Implementasi *Reverse Engineering* terhadap Perangkat Lunak untuk Melakukan *Reengineering* dengan Paradigma Prosedural menjadi Paradigma Berorientasi Objek

TUGAS AKHIR

Oleh :

Esti Rizki	**3311001054**
Maulidinata T.S	**3311001073**

Disusun untuk memenuhi syarat kelulusan Program Diploma III

PROGRAM STUDI TEKNIK INFORMATIKA

POLITEKNIK NEGERI BATAM

BATAM

2012

LEMBAR PENGESAHAN

Batam, …….. 2012

Pembimbing,

Metta Santiputri, M.Sc

NIK. 100017

LEMBAR PERNYATAAN

Dengan ini, saya:

NIM : 3311001054

Nama : Esti Rizki

adalah mahasiswa Teknik Informatika Politeknik Batam yang menyatakan bahwa tugas akhir dengan judul:

Implementasi *Reverse engineering* terhadap Perangkat Lunak untuk melakukan *Reengineering* dengan Paradigma Prosedural menjadi Paradigma Berorientasi Objek

disusun dengan:

1. tidak melakukan plagiat terhadap naskah karya orang lain
2. tidak melakukan pemalsuan data
3. tidak menggunakan karya orang lain tanpa menyebut sumber asli atau tanpa ijin pemilik

Jika kemudian terbukti terjadi pelanggaran terhadap pernyataan di atas, maka saya bersedia menerima sanksi apapun termasuk pencabutan gelar akademik.

Lembar pernyataan ini juga memberikan hak kepada Politeknik Batam untuk mempergunakan, mendistribusikan ataupun memproduksi ulang seluruh hasil Tugas Akhir ini.

Batam, 20 februari 2013

Esti Rizki
3311001054

LEMBAR PERNYATAAN

Dengan ini, saya:

NIM : 3311001073

Nama : Maulidinata Tri Susman

adalah mahasiswa Teknik Informatika Politeknik Batam yang menyatakan bahwa tugas akhir dengan judul:

Implementasi *Reverse engineering* terhadap Perangkat Lunak untuk melakukan *Reengineering* dengan Paradigma Prosedural menjadi Paradigma Berorientasi Objek

disusun dengan:

1. tidak melakukan plagiat terhadap naskah karya orang lain
2. tidak melakukan pemalsuan data
3. tidak menggunakan karya orang lain tanpa menyebut sumber asli atau tanpa ijin pemilik

Jika kemudian terbukti terjadi pelanggaran terhadap pernyataan di atas, maka saya bersedia menerima sanksi apapun termasuk pencabutan gelar akademik.

Lembar pernyataan ini juga memberikan hak kepada Politeknik Batam untuk mempergunakan, mendistribusikan ataupun memproduksi ulang seluruh hasil Tugas Akhir ini.

Batam, 20 februari 2013

Maulidinata Tri Susman
3311001073

KATA PENGANTAR

Dengan mengucapkan puji syukur kehadirat Tuhan Yang Maha Esa yang telah memberikan rahmat, hidayah dan karuniaNya sehingga kami dapat menyelesaikan laporan Tugas Akhir ini.

laporan ini kami susun untuk menyelesaikan masa pendidikan pada jurusan Teknik Informatika di Politeknik Negeri Batam. Dalam penyusunan laporan ini kami menyadari banyak sekali kekurangan, namun berhubung banyaknya pihak luar yang mendukung dan turut membantu, sehingga laporan ini dapat diselesaikan. Oleh karena itu dalam kesempatan ini kami sangat bersyukur dan berterima kasih kepada semua pihak yang telah membantu kelancaran laporan ini.

Laporan ini dapat terselesaikan dengan adanya bantuan dari pihak pembimbing materi maupun teknis, oleh karena itu kami mengucapkan terimakasih kepada :

1. Tuhan Yang Maha Esa sehingga kami dapat menyelesaikan Laporan Tugas Akhir ini.
2. Orang Tua dan keluarga kami yang telah memberikan dorongan moril.
3. Ibu Metta Santiputri, M.Sc selaku dosen pembimbing.
4. Bapak Ari Wibowo, MT selaku dosen pengampu.
5. Bapak Tri Ramadhani Ardjo, S.ST, Bapak Riwinoto, Bapak Tandhi, Ibu Yeni selaku dosen penguji.
6. Teman-teman satu kelas yang selalu mendukung.
7. Pihak-pihak lain yang banyak membantu dan memberikan support.

Batam, Februari 2013

Penulis

ABSTRAK

Implementasi *Reverse engineering* terhadap Perangkat Lunak untuk melakukan *Reengineering* dengan Paradigma Prosedural menjadi Paradigma Berorientasi Objek

Salah satu tujuan dari rekayasa perangkat lunak modern adalah untuk berubah menjadi paradigma dengan sistem yang revolusioner. *Reengineering* merupakan cara untuk migrasi atau pindah menjadi sistem pewarisan yang lebih revolusioner.

Reverse engineering adalah proses menganalisa sebuah sistem untuk mengidentifikasi komponen dari sistem dan hubungan di dalamnya untuk membuat representasi dari sistem di dalam sebuah form atau di dalam abstraksi yang lebih tinggi. *Software reengineering* adalah proses pengujian, pemahaman dan perubahan dari suatu sistem dengan tujuan untuk mengimplementasikan dalam bentuk yang baru. *Reengineering* merupakan solusi yang lebih baik untuk penanganan kode pewarisan jika dibandingkan dengan *re-developing* atau pengembangan ulang perangkat lunak dari kebutuhan aslinya.

Garis besar tugas akhir ini berisi mengenai perubahan sistem dari paradigma prosedural menjadi sistem berparadigma objek, dimana cara melakukan migrasi tersebut menggunakan *reverse engineering* dan *reengineering*. Hal ini dilakukan untuk membuktikan apakah bagian dari sistem yang lama dapat dipakai kembali pada sistem baru yang berbasis objek.

Kata Kunci: sistem pewarisan, reengineering, reverse engineering, software revolutioner

ABSTRACT

Implementasi *Reverse engineering* terhadap Perangkat Lunak untuk melakukan *Reengineering* dengan Paradigma Prosedural menjadi Paradigma Berorientasi Objek

One of the goals of modern software engineering is to move towards a paradigm of evolutionary systems. *Reengineering* offers an approach to migrate a legacy system towards an evolvable system in disciplined manners.

Software reverse engineering is the process of analyzing the components and component interrelationships of a software system in order to describe that system at a level of abstraction higher than that of the original system. Reengineering is the process of examination, understanding, and alteration of a system with the intent of implementing the system in a new form. Software reengineering is considered to be a better solution for handling legacy code when compared to re-developing software from the original requirements.

This final project presents a case study that illustrates how a combination of object-oriented (OO) and procedural techniques can be used in tandem to reengineer an existing application (in this case the e-Procedure software of State Batam Polytechnic). And to further prove –which part of the procedural system can be reused in the new object oriented system.

Key words: legacy system, reengineering, reverse engineering, revolutionarry software.

DAFTAR ISI

Bab I Pendahuluan 1
I.1 Latar Belakang 1
I.2 Rumusan Masalah 2
I.3 Batasan Masalah 2
I.4 Tujuan 2
I.5 Sistematika Penulisan 3
Bab II Landasan Teori 4
II.1 Reverse engineering 4
II.2 Reengineering 4
II.3 Penelitian Sebelumnya 5
II.3.1 Studi kasus yang dilakukan oleh Gannod (1995) 5
II.3.2 Studi kasus yang dilakukan oleh Frakes 8
II.4 Studi Kasus 9
Bab III Reverse engineering 10
III.1 Deskripsi Fungsional Sistem 11
III.2 Deskripsi Data 11
III.3 Context Diagram dan DFD 13
III.3.1 DFD Level 1 13
III.4 ER Diagram 14
III.5 Dekomposisi Fungsional Modul 14
III.6 Algoritma 14
III.7 Layar pada Aplikasi 15
III.8 Tabel Keterunutan 15
Bab IV Reeinginering 16
IV.1 Usecase 16
IV.2 Robustness Diagram 17
IV.3 Sequence Diagram 17
IV.4 Diagram Kelas 17
IV.5 Algoritma 18

IV.6 Tabel Keterunutan 18
Bab V Hasil dan Pembahasan 21
V.1 Implementasi 21
V.2 Pengujian 22
V.2.1 Skenario Pengujian 22
Bab VI Kesimpulan dan Saran 27
VI.1 Kesimpulan 27
VI.2 Saran 27
Lampiran A Deteksi Proses Awal dan DFD Level 2 dan 3 28
A.1 Deteksi Proses Awal 28
A.2 Pengelompokan Proses 28
A.3 DFD Level 2 Proses 1 Pengajuan Borang 29
A.4 DFD Level 2 Proses 2 Pemrosesan Borang 29
A.5 DFD Level 2 Proses 3 Pengaturan Prosedur dan Borang 30
A.6 DFD Level 3 Proses 3.4 Pengaturan Borang 30
Lampiran B Dekomposisi Fungsional Modul 31
Lampiran C Algoritma 34
C.1 Algoritma Proses pengajuan borang 34
C.2 Algoritma pengecekan status borang 36
C.3 Algoritma meneruskan pengajuan borang 37
C.4 Algoritma pemrosesan borang 37
C.5 Algoritma melihat statistik 39
C.6 Algoritma pengaturan prosedur dan borang 40
B.6.1 Algoritma pengaturan prosedur 40
C.6.2 Algoritma pengaturan borang 42
Lampiran D Layar Tampilan 45
D.1 Layar Utama 45
D.2 Layar Proses pengajuan borang 45
D.3 Layar Pengecekan Status Borang 46
D.4 Layar Penerusan pengajuan borang 47
D.5 Layar pemrosesan borang 48

D.6 Layar Statistik 49
D.7 Pengaturan Prosedur dan Borang 50
D.7.1 Layar Lihat Prosedur 50
D.7.2 Layar Tambah Prosedur 51
D.7.3 Layar Edit Prosedur 52
D.7.4 Layar Delete Prosedur 53
D.7.5 Layar Lihat Borang 54
D.7.6 Layar Tambah Borang 55
D.7.7 Layar Edit Borang 56
D.7.8 Layar Delete Borang 57
Lampiran E Robustness Diagram 59
E.1 Robutness Diagram Pengajuan Borang 59
E.2 Robustness Diagram Pengecekan Status 59
E.3 Robustness Diagram Pemrosesan Borang 60
E.4 Robustness Diagram Pengecekan Statistik 60
E.5 Robustness Diagram Pengaturan Prosedur dan Borang 60
E.5.1 Robustness Diagram Pengaturan Prosedur 60
E.5.2 Robustness Diagram Pengaturan Borang 61
Lampiran F Sequence Diagram 62
F.1 Sequence Diagram untuk Use Case Pengajuan Borang 62
F.2 Sequence Diagram untuk Use Case Pengecekan Status Borang 63
F.3 Sequence Diagram untuk Use Case Penerusan Pengajuan 63
F.4 Sequence Diagram untuk Use Case Pemrosesan Borang 64
F.5 Sequence Diagram untuk Use Case Pengecekan Statistik 64
F.7 Sequence Diagram untuk Use Case Pengaturan Prosedur dan Borang 65
F.7.1 Sequence Diagram untuk Use Case Tambah Prosedur 65
F.7.2 Sequence Diagram untuk Use Case Ubah Prosedur 65
F.7.3 Sequence Diagram untuk Use Case Delete Prosedur 66
F.7.4 Sequence Diagram untuk Use Case Tambah Borang 66
F.7.5 Sequence Diagram untuk Use Case Ubah Borang 67
F.7.6 Sequence Diagram untuk Use Case Hapus Borang 67

Lampiran G Diagram Kelas 68
G.1 kelas GUI pengajuan borang..... 68
G.2 kelas GUI pengecekan status borang 68
G.3 kelas GUI borang masuk..... 68
G.4 kelas GUI penerusan pengajuan borang 68
G.5 kelas GUI lihat statistik pengajuan borang 68
G.6 kelas GUI lihat daftar prosedur..... 68
G.7 kelas GUI tambah prosedur 69
G.8 kelas GUI edit prosedur 69
G.9 kelas GUI delete prosedur..... 69
G.9 kelas GUI lihat daftar borang..... 69
G.10 kelas GUI edit borang 69
G.11 kelas GUI delete borang..... 69
G.12 kelas pengajuan borang..... 69
G.13 kelas pengguna..... 70
G.14 kelas borang 70
G.15 kelas prosedur 70
Lampiran H Source Code..... 71
Lampiran I Skenario Pengujian..... 74

DAFTAR GAMBAR

Gambar 1 Alur Kerja 10
Gambar 2 Context Diagram eProsedur 13
Gambar 3 DFD Level 1 13
Gambar 4 ER Diagram 14
Gambar 5 Layar Utama Pengguna Personal 45
Gambar 6 layar upload borang 46
Gambar 7 Layar Lihat Status Borang setelah tanggal dipilih 47
Gambar 8 Contoh borang dengan status Disetujui 47
Gambar 9 Pengajuan borang kepada pihak lain 48
Gambar 10 Layar Lihat Borang Masuk 49
Gambar 11 Layar Lihat Statistik 50
Gambar 12 Layar Lihat Prosedur 51
Gambar 13 Layar Tambah Prosedur 52
Gambar 14 Layar Edit Prosedur 53
Gambar 15 Layar Delete Prosedur 54
Gambar 16 Layar Lihat Borang 55
Gambar 17 Layar Tambah Borang 56
Gambar 18 Layar Edit Borang 57
Gambar 19 Layar Delete Borang 58

DAFTAR TABEL

Tabel 1 tabel keterunutan 15
Tabel 2 tabel keterunutan *reengineering* 19

Bab I Pendahuluan

I.1 Latar Belakang

Menurut buku "*Software Engineering Institute"* (1995) bahwa sebagian besar perangkat lunak dibangun dengan asumsi bahwa perangkat lunak tersebut dapat dirawat dalam kurun waktu tertentu atau dalam beberapa hal digantikan dengan sistem yang baru. Selain itu, Hanna (1993), menyatakan bahwa perawatan perangkat lunak *(software maintenance*) merupakan fase paling mahal dalam siklus hidup perangkat lunak. Hampir 60% dari pengembangan perangkat lunak merupakan fase perawatan. Suksesnya pemeliharaan perangkat lunak sebuah sistem bergantung pada dokumentasi yang akurat dari desain sistem itu sendiri. Pada beberapa kasus, perangkat lunak dan dokumentasi tidak konsisten. Desain jarang diperbaharui untuk merefleksikan modifikasi yang dilakukan pada sistem. Pada kasus lain, sistem tidak memiliki dokumentasi dan dasar pemikiran di balik keputusan dalam implementasi sistem. Dalam kasus manapun, baik kehilangan atau kekurangan konsistensi desain memiliki dampak besar pada efektivitas dari upaya untuk memelihara dan memodifikasi sistem yang sudah ada. Proses pengembangan memakan banyak total waktu untuk melakukan perubahan pada sistem karena kompleksitas sistem, tidak adanya pengetahuan mengenai sistem, keterbatasan pengetahuan mengenai sistem dan tidak adanya dokumentasi pengujian.

