dengan menekan tombol simpan.

4. Penjadwalan Ujian Skripsi

Penjadwalan Skripsi dimulai dengan memasukkan NIM mahasiswa yang akan melakukan penjadwalan yang kemudian akan tampil data nama mahasiswa. Selanjutnya, petugas akan memasukkan waktu dan ruang yang akan digunakan ujian skripsi bagi mahasiswa tersebut.



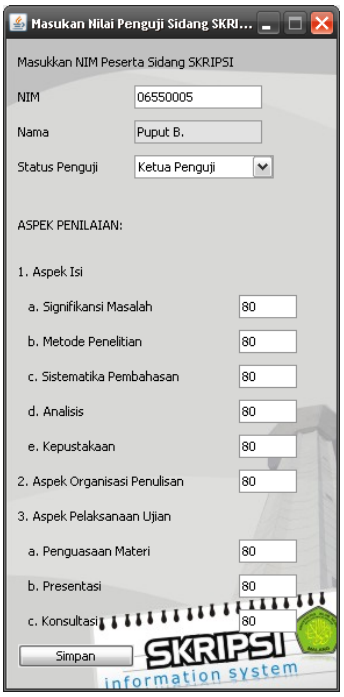
Gambar 16. Tampilan Penjadwalan Ujian Skripsi

Pada Menu Proses Skripsi terdapat 2 sub-menu, yang akan dijelaskan sebagi berikut:

a. Input Nilai Penguji

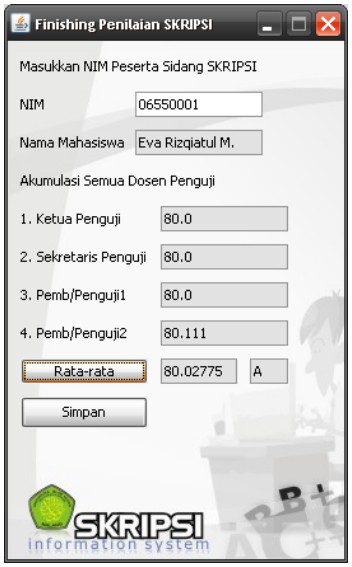
Input Nilai Penguji digunakan oleh dosen penguji untuk memasukkan nilainya bagi mahasiswa yang diujinya. Pertama kali dosen penguji diminta untuk memasukkan NIM Mahasiswa yang diuji yang kemudian program akan mengeluarkan data mahasiswa. Selanjutnya dosen penguji akan memilih status penguji. Status penguji disini ada 4 yakni: Ketua Penguji, Sekretaris Penguji, Penguji 1 dan yang terahir Penguji 2.

Berikutnya dosen penguji dapat memasukkan nilai dari 3 aspek standar yang telah ditentukan, kemudian setelah semuanya terisi dapat menyimpan dengan tombol simpan.



Gambar 17. Tampilan Input Nilai Penguji

b. Input Nilai Akhir

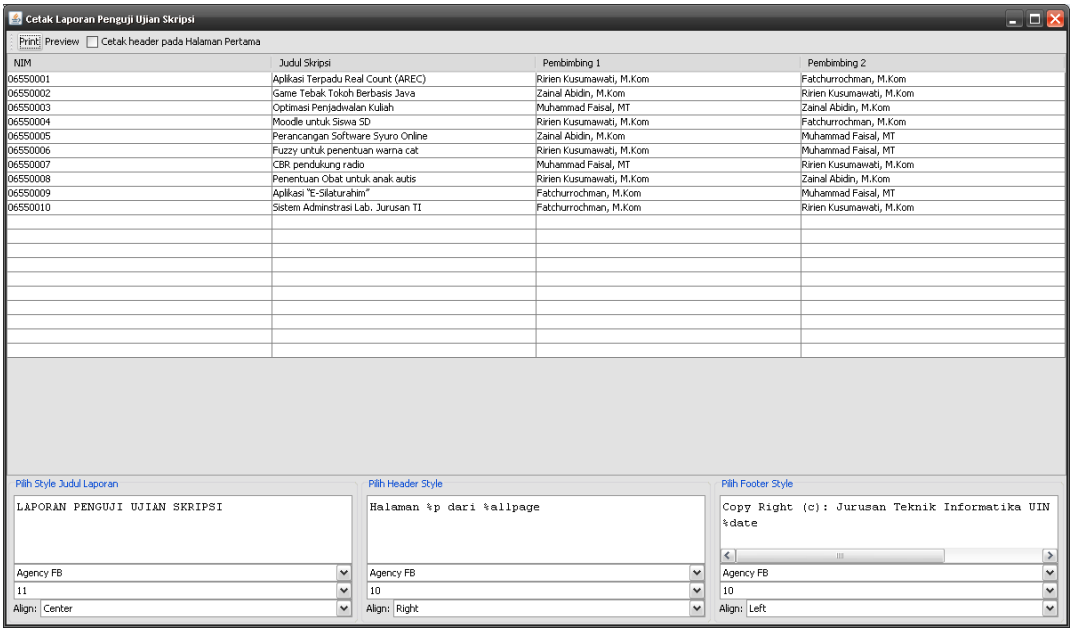


Gambar 18. Tampilan Input Nilai Akhir

Input Nilai Akhir adalah sub-menu yang digunakan admin sistem atau panitia skripsi sebagai masukan nilai akhir maha siswa yang telah malakukan ujian sidang skripsi.Pada Menu Utama yang ke-3 adalah menu Laporan, menu ini memiliki 3 sub-menu laporan yakni:

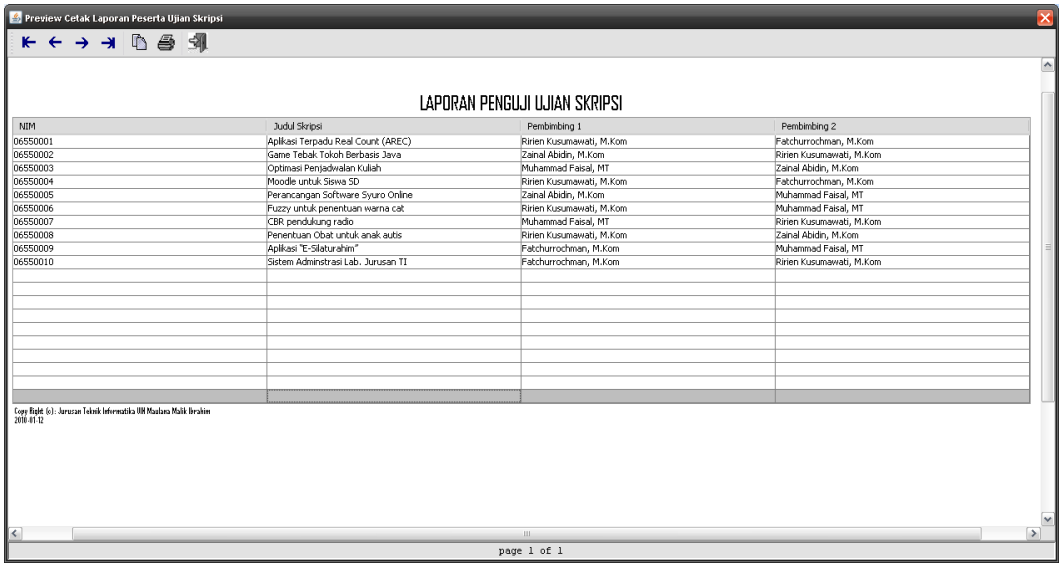
1. Laporan Penguji
2. Laporan Jadwal
3. Laporan Daftar Mahasiswa Skripsi

Tampilan dari ketiga sub-menu tersebut hampir sama, hanya isi dari laporan yang membedakan



Gambar 19. Tampilan Pengaturan Laporan

Petugas sebelum mencetak dari laporan dapat mengatur judul laporan, *header style*, serta *footer* dari laporan. Jika telah sesuai keinginan petugas dapat melakukan 2 aksi, yakni langsung mencetak laporan dengan menekan *button print* atau *preview* untuk melihat data laporan yang akan dilakukan untuk dicetak. Tampilannya sebagai berikut:



Gambar 20. Tampilan Laporan

D. PENUTUP

1. Kesimpulan

Berdasarkan hasil analisis, perancangan, dan pemrograman sistem sidang skripsi maka dapat diambil kesimpulan sebagai berikut:

a. Analisis secara *object oriented* menghasilkan pandangan logik (*logical view*) terhadap sistem sidang skripsi yang menggambarkan *actor* dan *use case model*.
b. Perancangan secara *object oriented* melahirkan kelas-kelas dan objek-objek yang secara langsung merefleksikan domain masalah dan tanggungjawab sistem sidang skripsi.
c. Pemrograman secara *object oriented* mengandung tiga dasar pendekatan *object oriented* program, yaitu adanya objek, *polymorphism*, dan pewarisan (*inheritance*).
d. Pendekatan *object oriented* terbukti menghasilkan produk yang bersifat lebih *reusable, easily maintenance, and easily development.*

2. Saran

Adapun saran yang dapat diberikan dari hasil penelitian yang telah dilaksanakan untuk penelitian hal yang sama berikutnya adalah sebagai berikut:

a. Perlunya memperhatikan sistematika dalam langkah *object oriented method*, karena akan berpengaruh terhadap tingkat OOP aplikasi yang kita bangun.
b. Sebaiknya memperhatikan kelengkapan atribut-atribut dan fungsi-fungsi terkecil pada aplikasi yang kita bangun selama masih bersangkutan dan diperlukan sistem.
c. Pentingnya implementasi OOP pada semua aplikasi yang kita rancang dan bangun di masa depan, karena merupakan metode hasil perpaduan beberapa metode yang sebidang.

DAFTAR PUSTAKA

[1]Fatchurrochman. 2008. *Pemrograman Berorientasi Objek dengan Bahasa Java*. UIN Press. Malang

[2]Hariyanto, Bambang. 2004. *Rekayasa Sistem Berorientasi Objek*. Penerbit Informatika. Bandung

[3]Nugroho, Adi. 2004. *Pemrograman Berorientasi Objek*. Penerbit Informatika. Bandung

[4]Sholiq. 2006. *Pemodelan Sistem Informasi Berorientasi Objek dengan UML*. Graha Ilmu. Yogyakarta.