Kebutuhan merancang-bangun perangkat lunak ulang atau *reenginerering* sudah meningkat, yakni ketika sistem perangkat lunak yang ada telah menjadi usang baik dalam hal arsitektur, platform, stabilitas maupun kesesuaian untuk mendukung evolusi kebutuhan serta hilangnya keberadaan dokumentasi sistem. *Software reengineering* penting untuk memulihkan kembali perangkat lunak yang ada, tingginya biaya pemeliharaan dibawah kontrol serta pembentukan dasar evolusi untuk mendukung kebutuhan di masa depan menjadi hal utama. Untuk melakukan *reengineering* dari sistem yang sudah ada, maka sebelumnya perlu melakukan proses *reverse engineering*.

Reverse engineering ditujukan untuk memulihkan dan mencatat informasi tingkat tinggi tentang sistem atau informasi yang dimengerti manusia. Setelah

dilakukan *reverse engineering*, komponen-komponen sistem dan fungsional diketahui, maka dilakukanlah *reengineering* untuk membangun kembali sistem baru dengan paradigma berorientasi objek. Menurut Cross (1993), *reeingineering* adalah proses pengujian, pemahaman dan perubahan dari suatu sistem dengan tujuan untuk mengimplementasikan dalam bentuk yang baru. *Reengineering* merupakan solusi yang lebih baik untuk penanganan pewarisan kode jika dibandingkan dengan *re-developing* atau pengembangan ulang perangkat lunak dari kebutuhan aslinya.

I.2 Rumusan Masalah

Beberapa rumusan masalah yang akan dibahas dalam Tugas Akhir ini adalah sebagai berikut :

1. Bagaimana *reverse engineering* digunakan untuk membangun kembali sistem untuk memenuhi kebutuhan sistem yang baru ?
2. Apakah bagian pada sistem lama yang berorientasi prosedural dapat dipakai kembali pada sistem baru yang berorientasi objek ?

I.3 Batasan Masalah

Batasan masalah yang akan dibahas dalam Tugas Akhir ini adalah mengimplementasikan *reverse engineering* dan *reeingineening* perangkat lunak dengan fungsi yang sama tanpa ada penambahan fungsi baru pada sistem, bukan membangun sistem baru dengan fungsi yang berbeda.

I.4 Tujuan

Tujuan dari penelitian ini adalah:

1. Mengetahui bagaimana cara mengimplementasikan *reverse engineering* untuk melakukan *reengineering* perangkat lunak dari sistem berorientasi prosedural menjadi objek.
2. Mengetahui apakah bagian pada sistem lama yang berbasis prosedural dapat dipakai kembali pada sistem baru yang berorientasi objek setelah perangkat lunak dimigrasi.

I.5 Sistematika Penulisan

Dokumen ini dibagi menjadi enam bagian utama, yaitu :

1. Bab I berisi penjelasan tentang latar belakang masalah, lingkup masalah yang dipengaruhi dan sistematika.
2. Bab II berisi landasan teori yang berisi penjelasan secara umum mengenai *reverse engineering* dan *reenginering* serta referensi tulisan ilmiah.
3. Bab III berisi penjelasan mengenai *reverse engineering* pada aplikasi eProsedur.
4. Bab IV berisi penjelasan mengenai *reengineering* perangkat lunak eProsedur.
5. Bab V berisi penjelasan mengenai implementasi
6. Bab VI berisi kesimpulan dan saran

Bab II Landasan Teori

II.1 Reverse engineering

Menurut *Chikofskyand Cross [in Arnold, 1993], Reverse engineering* adalah sebuah proses menganalisa sebuah sistem untuk diidentifikasi komponen dari sistem dan hubungan di dalamnya juga membuat representasi dari sistem di dalam sebuah form atau di dalam abstraksi yang lebih tinggi.

Dalam bahasa yang lebih mudah, *reverse engineering* adalah suatu proses menganalisa sistem dari suatu subjek (dalam hal ini *software*) untuk mempelajari cara kerja, teknik apa yang digunakan dan sebagainya, sehingga bisa mencapai target yang di inginkan oleh programmer.

Pada umumnya, programmer membuat suatu program berbentuk *executable* dari *source code*, tetapi dengan *reverse engineering* programmer membongkar sesuatu yang *excutable* menjadi sebuah *source code*.

II.2 Reengineering

Menurut Cross (1993), *Reeingineering* adalah proses pengujian, pemahaman dan perubahan dari suatu sistem dengan tujuan untuk mengimplementasikan dalam bentuk yang baru. *Reengineering* merupakan solusi yang lebih baik untuk penanganan kode pewarisan jika dibandingkan dengan *re-developing* atau pengembangan ulang perangkat lunak dari kebutuhan aslinya. *Reengineering* menawarkan sebuah pendekatan untuk migrasi sistem warisan terhadap sistem *evolvable* secara disiplin. Proses rekayasa ulang dapat dipandang sebagai penerapan prinsip-prinsip rekayasa untuk sistem yang ada untuk memenuhi persyaratan baru. Namun agar sukses, rekayasa ulang membutuhkan wawasan dari sejumlah perspektif yang berbeda.

Menurut buku "*Software Engineering Institute"* (1995), *Reengineering* adalah transformasi sistematis sistem yang ada menjadi bentuk baru untuk mewujudkan peningkatan kualitas dalam operasi, kemampuan sistem, fungsi, kinerja, atau *evolvability* dengan biaya lebih rendah, jadwal, atau risiko kepada pelanggan. Definisi ini menekankan bahwa fokus rekayasa ulang adalah pada peningkatan sistem yang ada dengan pengembalian yang lebih besar atas investasi (ROI) daripada yang dapat diperoleh melalui upaya pengembangan baru. Jika

rekayasa ulang tidak lebih murah, tidak dapat dicapai dalam kerangka waktu yang lebih singkat, tidak kurang berisiko, atau tidak menawarkan nilai yang lebih baik kepada pelanggan, maka upaya pengembangan baru harus dipertimbangkan

II.3 Penelitian Sebelumnya

Dalam laporan Tugas Akhir ini referensi tulisan ilmiah yang digunakan adalah tulisan ilmiah oleh Gannod(1995) dan Frakes(1995). Kedua tulisan ilmiah ini sama-sama bertujuan untuk membangun sistem baru dari sistem lama yang berorientasi prosedural menjadi berorientasi objek.

II.3.1 Studi kasus yang dilakukan oleh Gannod (1995)

Studi kasus yang dilakukan oleh Gannod (1995) ini bertujuan untuk menganalisa bagaimana kombinasi dari *objek oriented* dengan struktur analisis dan desain dapat digunakan untuk melakukan *reengineering* pada *software* Packrat yang digunakan oleh Texas Instruments untuk menganalisa trafik LAN (*Local Area Network*). Packrat adalah aplikasi untuk memonitor trafik jaringan yang diimplementasikan pada lingkungan Microsoft Windows 95. Pada studi kasus ini, metode yang dilakukan adalah *Reverse engineering* dan *Reengineering*. Implementasi pada sistem originalnya mencakup fasilitas untuk berkomunikasi dengan driver antarmuka jaringan via Windows 95 *Network Device Interface Spesification* (NDIS). Sistem Packrat melakukan perubahan pada antarmuka pengguna dan memiliki perangkat tambahan untuk membaca kode paket data.

Tulisan ilmiah ini mendeskripsikan studi kasus yang mencakup tiga fase investigasi, yaitu:

1. *Design Recovery* atau level abstraksi atau *reverse engineering*
2. Modifikasi Desain atau level perubahan
3. Implementasi Desain atau level perbaikan.

Tujuan utama dari abstraksi adalah agar pengembang dapat mengetahui sistem secara detail sehingga pengembang mampu memodifikasi desain dan sistem sesuai dengan permintaan. Langkah selanjutnya adalah mengubah desain dari sistem sebelumnya dengan menganalisa kebutuhan modifikasi sesuai

permintaan pelanggan. Target dari sistem ini adalah dengan sistem operasi yang sama yaitu Windows 95 dan bahasa pemrograman yang sama yaitu C.

Desain recovery dibagi menjadi 3 langkah, yaitu (1) High-level concept model, (2) *Construction of a low-level source code model* , (3) Penyempurnaan dari model yang dihasilkan sampai ke *medium-level model*.

Dua langkah pertama dilakukan bersamaan. Langkah ketiga selesai jika *high-level concept model dan low-level source code model* telah dibangun. *High-level concept model* dibangun menggunakan dua sumber informasi. Informasi pertama adalah hasil pengumpulan data dari analisis empiris pada sistem. Informasi kedua adalah data yang dikumpulkan dari pengembang perangkat lunak yang asli.

Langkah selanjutnya adalah Modifikasi Desain. Tujuannya adalah mengubah desain dari sistem yang ada dengan menganalisis kebutuhan modifikasi yang diminta oleh *customer* proyek. Kebutuhan ini memungkinkan para pengembang untuk mengidentifikasi konteks modifikasi yang diminta. Modifikasi desain ini dibagi menjadi 3 langkah, yaitu (1) Analisa kebutuhan dilakukan untuk membahas kebutuhan baru pada sistem, (2) Analisa dampak (*impact*) dilakukan untuk mengidentifikasi bagian-bagian sistem yang akan dimodifikasi berdasarkan kebutuhan yang baru, (3) *Design Recovery* dimodifikasi untuk menggabungkan kebutuhan-kebutuhan baru ke fungsi sistem yang ada. Pada setiap fase, analisis dan modeling dilakukan dalam konteks notasi OMT (*Object Modelling Technique*).

Langkah selanjutnya adalah Modifikasi Sistem. Modifikasi dari desain untuk menggabungkan kebutuhan-kebutuhan baru disebut dengan tahap alterasi atau tahap perubahan. Setelah desain diubah untuk menggabungkan kebutuhan-kebutuhan atau batasan baru, teknik-teknik pengembangan perangkat lunak tradisional dapat digunakan untuk membangun kode yang sesuai dengan desain yang telah dimodifikasi. Kendala yang tersembunyi pada fase pengembangan proyek adalah saat menggunakan kembali *source code* atau pengkodingan yang telah ada.

Langkah terakhir adalah implementasi perubahan pada sistem, termasuk pengujian, terjadi paling lama 4 minggu. Tantangan utama adalah saat memodifikasi kebutuhan yang terkait secara spesifik dengan bahasa pemrograman

dan lingkungan yang terbatas pada proyek. Pada akhirnya, sistem baru terdiri dari kurang lebih 11.000 baris kode, 2 kali lebih besar daripada sistem Packrat yang asli. Hal ini dikarenakan kompleksitas dan banyaknya kebutuhan baru yang diminta oleh *customer*. Proyek ini selesai tepat waktu dan menerima tanggapan positif dari customer di Texas Instruments.

Kesimpulannya adalah untuk melakukan *reengineering* perangkat lunak proses identifikasi lebih sulit dilakukan karena metode standard untuk melakukan *reengineering* tidak ada. Namun, ada beberapa penelitian mengenai proses *reengineering* perangkat lunak untuk dijadikan acuan. Beberapa pengetahuan yang diperoleh dari tulisan ilmiah ini adalah:

- Penerapan dan konsistensi terhadap suatu proses yang sistematis untuk melakukan *reengineering* menghasilkan banyak manfaat dan merupakan faktor utama dalam keberhasilan proyek.
- Teknik *reverse engineering* menggabungkan kegunaan metodologi desain OO dan desain SA / SD dapat mengubah dari sistem prosedural ke berorientasi objek.
- Penggunaan metode *reengineering* yang sistematis dapat menyelesaikan proyek sesuai jadwal.
- Kurangnya dokumentasi rinci pada sistem yang asli menyulitkan langkah *design recovery*, khususnya pada kendala waktu.
- Secara keseluruhan, kurangnya *software* yang mendukung proses *reengineering* menbuat proses lebih sulit, banyaknya waktu yang dihabiskan dan rawan kesalahan/*error*.
- Tim pengembangan menekankan bahwa perlunya suatu proses yang sistematis untuk memulihkan dan mendokumentasikan desain sebelum re-implementasi. Tim pengembangan menemukan bahwa proses pemulihan desain dan produk sampingan mengurangi waktu untuk melakukan re-implementasi dan pengujian.
- *Software reengineering* adalah metode yang sangat menarik untuk membuat penyempurnaan sistem yang ada. Namun, kurva pembelajaran untuk rekayasa ulang perangkat lunak terlalu tinggi untuk dilaksanakan dan dipelajari pada saat yang sama. Pengguna *software reengineering* proses harus mengetahui

tentang teknik dan alat sebelum menerapkannya ke sebuah sistem perangkat lunak.

II.3.2 Studi kasus yang dilakukan oleh Frakes

Studi kasus empiris ini membandingkan dua metode untuk melakukan *reeinginering* pada sistem berbasis prosedural menjadi sistem berorientasi objek. Metode pertama adalah manual dan digunakan sebagai dasar untuk mengevaluasi metode kedua yang berulang dan didasarkan pada analisis prosedur berpasangan. Metode perulangan sangat efektif untuk mengidentifikasi objek. Metode ini menghasilkan kode yang lebih sedikit, lebih efisien dan melewati test regresi lebih daripada metode manual.

Tujuan utama dari riset ini adalah untuk menemukan metode dan alat untuk membantu proses konversi data di bahasa pemrograman, seperti bahasa C ke bahasa pemrograman berorientasi objek seperti C++ atau Java. Ada kebutuhan untuk metodologi yang dapat menganalisis kode prosedural yang ada dan mengidentifikasi fungsi terkait dan data yang dapat dienkapsulasi ke dalam obyek dapat digunakan kembali dalam domain aplikasi.

Studi sebelumnya membandingkan hasil dari 2 studi sebelumnya, yang mereengineer *tool ccount metric* yang ditulis dengan bahasa C menjadi program berorientasi objek dengan bahasa C++. Pada studi pertama, *programmer* melakukan *reengineering* kode prosedural menggunakan metode manual. *Programmer* memeriksa kode C dan mendesain kode berorientasi objek berdasarkan prinsip yang dianggapnya tepat. Mereka menggunakan metode perulangan unutk menganalisa kode prosedural dan membantu *programmer* dalam menentukan bagaimana membuat kelas-kelas dari sekelompok fungsi. Metode ini menggunakan bermacam-macam pasangan *metrics* untuk keterkaitan antara prosedur dan karena itu ditempatkan di kelas yang sama pada desain berorientasi objek.

Pada studi ini, *programmer* menyajikan perbandingan empiris dari kode C++ yang dihasilkan oleh metode manual dengan kode C++ yang dihasilkan dari metode perulangan.

Programmer melakukan dua studi *reengineering* dari prosedural ke paradigma objek. Sistem prosedural menggunakan studi *ccount*, *tool metric* untuk

bahasa C yang menghitung sejumlah baris kode *commentary* (komentar) dan *non commentary* pada program C. Ccount di *reengineering* menjadi C++ menggunakan komponen standar yang dapat dipakai kembali dan pola desain tunggal untuk menangkap kelas utilitas. Ada beberapa cara untuk mengkonversi *software* produk dari satu bahasa ke bahasa lain. Hal ini untuk membuktikan bahwa hasil sesuai dengan bahasa yang diinginkan dengan tidak mengubah fungsionalitas terlalu banyak. Cara ini tidak efektif karena walaupun konversi telah selesai, produk yang baru tidak menggunkana semua kegunaan dan fitur dari bahasa pemrograman yang baru. Contohnya, konversi pada C++ menjadi berekstensi .cpp dan hanya mengubah 'printf' menjadi 'cout', tetapi produk yang dihasilkan tetap seperti C hanya saja dalam bahasa C++.

Pada studi kedua, programmer menggunakan metode Pole dengan mendefinisikan pasangan metric dan menggunakannya untuk mengidentifikasi objek yang berpotensi untuk digunakan ulang pada *tool metric ccount*. Studi ini membentuk dasar dari penelitian yang diusulkan.

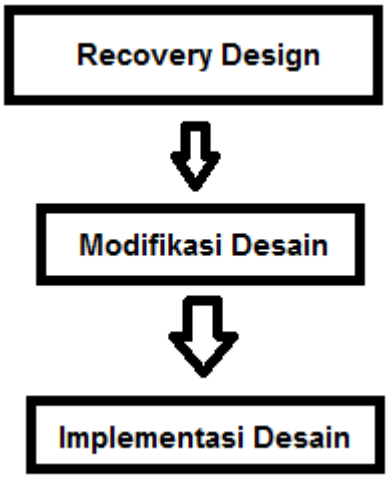
II.4 Studi Kasus

Pada tugas akhir ini, perangkat lunak yang kami gunakan adalah eProsedur. eProsedur merupakan sebuah perangkat lunak yang dibangun untuk membantu mengelola lalu lintas borang dalam Sistem Managemen Mutu (SMM) Politeknik Negeri Batam. Perangkat lunak ini dirancang agar proses pengajuan dan persetujuan borang menjadi lebih sederhana, lebih mudah dan lebih cepat sehingga dapat meningkatkan efektivitas kerja di Politeknik Negeri Batam. eProsedur diharapkan juga dapat meningkatkan komunikasi dalam organisasi Politeknik.

eProsedur dibangun menggunakan bahasa pemrograman PHP berorientasi prosedural. Pembangunan perangkat lunak ini tidak memiliki dokumentasi saat perancangan maupun implementasi.

Bab III Reverse engineering

Alur kerja yang digunakan adalah (1) tahap *recovery design* atau level abstraksi, (2) tahap modifikasi design atau level perubahan dan (3) tahap implementasi design atau level perbaikan.



Gambar 1 Alur Kerja

Pada tahap ini akan dilakukan *design recovery* atau disebut juga *reverse engineering* dan dihasilkan desain tingkat tinggi untuk aplikasi eProsedur yang terdiri dari:

1. Deksripsi fungsional sistem
2. Deskripsi data atau meta data
3. *Context diagram dan Data Flow Diagram* (DFD)
4. ER Diagram
5. Dekomposisi fungsional modul
6. Algoritma
7. Layar tampilan

eProsedur sendiri tidak memiliki dokumentasi pada saat pengembangannya. Sumber yang digunakan sebagai acuan pada tahap ini adalah manual eProsedur, *source code* eProsedur dan *database* eProsedure.

III.1 Deskripsi Fungsional Sistem

Berdasarkan manual eProsedur, dapat dihasilkan deskripsi fungsional sistem sebagai berikut:

1. Pengajuan borang untuk permintaan persetujuan
2. Pemrosesan borang yang diajukan
3. Notifikasi apabila terjadi pengajuan borang
   Notifikasi dilakukan dengan pengiriman email ke penerima borang
4. Notifikasi apabila borang yang diajukan telah diproses
   Notifikasi dilakukan dengan pengiriman email ke pengaju borang
5. Statistik lalu lintas borang
6. Pengelolaan prosedur dan borang dalam SMM

Pengguna aplikasi eProsedur dikelompokkan menjadi 3 (tiga) yaitu:

1. Admin
   Yaitu pengguna yang memiliki hak akses tertinggi. Dalam aplikasi ini, pengguna admin memiliki hak akses tambahan berupa pengaturan prosedur dan borang. Untuk saat ini, pengguna admin adalah UPT Penjaminan Mutu.
2. Ex-officio
   Yaitu pengguna berdasarkan jabatan strukturalnya (misal direktur, pembantu direktur, dan sebagainya). Dalam prosedur, biasanya pihak pengguna ex- officio ini adalah pihak yang berwenang untuk memberikan persetujuan terhadap sebuah pengajuan borang.
3. Personal
   Yaitu pengguna berdasarkan nama user pribadi.

III.2 Deskripsi Data

Dengan melihat database dari aplikasi eProsedur, maka dihasilkan deksripsi data sebagai berikut:

1. Nama Data : data user
   Deskripsi : berisi informasi mengenai user
   Terdiri dari :
   - username : nama yang digunakan untuk *login* pengguna

- password : kata sandi setiap pengguna untuk *login*
- nama lengkap : nama lengkap pengguna
- email : alamat email pengguna
- struktural : jabatan pengguna yang bernilai ‘1’ untuk user ex-officio dan ‘0’ untuk user pribadi

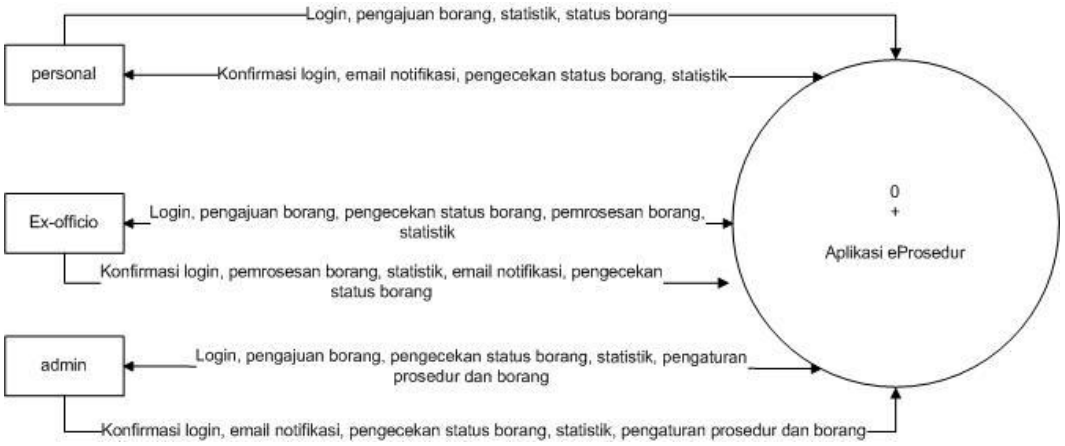
2. Nama Data : data prosedur
   Deskripsi : berisi informasi mengenai prosedur
   terdiri dari :
   - no prosedur : nomor pada setiap prosedur
   - nama prosedur : nama setiap prosedur

3. Nama Data : data borang
   Deskripsi : berisi informasi mengenai borang
   terdiri dari :
   - no borang : nomor pada setiap borang
   - nama borang : nama setiap borang

4. Nama Data : data pengajuan borang
   Deskripsi : berisi informasi mengenai pengajuan borang
   terdiri dari :
   - no borang : nomor borang yang diajukan
   - tgl pengajuan : tanggal borang diajukan
   - diajukan oleh : nama pengguna yang mengajukan borang
   - diajukan ke : pejabat struktural yang dimintai pengajuan borang
   - nama file : nama file yang diupload
   - path file : direktori tempat penyimpanan file
   - tgl persetujuan : tanggal borang disetujui oleh pejabat struktural
   - status : status pengajuan borang disetujui atau ditolak

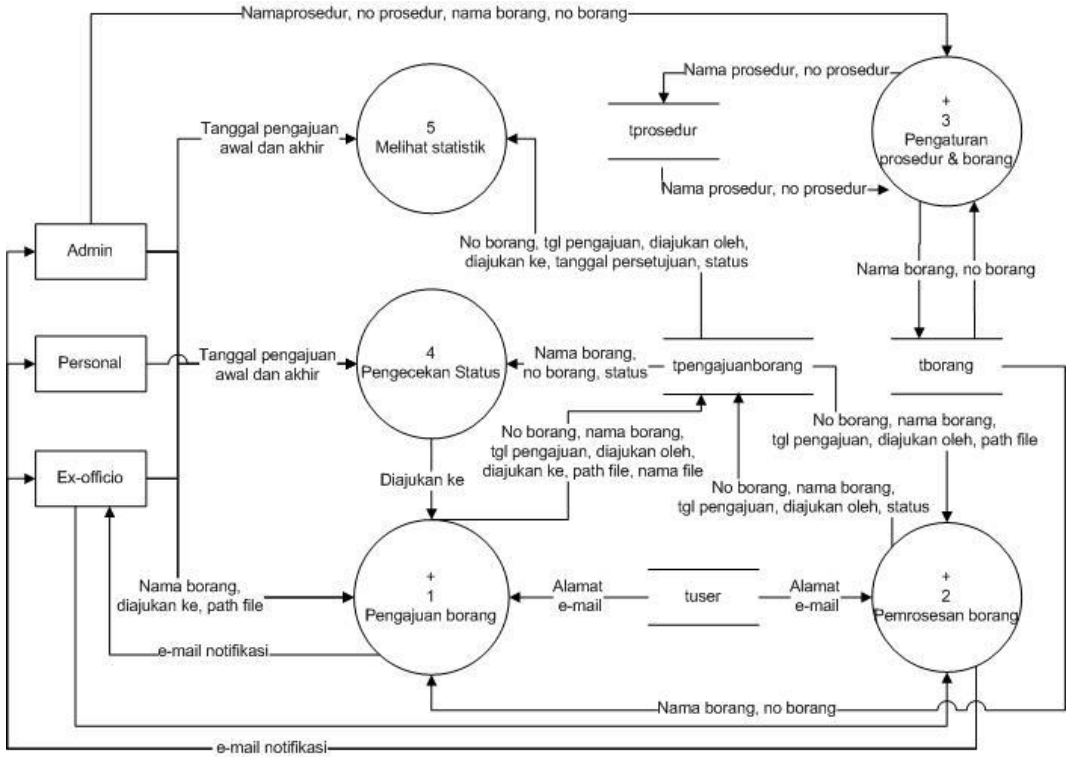
III.3 Context Diagram dan DFD

Berdasarkan deteksi proses awal dan pengelompokan proses, dapat dihasilkan *context diagram* dan DFD sebagai berikut:



Gambar 2 Context Diagram eProsedur

III.3.1 DFD Level 1

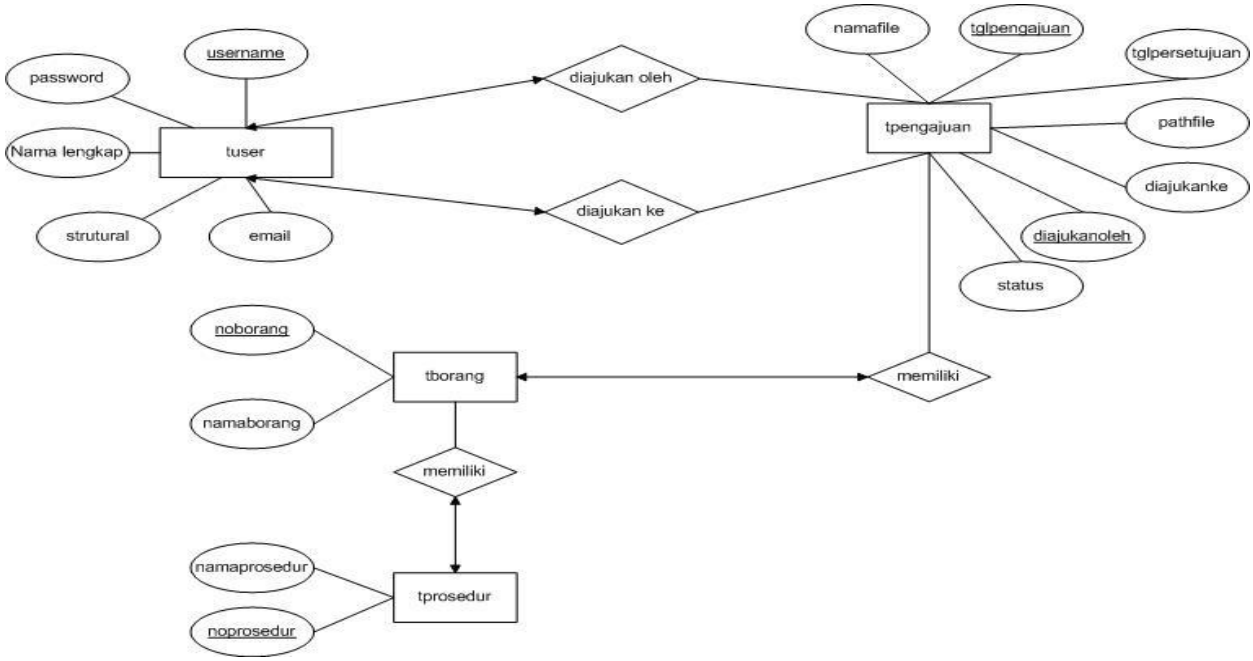


Gambar 3 DFD Level 1

Untuk deteksi proses awal dan pengelompokkan proses serta DFD level 2 dan 3, terdapat pada lampiran A.

III.4 ER Diagram

Dari deskripsi data, maka dapt ditentukan hubungan antar data yang digambarkan dengan ER diagram berikut ini:



Gambar 4 ER Diagram

Diagram ER tersebut juga dicocokkan kembali dengan basis data yang digunakan oleh aplikasi eProsedur.

III.5 Dekomposisi Fungsional Modul

Selanjutnya dilakukan penamaan fungsi dengan hubungan data yang ditampilkan dalam tabel dekomposisi fungsional modul. Secara lengkap tabel tersebut terdapat pada lampiran B.

III.6 Algoritma

Berdasarkan *source code* aplikasi eProsedur dapat dihasilkan algoritma. Secara lengkap, algoritma aplikasi eProsedur ditampilkan pada lampiran C.

III.7 Layar pada Aplikasi

Berdasarkan manual eProsedur dan *source code* dapat langsung diambil layar atau tampilan aplikasi eProsedur. Layar aplikasi ini secara lengkapnya terdapat pada lampiran D.

III.8 Tabel Keterunutan

Berdasarkan berbagai hasil *reverse engineering* di atas, maka disusun tabel keterunutan yang menghubungkan antara fungsi, layar dan data sebagai berikut:

Fungsi	Layar	Algoritma	Data
F-1.1 F-1.2	L0002	A0002 A0003	Data borang, data user, data pengajuan borang
F-1.3	-	A0004	-
F-4.0	L0003	A0005	Data borang, data pengajuan borang
F-1.1 F-1.2	L0004	A0006	Data user, data pengajuan borang
F-2.1	L0005	A0007	Data pengajuan borang
F-2.2	-	A0008	-
F-2.2	-	A0008	-
F-5.0	L0006	A0009	Data pengajuan borang
F-3.0	L0007	A0010	Data prosedur
F-3.1	L0008	A0011	Data prosedur
F-3.2	L0009	A0012	Data prosedur
F-3.3	L0010	A0013	Data prosedur
F-3.0	L0011	A0014	Data borang
F-3.4	L0012	A0015	Data borang
F-3.5	L0013	A0016	Data borang
F-3.6	L0014	A0017	Data borang

Tabel 1 tabel keterunutan

Bab IV Reeinginering

Pada bab ini akan dilakukan *reengineering* atau modifikasi desain. Proses *reengineering* ini menggunakan metode Iconix yang ditemukan oleh Doug Rosenberg dan Matt Stephens dan dijelaskan dalam bukunya yang berjudul *Use Case Driven Object Modelling with UML.*

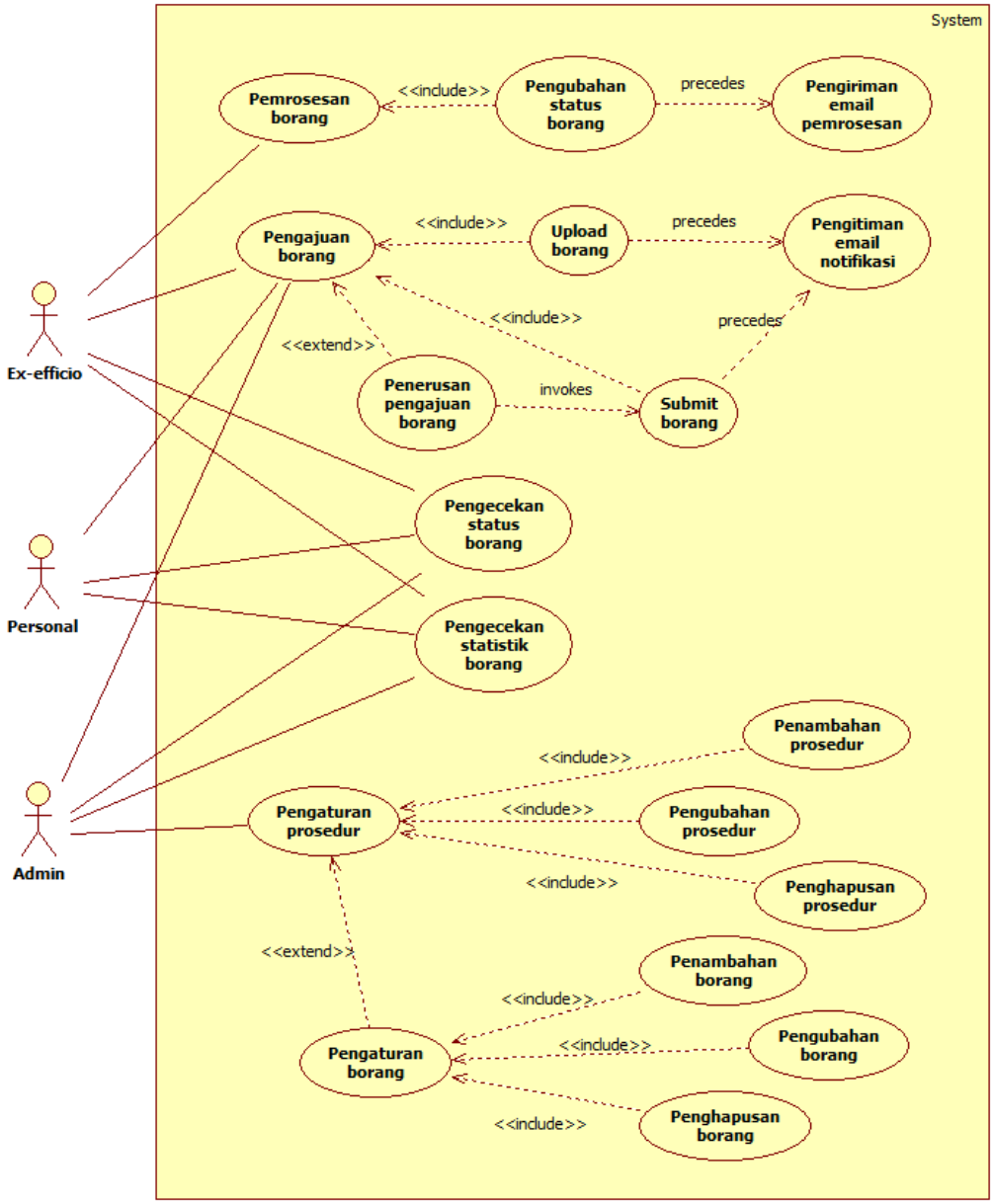
Pada proses *reeingineering* ini tidak ada penambahan fungsionalitas pada aplikasi eProsedur, sehingga fungsionalitas pada sistem yang lama masih dapat ditemukan pada sistem yang baru.

Metode tersebut meliputi langkah-langkah sebagai berikut:

1. *Requirement* : menentukan fungsionalitas dan membangun *use case*
2. Analisis atau *preliminary design* : membangun *robustness diagram*
3. *Detailed Design* : membangun *sequence diagram*
4. Implementasi : melakukan pengkodingan

IV.1 Usecase

Use case dibangun berdasarkan fungsionalitas sistem yang didapatkan dari fungsional sistem hasil *reverse engineering*, kerena tidak ada penambahan fungsional pada *reengineering*.



IV.2 Robustness Diagram

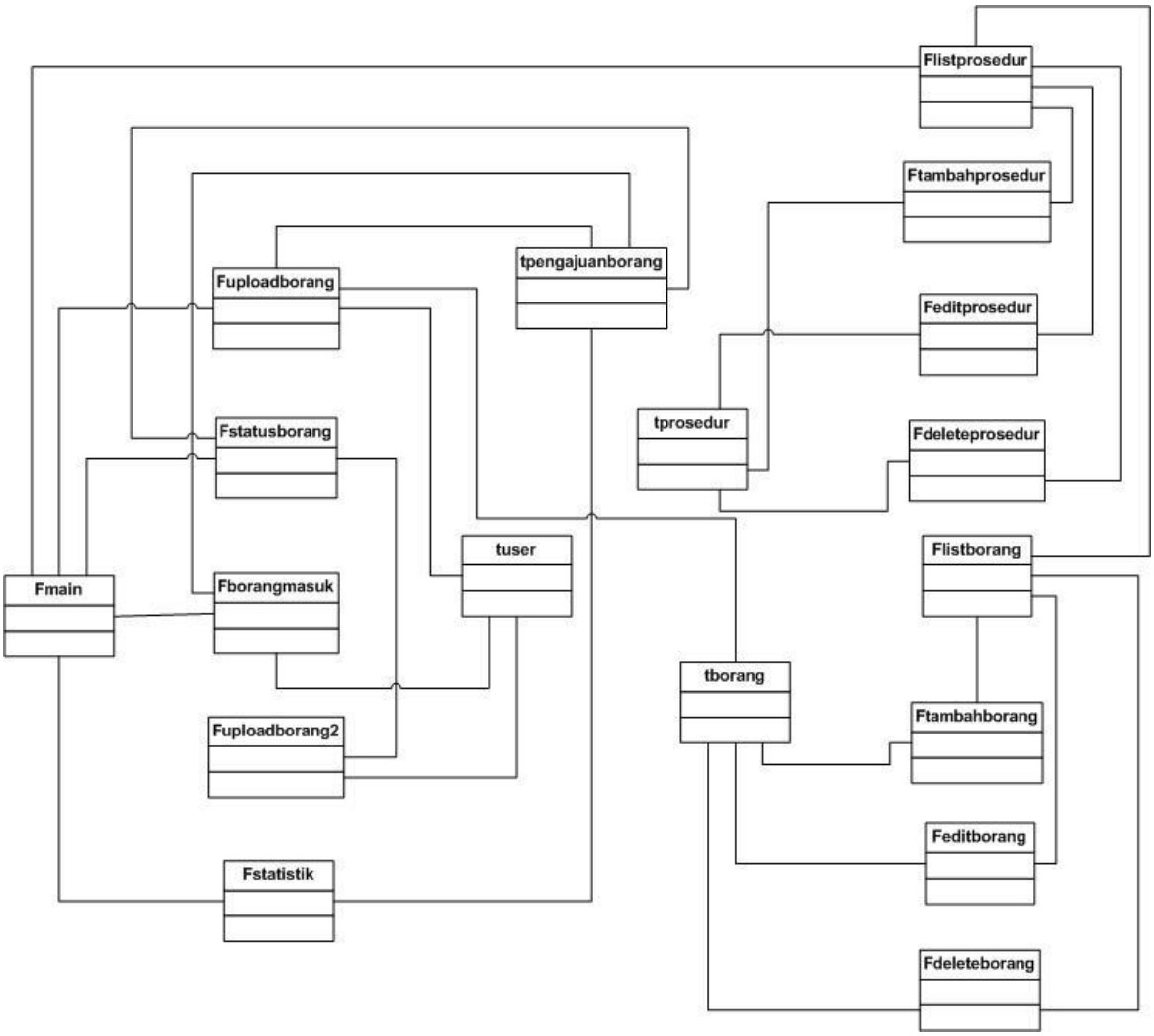
Robustness diagram dibangun berdasarkan use case diagram. Secara lengkapnya terdapat pada lampiran E.

IV.3 Sequence Diagram

Sequence diagram dibangun berdasarkan robustness diagram. Secara lengkapnya sequence diagram terdapat pada lampiran F.

IV.4 Diagram Kelas

Diagram kelas dibangun berdasarkan proses pada *sequence diagram*.



Setiap kelas mempunyai *method* masing-masing yang secara lengkapnya terdapat pada lampiran G.

IV.5 Algoritma

Algoritma pada sistem baru eProsedur menggunakan algoritma sistem yang lama karena tidak ada perubahan fungsionalitas.

IV.6 Tabel Keterunutan

Tabel keterunutan ini merupakan hubungan antar kelas yang didefinisikan dalam proses *reengineering* ini dengan hasil yang ditemukan di *reverse engineering*. Yang didefinisikan pada *reengineering* ini adalah kelas dan *method*, sedangkan yang digunakan dari sistem lama adalah algoritma dan layar.

No	Kelas	Method	Algoritma	Layar
1	fuploadborang	display()		L0002
		cekukuran()	A0003	
		uploadfile()	A0003	
		kirimemail()	A0004	
2	fstatusborang	display()		L0004
3	fborangmasuk	display()		L0005
		kirimemailstatus()	A0008	
4	fuploadborang2	display()		L0004
		kirimemail()	A0004	
5	fstatistik	display()		L0006
6	flistprosedur	display()		L0007
7	ftambahprosedur	display()		L0008
8	feditprosedur	display()		L0009
9	fdeleteprosedur	display()		L0010
10	flistborang	display()		L0011
11	ftambahborang	display()		L0012
12	feditborang	display()		L0013
13	fdeleteborang	display()		L0014

14	tpengajuanborang	getstatus()	A0005	
		getborangmasuk()	A0007	
		getstatistik()	A0009	
		submit()	A0002	
		submit2()	A0006	
		prosesborang()	A0007	
15	tuser	emaildiajukanke()	A0004	
		emaildiajukanoleh()	A0008	
		getpemohon()	A0007	
		gettermohon()	A0005	
		getnama()	A0001	
16	tborang	tambahborang()	A0015	
		editborang()	A0016	
		deleteborang()	A0017	
		getnonama()	A0001	
		getnamaborang()	A0005, A0007	
		getnamaborang2()	A0006	
		getlistborang()	A0014	
17	tprosedur	tambahprosedur()	A0011	
		editprosedur()	A0012	
		deleteprosedur()	A0013	
		getlistprosedur()	A0010	

Tabel 2 tabel keterunutan *reengineering*

Bab V Hasil dan Pembahasan

V.1 Implementasi

Dalam melakukan proses *reverse engineering*, acuan yang digunakan adalah manual aplikasi, *source code* dan *database*. Dari ketiga acuan terebut dapat dihasilkan Deskripsi Fungsional Sistem, Meta Data, DFD, ER Diagram, Dekomposisi Fungsional Modul, algoritma dan layar. Seluruh hasil dari *reverse engineering* ini berikutnya merupakan acuan dalam melakukan proses *reengineering*. Pada pengimplementasian *reengineering*, seluruh hasil desain digunakan kembali termasuk *source code* pada aplikasi proseduralnya.

Di awal pengerjaan, diperkirakan aplikasi berbasis OOP ini akan menggunakan seluruh *source code* aplikasi berbasis prosedural. Ketika *source code* pada aplikasi prosedural diubah tempatnya dan diletakkan menjadi fungsi tertentu, ada beberapa fungsi yang tidak adapat berjalan dengan lancar. kesulitan ini terutama dikarenakan oleh penyatuan HTML dan PHP. Pada beberapa fungsi, HTML dan PHP tidak dapat dipisahkan karena fungsinya tidak dapat berjalan dengan baik atau bahkan tidak dapat berjalan sama sekali. Implementasi dilakukan sesuai dengan rumusan masalah yang dikemukakan, bahwa tidak adanya penambahan fungsional dari aplikasi yang lama merupakan fokus utama dalam melakukan *reeingeering* ini. Oleh karena itu, *source code* yang ada benar-benar dipakai dalam pengimplementasian aplikasi ini, hanya tempatnya saja yang dipindahkan. Akan tetapi, pengerjaan untuk memotong, menyalin dan menempel bagian-bagian *source code* lama ke aplikasi yang baru tidak semua berjalan lancar sesuai perkiraan. Hal ini disebabkan karena perbedaan struktur pengkodean pada prosedural dan OOP. Ada beberapa fungsi yang dapat dipotong dan ditempl begitu saja dan ada juga fungsi yang perlu rumit untuk dijalankan. Beberapa contoh fungsi terlihat pada lampiran H.

Karena penggunaan *source code* pada aplikasi berbasis prosedural menjadi berbasis OOP tidak dapat dilakukan dengan hanya menempelkan *source code* yang ada begitu saja, atau hanya dengan merubah tempat peletakan *source code* menjadi fungsi-fungsi tertentu untuk dipanggil ketika dibutuhkan, tetapi struktur *coding* itu sendiri berbeda, maka untuk melakukan *reengineering* dari hasil *reverse engineering* yang dapat digunakan dari desain hanya sampai algoritma saja.

Salah satu fungsi utama, yaitu mengupload borang, pengkodean dapat dilakukan dengan memotong dan memindahkan letak *source code* saja dan fungsi dapat berjalan. Tetapi ada fungsi lain yang belum dapat berjalan seperti pengecekan statistik. Kendalanya berupa *source code* yang tidak dapat dipotong, dipindahkan dan ditempel begitu saja karena struktur pengkodean pada aplikasi proseduralnya berbeda dengan pengkodean pada aplikasi berorientasi objeknya, dan untuk itu solusinya masih dicoba dengan tetap menggunakan *source code* yang ada.

V.2 Pengujian

Fungsi yang diuji adalah fungsi utama pada aplikasi ini, yaitu upload borang dan pemrosesan borang. Skenario pengujian fungsi-fungsi ini terdapat di lampiran I.

V.2.1 Skenario Pengujian

Skenario pengujian fungsi-fungsi terdapat dilampiran I. Hasil pengujian terdapat pada Tabel 3.

No	Fungsi di awal aplikasi	Nama Usecase	Aktor	Kondisi Awal	Aksi	Target yang ingin dicapai	Hasil
1	uploadfile()	Pengajuan Borang	• admin • ex-officio • personal	Berada di form pengajuan borang	• mengisi nama • mengisi tujuan borang • mengupload file	• muncul pesan "file berhasil diupload" • Kembali ke form fuploadborang	Berjalan
2	getstatus()	Pengecekan Status Borang		Berada di form status pengajuan borang	Memasukkan tanggal pengajuan awal dan akhir	Menampilkan status pengajuan borang telah diproses atau belum	Berjalan
3	getborangmasuk()	Pengecekan Status Borang		Berada di form borang diajukan	Klik tombol cek borang masuk	Menampilkan data borang yang diajukan dan belum diproses	Berjalan
4	getstatistik()	Pengecekan Statistik		Berada di form statistic	Memasukkan tanggal pengajuan awal dan akhir	Menampilkan statistic pengajuan borang	Berjalan

						sesuai tanggal pengajuan	
5	submit()	Pengajuan Borang		Field pengajuan borang telah diisi	Menekan tombol submit	Menyimpan data pengajuan borang di tpengajuanborang	Berjalan
6	submit2()	Penerusan Pengajuan Borang		Field pengajuan untuk dituju ke telah diisi	Menekan tombol submit	Menyimpan data pengajuan borang yang diteruskan di tpengajuanborang	Berjalan
7	prosesborang()	Pemrosesan Borang		Berada di form borang diajukan dan menekan tombol cek borang masuk	Menekan tombol setuju	Mengubah status pengajuan borang menjadi disetujui	Berjalan
					Menekan tombol tidak setuju	Mengubah status pengajuan borang menjadi ditolak	Berjalan
8	getnama()	Pengajuan borang				Menampilkan	Berjalan

						nama lengkap dari user ex-officio	
9	tambahborang()		admin	Berada di form tambah borang	Mengisi field tambah borang	Borang berhasil ditambahkan ke database	Berjalan
10	editborang()			Berada di form edit borang	Mengubah field tambah borang	Data borang diubah dan tersimpan di database	Berjalan
11	deleteborang()			Berada di form delete borang	Menekan tombol delete borang	Borang pada prosedur tertentu berhasil dihapus	Berjalan
12	getnonama()			Berada di form upload borang	Memilih nama borang	Menampilkan no dan nama borang	Berjalan
13	getlistborang()			Berada di form list borang	-	Menampilkan daftar borang pada suatu prosedur	Berjalan

14	tambahprosedur()			Berada di form tambah prosedur	Mengisi field tambah prosedur	prosedur berhasil ditambahkan ke database	Berjalan
15	editprosedur()			Berada di form edit borang	Mengubah field edit prosedur	Data prosedur diubah dan tersimpan di database	Berjalan
16	deleteprosedur()			Berada di form delete prosedur	Menekan tombol delete prosedur	Menghapus data prosedur	Berjalan
17	getlistprosedur()			Berada di form list borang	-	Menampilkan daftar prosedur	Berjalan

Bab VI Kesimpulan dan Saran

VI.1 Kesimpulan

Berdasarkan pengujian yang telah dilakukan pada bab sebelumnya, maka kesimpulan yang dapat diambil adalah:

1. Dalam mengimplementasikan *reverse engineering* diperoleh deskripsi fungsional sistem dari manual, meta data dari database, DFD dari deskripsi fungsional sistem, ER Diagram dari meta data, dekomposisi fungsional modul, algoritma dari *source code* dan layar aplikasi. Setelah melakukan reverse engineering, maka hasilnya dapat dipakai pada sistem baru yang berorientasi objek.
2. Dalam proses *reengineering* perangkat lunak keseluruhan desain dari fungsionalitas sampai algoritma dapat dipakai kembali, tetapi tidak semua *source code* pada aplikasi lama dapat dipakai kembali pada aplikasi yang baru. Proses pengkodean tidak hanya menempelkan bagian-bagian tertentu dan mengubah letaknya, beberapa *source code* pada aplikasi prosedural cocok untuk digunakan kembali menjadi aplikasi berbasis OOP, tetapi ada beberapa fungsi yang masih belum dapat berjalan karena struktur pengkodean yang tidak cocok diterapkan dari *source code* prosedural menjadi OOP.

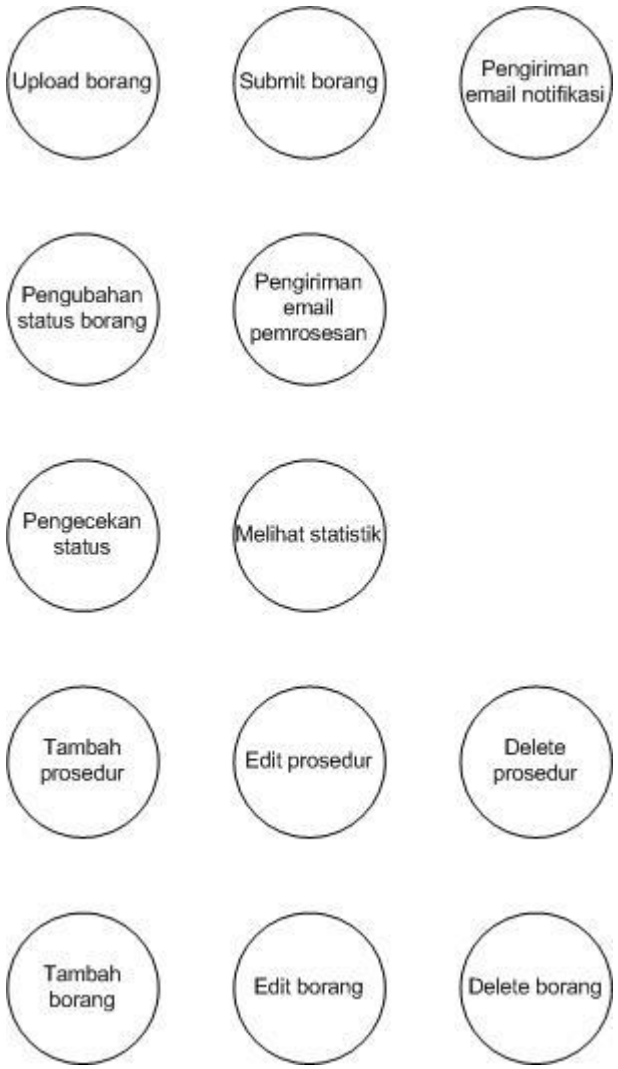
VI.2 Saran

Setelah dilakukan pengujian serta evaluasi, penulis menyarankan akan lebih baik jika aplikasi dibuat dari awal dengan hanya menggunakan desain sampai algoritma saja, karena tidak semua source code dapat digunakan kembali, jadi akan lebih sulit jika mengubah yang telah ada, sehingga akan lebih mudah jika memulai dari form baru dengan hanya menggunakan hasil dari *reverse engineering* saja.

Lampiran A Deteksi Proses Awal dan DFD Level 2 dan 3

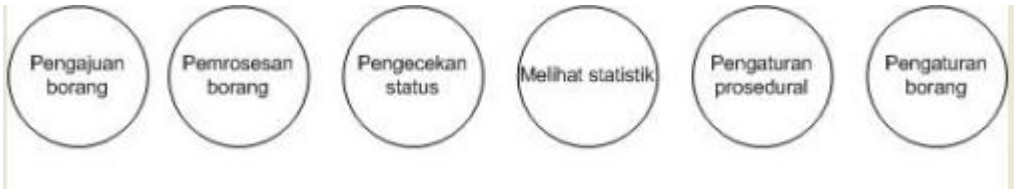
A.1 Deteksi Proses Awal

Berdasarkan dari manual dan aplikasi eProsedur maka menghasilkan proses sebagai berikut:



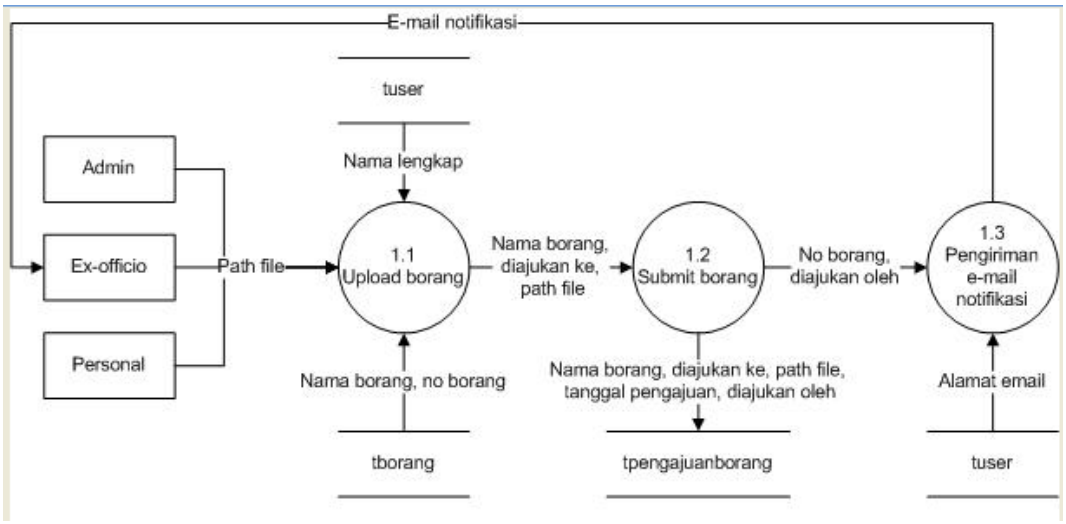
A.2 Pengelompokan Proses

Berdasarkan deteksi proses awal, proses-proses tersebut dapat dikelompokkan menjadi sebagai berikut:

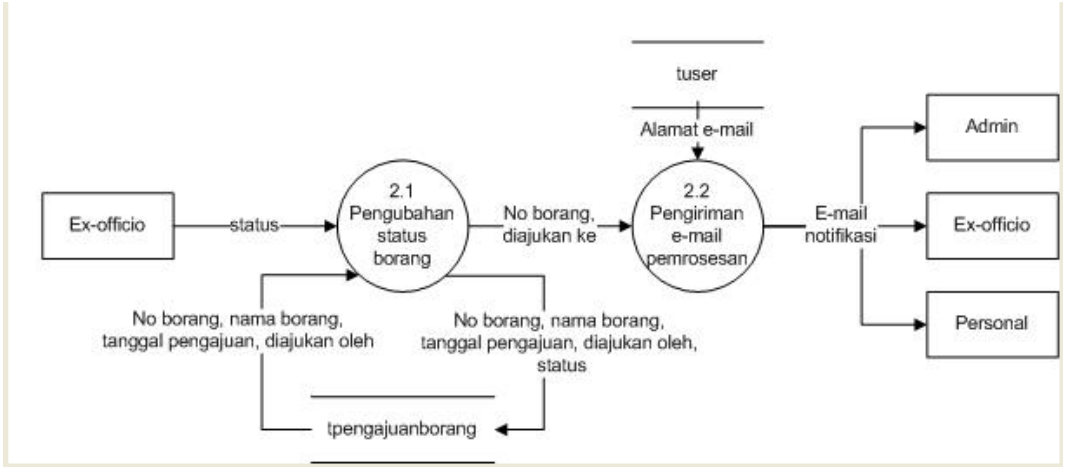


Berdasarkan pengelompokan proses, proses pengaturan prosedur dan pengaturan borang dapat dikelompokkan lagi menjadi sebagai berikut:

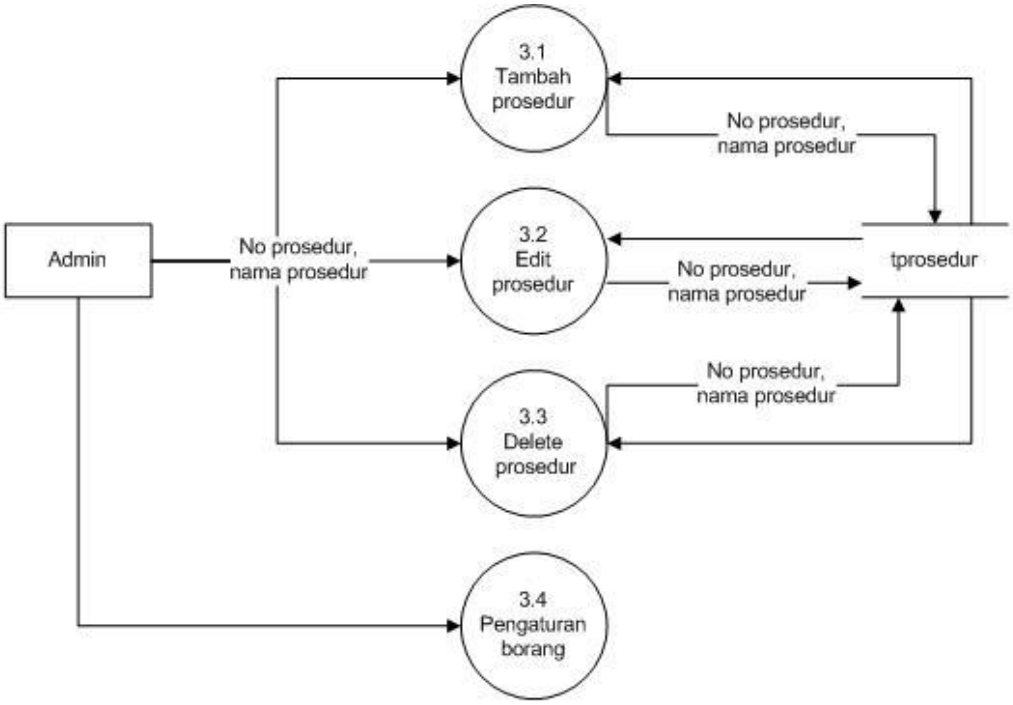
A.3 DFD Level 2 Proses 1 Pengajuan Borang



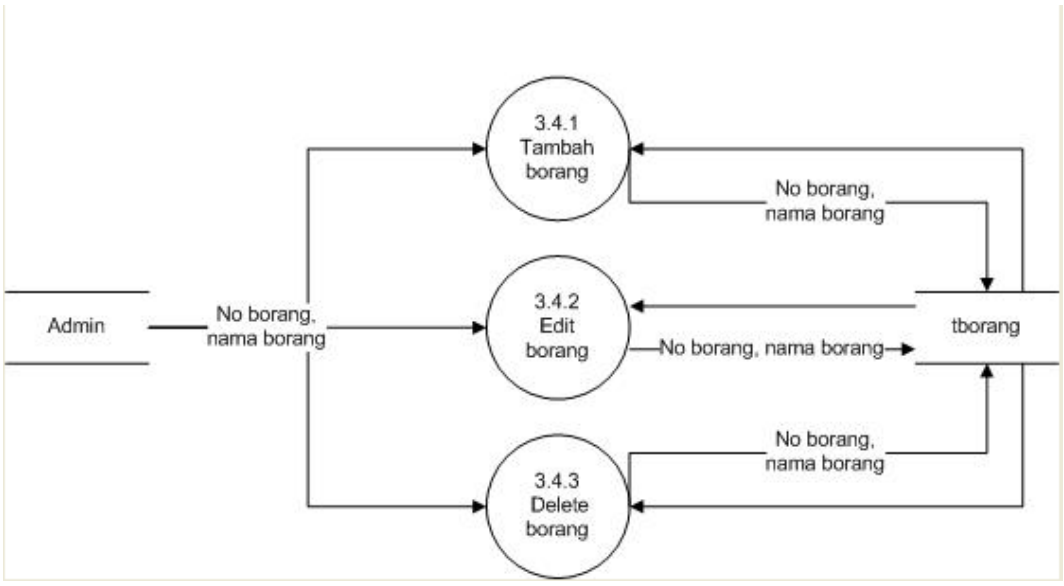
A.4 DFD Level 2 Proses 2 Pemrosesan Borang



A.5 DFD Level 2 Proses 3 Pengaturan Prosedur dan Borang



A.6 DFD Level 3 Proses 3.4 Pengaturan Borang



Lampiran B Dekomposisi Fungsional Modul

No	Fungsi/Proses	Data Input	Data Output	Keterangan
F-1.1	Upload borang	Nama borang, penerima borang, file	-	Fungsi ini adalah fungsi untuk mengupload borang yang ingin diajukan
F-1.2	Submit borang	-	Pesan pengiriman berhasil	Fungsi ini untuk memasukkan data yang diajukan ke database.
F-1.3	Kirim email notifikasi	-	Email	Fungsi ini untuk mengirimkan email secara otomatis ke pengguna yang mendapat pengajuan borang
F-4.0	Lihat status borang	Tanggal pengajuan awal, tanggal pengajuan akhir	Info status borang	Fungsi ini untuk melhat status pengajuan borang apakah disetujui atau ditolak
F-1.1 F-1.2	Penerusan pengajuan	Penerima borang	Pesan pengiriman berhasil	Fungsi ini untuk meneruskan pengajuan borang kepada pihak selanjutnya
F-2.1	Pemrosesan borang		Email notifikasi	Fungsi ini untuk memproses borang

			penyetujuan atau penolakan borang	yang diajukan apakah akan disetujui atau ditolak
F-2.2	kirim email notifikasi borang disetujui		Email	Fungsi ini untuk mengirimkan email kepada pengaju borang bahwa borangnya disetujui
F-2.2	Kirim email notifikasi borang ditolak		Email	Fungsi ini untuk mengirimkan email kepada pengaju borang bahwa borangnya ditolak
F-5.0	Lihat statistic	Tanggal pengajuan awal, tanggal pengajuan akhir	Info statistk borang	Fungsi ini untuk menampilkan statistic borang diantara dua tanggal yang diinputkan
F-3.0	Lihat daftar prosedur		Daftar prosedur	Fungsi ini untuk melihat daftar prosedur
F-3.1	Tambah prosedur	Nama prosedur, nomor prosedur	Pesan prosedur berhasil ditambahkan	Fungsi ini untuk menambahkan prosedur baru ke database
F-3.2	Edit prosedur	Nama prosedur	Pesan prosedur berhasil diubah	Fungsi ini untuk mengubah nama prosedur yang telah ada
F-3.3	Delete prosedur	Nama prosedur, no prosedur	Pesan prosedur berhasil dihapus	Fungsi ini untuk menghapus prosedur yang ada di database

F-3.0	Lihat daftar borang		Daftar borang	Fungsi ini untuk melihat daftar borang yang ada diprosedur tertentu
F-3.4	Tambah borang	Nama borang, nomor borang	Pesan borang berhasil ditambahkan	Fungsi ini untuk menambahkan borang baru pada prosedur tertentu
F-3.5	Edit borang	Nama borang	Pesan borang berhasil diubah	Fungsi ini untuk mengubah borang yang ada di database
F-3.6	Delete borang	Nama borang	Pesan borang berhasil dihapus	Fungsi ini untuk menghapus borang yang ada di database

Lampiran C Algoritma

C.1 Algoritma Proses pengajuan borang

Nama Algoritma : A0001
Deskripsi : Algoritma untuk menampilkan nama borang dan penerima borang pada form pengajuan borang

Initial state : data telah ada di database
Final State : data ditampilkan ke form untuk dipilih user

```
Algoritma:

eksekusi query "SELECT noborang,namaborang FROM tborang ORDER BY
noborang";
write (no borang, nama borang)
then
eksekusi query "SELECT * FROM tuser WHERE struktural = '1' ORDER
BY namalengkap";
write (structural)
```

Nama Algoritma : A0002
Deskripsi : Algoritma untuk mengajukan borang

Initial state : data belum ditambahkan
Final state : data ditambahkan ke database

```
Algoritma :
input (nama borang)
input (username tujuan)
input (noborang)
```

ambil no_borang dari database

Nama Algoritma : A0003
Deskripsi : mengupload borang yang diajukan

Initial state : file belum ada di direktori server
Final state : file teruplоad dan disimpan di direktori server

algoritma :

Read (nama borang)
Read (username tujuan)
Read (file)

if ["filename"]["size"] > 20000000, maka akan muncul pesan “size file terlalu besar ! silahkan coba lagi..”
else
"INSERT INTO tpengajuanborang (noborang, tglpengajuan, diajukanoleh, diajukanke, namafile, pathfile, status) VALUES ('$noborang', '$tanggal', '$pengirim', '$struktural', '$namaFile','$dir','0')";

Nama Algoritma : A0004
Deskripsi : Algoritma mengirim email notifikasi borang masuk

Initial State : file telah ada di direktori server
Final State : file dikirim ke penerima borang

Algoritma:

Read (username penerima)
Alamat email dicari berdasarkan username penerima pada table tpengajuan dan didapatkan sesuai dengan email pada table tuser.

Do query SELECT email FROM tuser WHERE username='$struktural' untuk mendapatkan alamat email penerima,
Then
Send (to $rowemail, from $username, message yang berisi “Anda mendapatkan email ini karena terdapat pengajuan borang ".$noborang." oleh ".$username.". Harap segera diproses melalui aplikasi eProsedur”.

C.2 Algoritma pengecekan status borang

Nama Algoritma : A0005
Deskripsi : Algoritma mengecek status borang yang diajukan

Initial State : data telah ada di database
Final State : menampilkan data borang yang diajukan beserta status

Algoritma :

Read (tanggal pengajuan awal)
Read (tanggal pengajuan akhir)

eksekusi query :
SELECT * FROM tpengajuanborang WHERE diajukanoleh='$username' AND tglpengajuan BETWEEN '$tglawal' AND '$tglakhir' ORDER BY tglpengajuan";

Then eksekusi query
SELECT namaborang FROM tborang WHERE noborang='$nobor', untuk memperoleh nama borang berdasarkan no borang.

Then eksekusi query
$query2 = "SELECT namalengkap FROM tuser WHERE username='$termohon', untuk memperoleh nama lengkap user ang mendapa pengajuan borang.

C.3 Algoritma meneruskan pengajuan borang

Nama Algoritma	: A0006
Deskripsi	: Algoritma meneruskan pengajuan borang ke setelah disetujui penerima pertama
Initial State	: data telah ada database
Final State	: data di database ditambahkan

Algoritma :

Eksekusi query SELECT namaborang FROM tborang WHERE noborang = '$noborang';
SELECT * FROM tuser WHERE struktural = '1' ORDER BY namalengkap;
Write (nama borang, file)
Read penerima borang
Eksekusi query:
INSERT INTO tpengajuanborang (noborang, tglpengajuan, diajukanoleh, diajukanke, namafile, pathfile, status) VALUES ('$noborang' ,'$tanggal', '$pengirim', '$struktural', '$namaFile', '$dir','0')";
Setelah itu akan muncul pesan “borang berhasil diupload”.

C.4 Algoritma pemrosesan borang

Nama Algoritma	: A0007
Deskripsi	: Algoritma melihat borang masuk dan menyetujui atau menolak pengajuan
Initial state	: borang telah ada di database
Final state	: status borang diperbaharui

Algoritma :

Eksekusi query "SELECT * FROM tpengajuanborang WHERE diajukanke='$username' AND status='0' ORDER BY tglpengajuan";

Then eksekusi query
"SELECT namaborang FROM tborang WHERE noborang='$nobor'"; untuk mengambil nama borang berdasarkan no borang.

Then eksekusi query
"SELECT * FROM tuser WHERE username='$pemohon'"; untuk mendapatkan nama lengkap pengaju borang.
Write (daftar borang masuk)

Then eksekusi query

if UPDATE tpengajuanborang SET status='1', tglpersetujuan='$tanggal2' WHERE noborang='$nobor' AND tglpengajuan='$tanggal' AND diajukanoleh='$pemohon', maka borang disetujui
Else
borang ditolak

Nama Algoritma : A0008
Deskripsi : mengirimkan email notifikasi penyetujuan atau penolakan borang

Initial State : data borang di database telah diperbaharui
Final State : email notifikasi dikirim ke pemohon

Algoritma :
Read (username pemohon)
Inisialisasi (alamat email pemohon)

Eksekusi query SELECT * FROM tuser WHERE username='$pemohon'";

untuk mendapatkan email pengirim.
Then
send($to, $headers, $message);
if UPDATE tpengajuanborang SET status='1', tglpersetujuan='$tanggal2' WHERE noborang='$nobor' AND tglpengajuan='$tanggal' AND diajukanoleh='$pemohon', maka pesan yang dikirimkan adalah “Borang Anda disetujui” dan “Anda mendapatkan email ini karena pengajuan ".$nobor." kepada ".$username." telah diproses dan disetujui. Silakan dicek melalui aplikasi eProsedur.”

ELSE

Pesan yang dikirim berisi “Borang Anda tidak disetujui” dan “Anda mendapatkan email ini karena pengajuan ".$nobor." kepada ".$username." telah diproses dan tidak disetujui. Silakan dicek melalui aplikasi eProsedur.”

C.5 Algoritma melihat statistik

Nama Algoritma	: A0009
Deskripsi	: Algoritma melihat statistic pengajuan borang diantara 2 tanggal pengajuan
Initial state	: data pengajuan telah ada di database
Final state	: data statistik pengajuan borang

Algoritma :

Read (tanggal pengajuan awal)
Read (tanggal pengajuan akhir)

Switch {
Case 1 : SELECT COUNT(*) FROM tpengajuanborang WHERE tglpengajuan BETWEEN '$tglawal' AND '$tglakhir'"; Lalu akan ditampilkan jumlah borang masuk.

Case 2 : "SELECT * FROM tpengajuanborang WHERE tglpengajuan BETWEEN '$tglawal' AND '$tglakhir' AND (tglpersetujuan-tglpengajuan) IN (SELECT MIN(tglpersetujuan-tglpengajuan) FROM tpengajuanborang WHERE tglpengajuan BETWEEN '$tglawal' AND '$tglakhir')"; akan ditampilkan data borang yang paling cepat diproses.

Case 3 : SELECT * FROM tpengajuanborang WHERE tglpengajuan BETWEEN '$tglawal' AND '$tglakhir' AND (tglpersetujuan-tglpengajuan) IN (SELECT MAX(tglpersetujuan-tglpengajuan) FROM tpengajuanborang WHERE tglpengajuan BETWEEN '$tglawal' AND '$tglakhir')"; menampilkan data borang yang paling lama diproses.

Case 4 : SELECT diajukanoleh FROM tpengajuanborang WHERE tglpengajuan BETWEEN '$tglawal' AND '$tglakhir' GROUP BY diajukanoleh HAVING count(*) >= ALL (SELECT count(*) FROM tpengajuanborang WHERE tglpengajuan BETWEEN '$tglawal' AND '$tglakhir' GROUP BY diajukanoleh)"; menampilkan data user yang paling banyak mengajukan borang.

Case 5 : SELECT diajukanke FROM tpengajuanborang WHERE tglpengajuan BETWEEN '$tglawal' AND '$tglakhir' GROUP BY diajukanke HAVING count(*) <= ALL (SELECT count(*) FROM tpengajuanborang WHERE tglpengajuan BETWEEN '$tglawal' AND '$tglakhir' GROUP BY diajukanke)"; menampilkan data user yang paling banyak mendapatkan pengajuan borang.

C.6 Algoritma pengaturan prosedur dan borang

B.6.1 Algoritma pengaturan prosedur

Nama Algoritma : A0010
Deskripsi : Algoritma menampilkan daftar prosedur

Initial state : data telah ada di database
Final state : data prosedur ditampilkan

Algoritma :

Read (username)
username harus merupakan admin.
eksekusi query SELECT * FROM tprosedur"; untuk menampilkan daftar prosedur.

Nama Algoritma : A0011
Deskripsi : Algoritma menambahkan data prosedur ke database

Initial state : data belum ada di database
Final state : menambahkan data prosedur ke database

Algoritma :

Read (nomor prosedur)
Read (nama prosedur)

eksekusi query : INSERT INTO tprosedur VALUES ('$nopros','$nama')";
nopros dan nama sesuai dengan no prosedur yang diinputkan user

Nama Algoritma : A0012
Deskripsi : Algoritma mengubah data prosedur baik nomor maupun nama

Initial state : data telah ada di database
Final state : data diperbaharui

Algoritma :

Inisialisasi (noprosedur)

Read (nomor prosedur)
Read (nama prosedur)

eksekusi query:
"UPDATE tprosedur set noprosedur= '$noprosubah', namaprosedur = '$namaprosubah' WHERE noprosedur= '$nodiubah'"; sesuai dengan nompr prosedur dan nama prosedur yang diinputkan user.

Nama Algoritma : A0013
Deskrisi : Algoritma menghapus prosedur pada database

Initial state : data telah ada di database
Final state : data dihapus

Algoritma :

inisialisasi (noprosedur)
Read (nomor prosedur)
Read (nama prosedur)
nomor prosedur dan nama prosedur diambil dari noprosedur di database untuk dihapus.
eksekusi query:
"DELETE FROM tprosedur WHERE noprosedur='$nodihapus'";

C.6.2 Algoritma pengaturan borang

Nama Algoritma : A0014
Deskripsi : Algoritma melihat daftar borang pada setiap prosedur

Initial state : menampilkan daftar prosedur
Final state : menampilkan daftar borang

algoritma :

Read (no prosedur)
Read (nama prosedur)
eksekusi query:
"SELECT * FROM tborang WHERE noprosedur = '$nopros'";
Write Daftar borang pada perosedur yang dipilih ditampilkan.

Nama Algoritma : A0015
Deskripsi : Algoritma menambahkan borang ke database

Initial state : data borang belum ada di database
Final state : data borang ditambahkan ke database

algoritma :

Write (noprosedur)
Read (nomor borang)
Read (nama borang)

eksekusi query:
"INSERT INTO tborang VALUES ('$noborang','$nama','$nopros')";

Nama Algortima : A0016
Deskripsi : Algoritma mengubah data borang pada database

Initial state : data borang telah ada di database
Final state : data borang di database diperbaharui

algoritma :

Write (noprosedur)
Read (nomor borang)
Read (nama borang)
inisialisasi (noborang)
nomor borang dan nama borang diambil dari no borang di database untuk diubah
eksekusi query:
"UPDATE tborang SET noborang= '$noborangubah', namaborang= '$namaborangubah' WHERE noborang= '$nodiubah'";

Nama Algoritma : A0017
Deskripsi : Algoritma menghapus borang di database

Initial state : data borang telah ada di database
Final state : data borang di database dihapus

algoritma :

Write (noprosedur)
Read (nomor borang)
Read (nama borang)
inisialisasi (noborang)

nomor borang dan nama borang diambil dari no borang di database untuk dihapus.
eksekusi query:
"DELETE FROM tborang WHERE noborang='$nodihapus'";

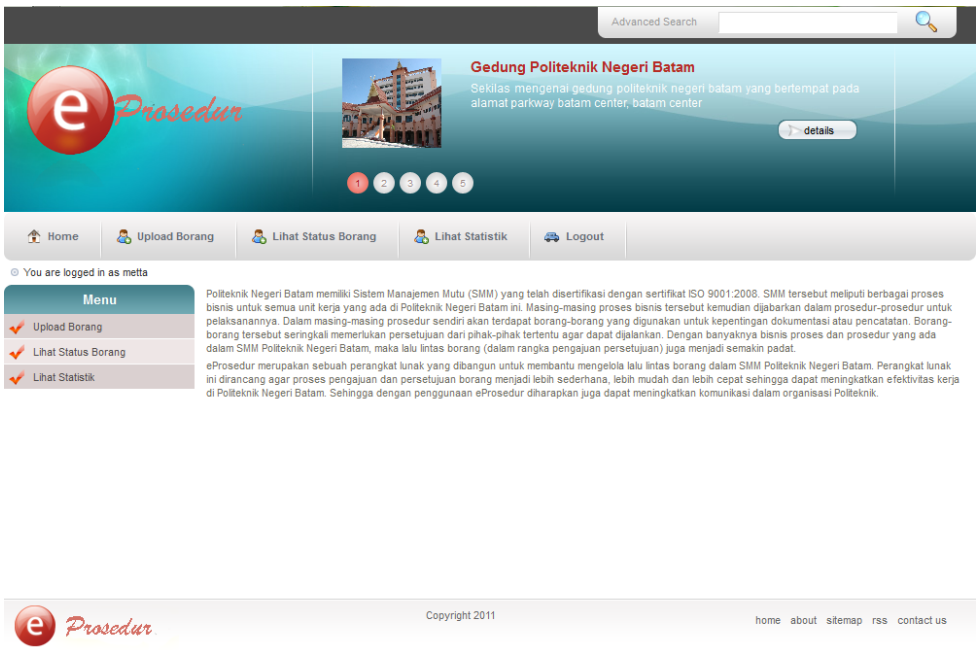
Lampiran D Layar Tampilan

D.1 Layar Utama

Nama Layar : L0001

Deskripsi : Layar Utama yang berisi informasi mengenai Sistem Managemen Mutu (SMM) politeknik negeri batam.

Terdiri dari : Menu-menu sesuai dengan hak akses pengguna



Gambar 5 Layar Utama Pengguna Personal

D.2 Layar Proses pengajuan borang

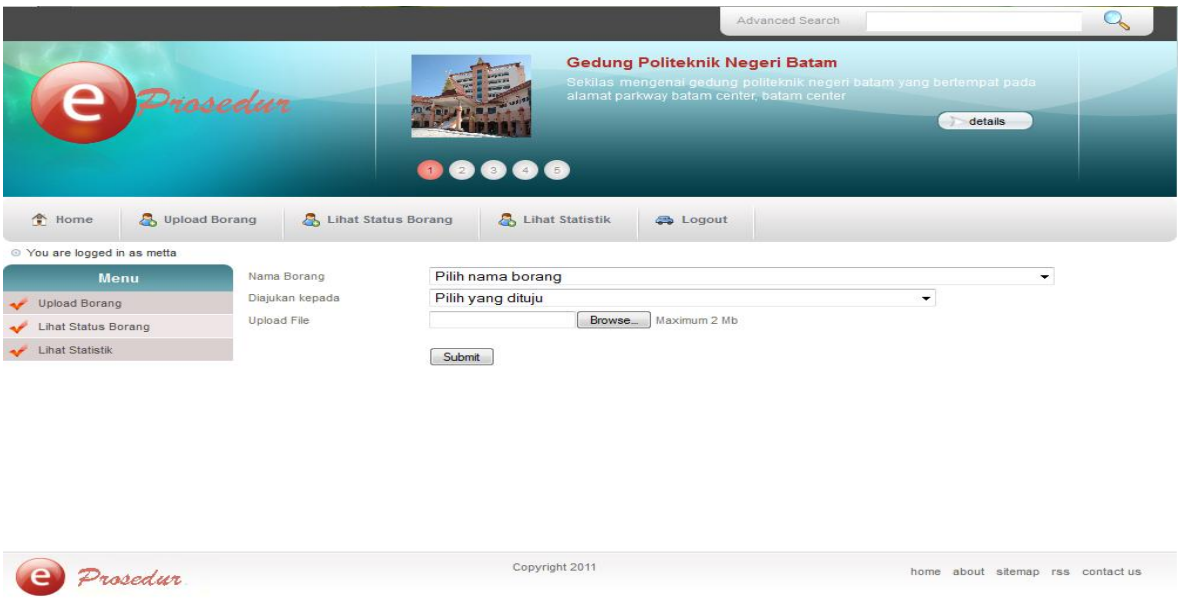
Nama Layar : L0002

Deskripsi : Layar untuk pengajuan borang

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- combobox Nama Borang untuk mengisi nama borang yang ingin diajukan

- combobox Diajukan Kepada untuk mengisi penerima borang. Penerima borang disini adalah pengguna structural.
- textfield Upload File untuk memasukkan file yang ingin di upload untuk dikirim.
- Tombol Submit untuk mengajukan borang yang sudah diisi datanya dan sekaligus mengupload file



Gambar 6 layar upload borang

D.3 Layar Pengecekan Status Borang

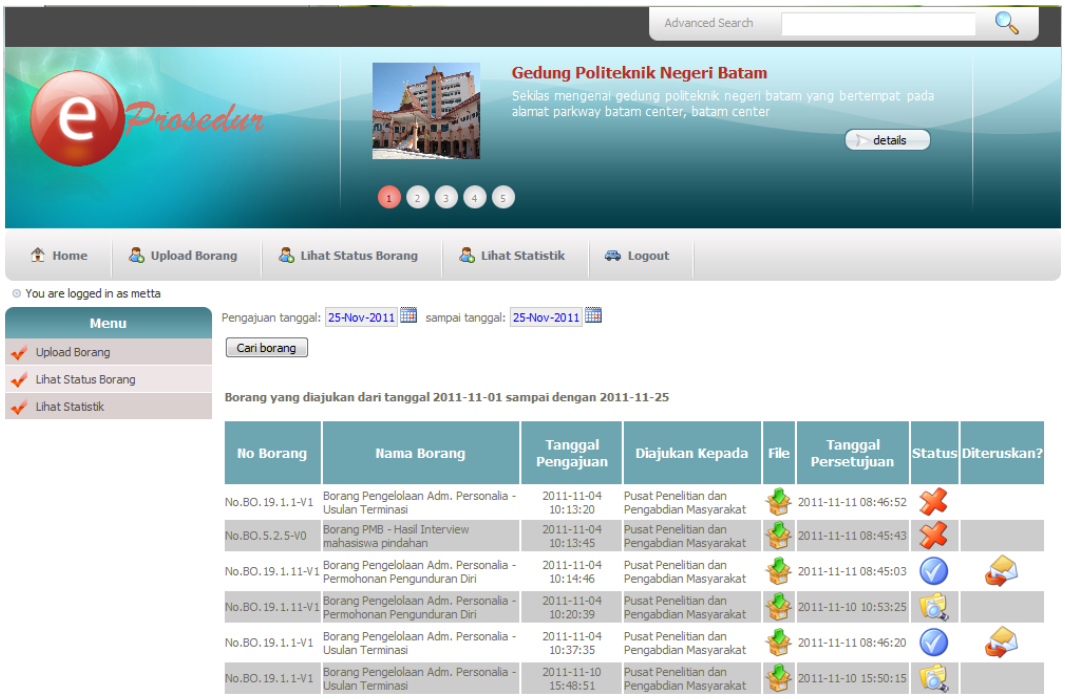
Nama Borang : L0003

Deskripsi : Layar untuk pengecekan status borang

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- textxfield calendar untuk mengisi tanggal awal dan akhir pengajuan borang yang ingin di cek
- tombol cari borang untuk mencari borang sesuai dengan tanggal yang diisi
- tabel data borang menampilkan daftar borang yang pernah diajukan sesuai dengan tanggal awal dan akhir yang diinputkan beserta status pengajuan apakah telah disetujui atau ditolak maupun belum diproses

- icon penerusan pengajuan untuk meneruskan pengajuan borang kepada pihak kedua.



Gambar 7 Layar Lihat Status Borang setelah tanggal dipilih

D.4 Layar Penerusan pengajuan borang

Gambar 8 Contoh borang dengan status Disetujui

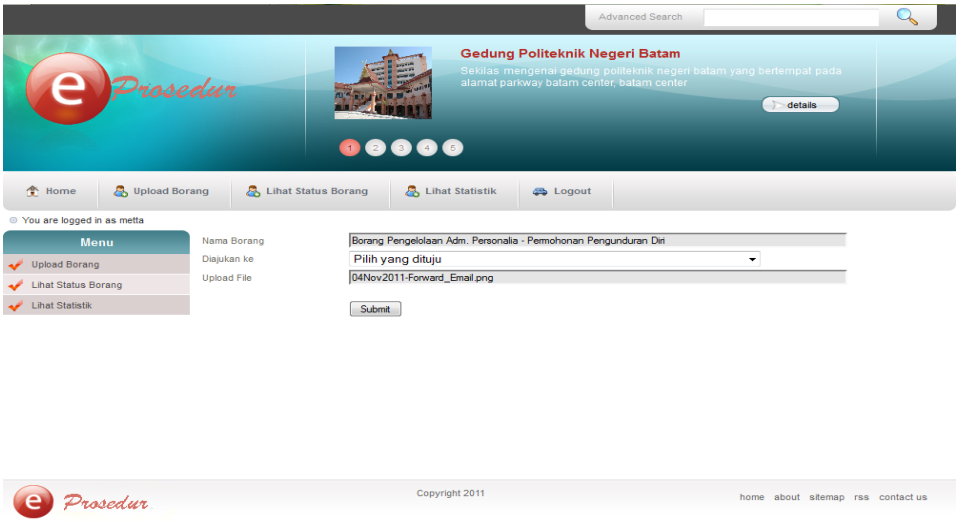
Nama Layar : L0004

Deskripsi : Layar untuk meneruskan pengajuan borang ke pihak kedua dst

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar

- textfield Nama Borang yang telah terisi nama borang yang pernah diajukan dan akan diteruskan ke pihak kedua. Nama borang ini tidak dapat diubah.
- combobox Diajukan Kepada untuk memilih pihak kedua penerima borang. dalam hal ini penerima borang adalah pengguna struktural
- texfield upload file berisi file yang pernah diajukan. File tidak dapat diubah.
- tombol submit untuk mengirim borang



Gambar 9 Pengajuan borang kepada pihak lain

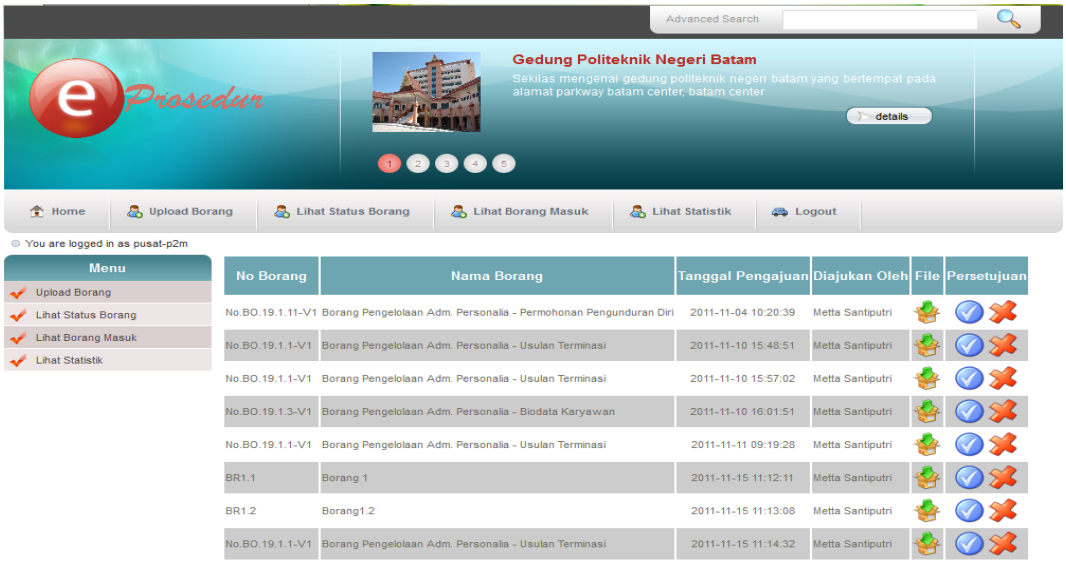
D.5 Layar pemrosesan borang

Nama Layar : L0005

Deskripsi : Layar untuk melihat daftar borang masuk

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- tabel daftar borang masuk yang berisi daftar borang yang diajukan
- icon persetujuan dan untuk menyetujui atau menolak pengajuan borang



Gambar 10 Layar Lihat Borang Masuk

D.6 Layar Statistik

Nama Layar : L0006

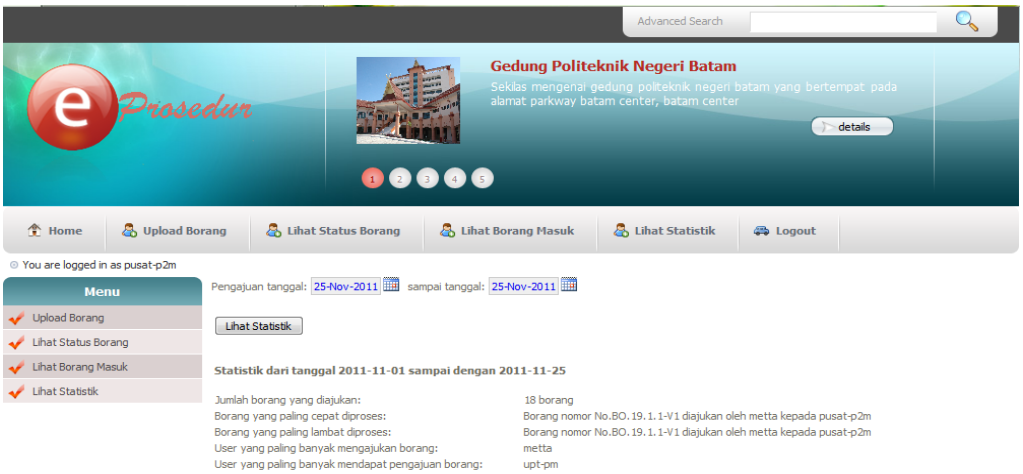
Deskripsi : Layar untuk melihat statistik pengajuan borang, yaitu

- Jumlah borang yang diajukan yaitu jumlah borang yang diupload
- Borang yang paling cepat diproses yaitu borang yang memiliki rentang waktu antara tanggal pengajuan dengan tanggal persetujuan yang paling singkat
- Borang yang paling lambat diproses yaitu borang yang memiliki rentang waktu antara tanggal pengajuan dengan tanggal persetujuan yang paling lama
- User yang paling banyak mengajukan borang yaitu pengguna yang paling banyak melakukan upload borang
- User yang paling banyak mendapat pengajuan borang yaitu pengguna yang paling banyak menjadi tujuan borang

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- textxfield calendar untuk mengisi tanggal awal dan akhir pengajuan borang yang ingin di cek

- tombol cari borang untuk mencari statistik borang sesuai dengan tanggal yang diisi
- statistik borang sesuai dengan tanggal awal dan akhir yang diinputkan



Gambar 11 Layar Lihat Statistik

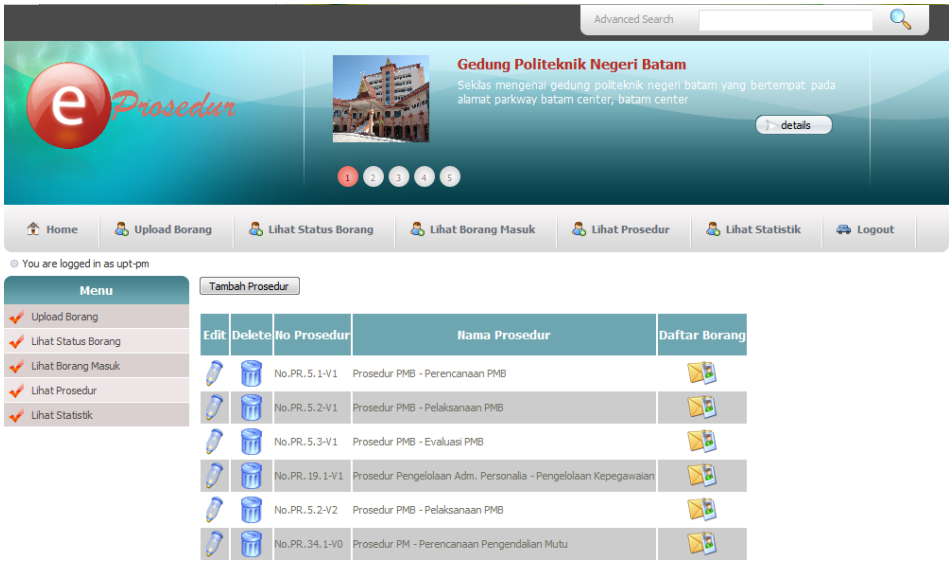
D.7 Pengaturan Prosedur dan Borang

D.7.1 Layar Lihat Prosedur

Nama Layar : L0007
Deskripsi : Layar untuk melihat daftar prosedur
Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- button Tambah Prosedur untuk menambahkan prosedur baru ke database
- tabel daftar prosedur yang berisi nama prosedur, nama prosedur dan daftar borang
- icon edit prosedur untuk melakukan pengubahan pada prosedur yang ada

- icon delete prosedur untuk melakukan penghapusan pada prosedur yang ada



Gambar 12 Layar Lihat Prosedur

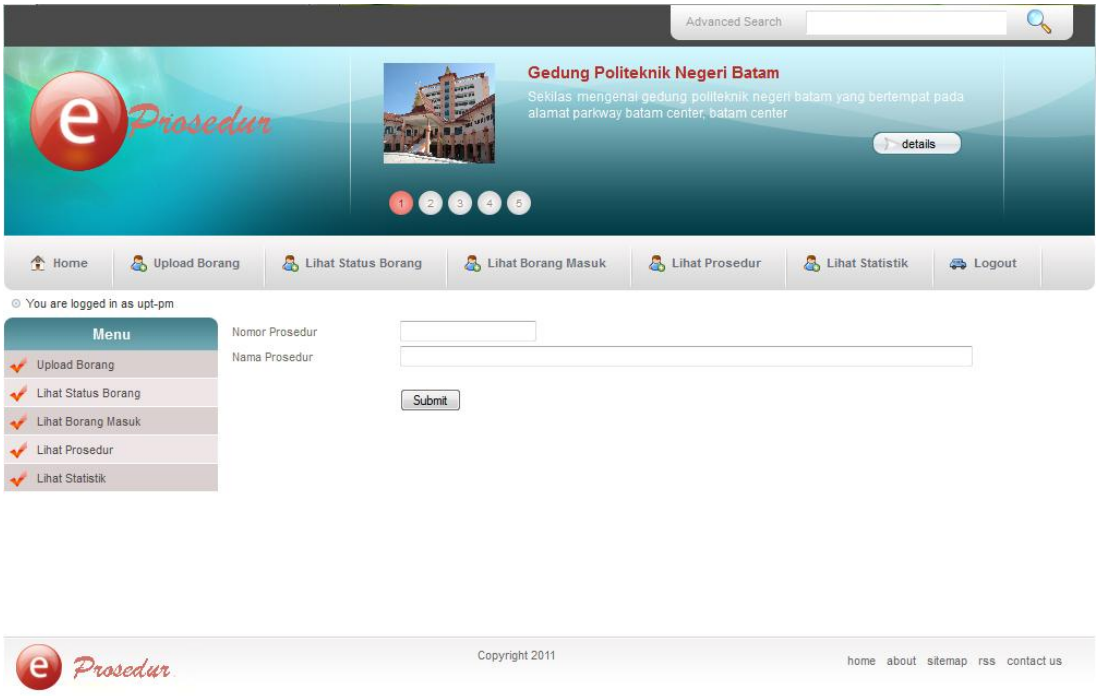
D.7.2 Layar Tambah Prosedur

Nama Layar : L0008

Deskripsi : Layar untuk menambah prosedur

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- textfield nomor prosedur untuk menginputkan nomor prosedur yang baru
- textfield nama prosedur untuk menginputkan nama prosedur yang baru
- button submit untuk menyimpan data prosedur yang baru ditambahkan



Gambar 13 Layar Tambah Prosedur

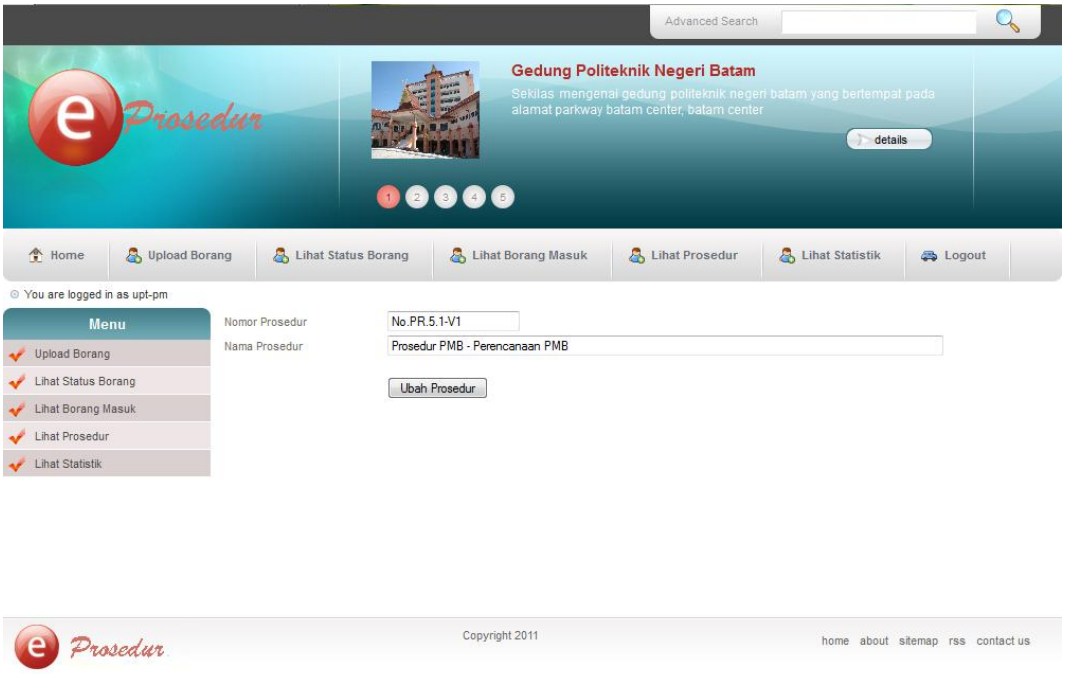
D.7.3 Layar Edit Prosedur

Nama Layar : L0009

Deskripsi : Layar untuk mengubah prosedur di database

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- textfield nomor prosedur untuk menginputkan nomor prosedur yang akan diubah
- textfield nama prosedur untuk menginputkan nama prosedur yang akan diubah
- button submit untuk menyimpan data prosedur yang baru diubah



Gambar 14 Layar Edit Prosedur

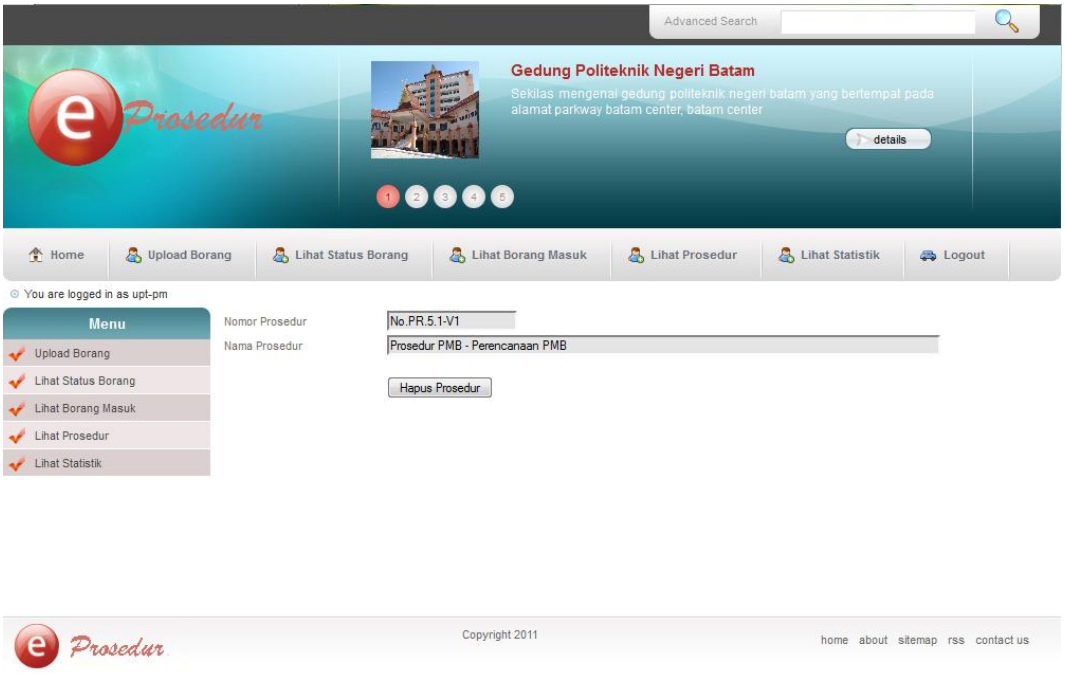
D.7.4 Layar Delete Prosedur

Nama Layar : L0010

Deskripsi : Layar untuk menghapus prosedur

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- textfield nomor prosedur yang berisi nomor prosedur yang ingin dihapus
- textfield nama prosedur yang berisi nama prosedur yang ingin dihapus
- button Hapus Prosedur untuk menghapus data prosedur yang ingin dihapus



Gambar 15 Layar Delete Prosedur

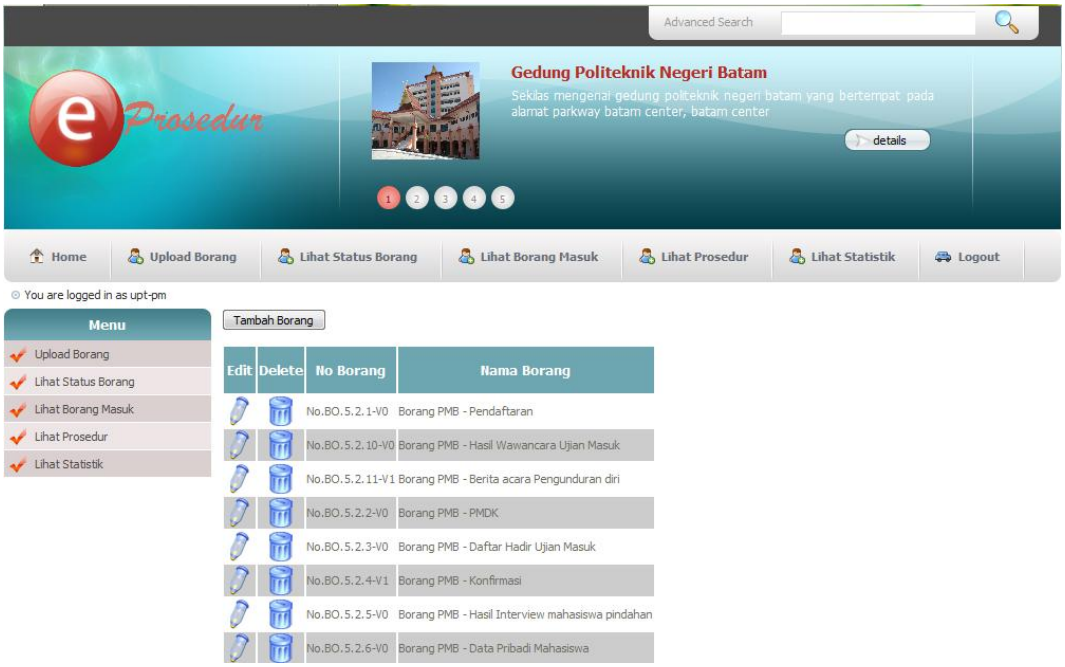
D.7.5 Layar Lihat Borang

Nama Layar : L0011

Deskripsi : Layar untuk melihat borang

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- button Tambah Prosedur untuk menambahkan prosedur baru ke database
- tabel daftar prosedur yang berisi nama prosedur, nama prosedur dan daftar borang
- icon edit prosedur untuk melakukan pengubahan pada prosedur yang ada
- icon delete prosedur untuk melakukan penghapusan pada prosedur yang ada



Gambar 16 Layar Lihat Borang

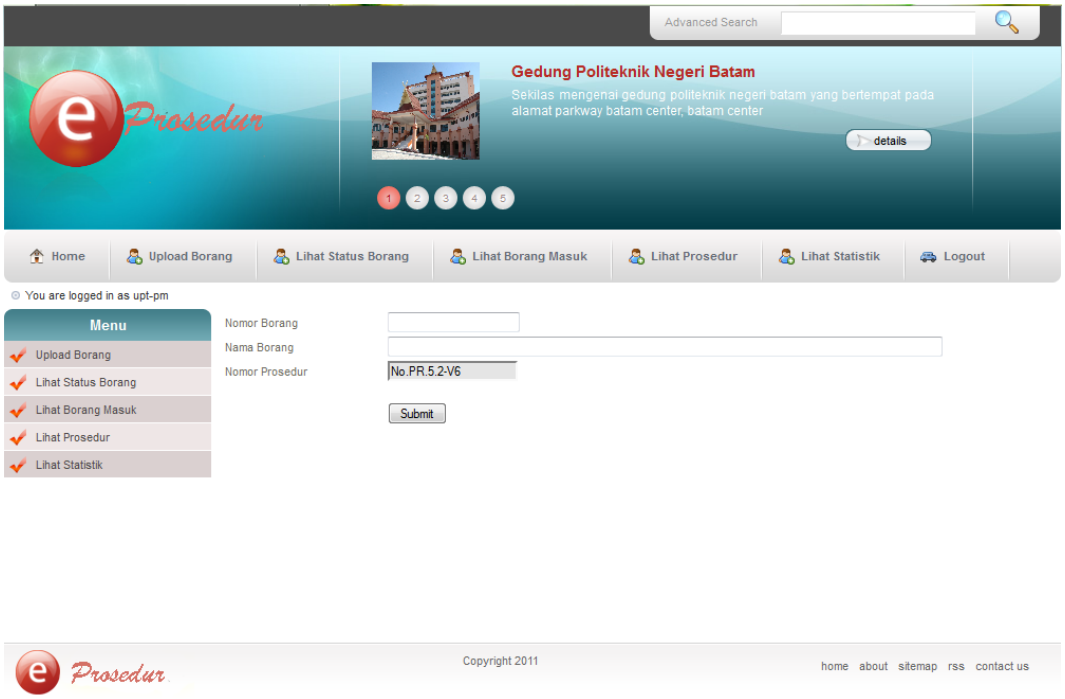
D.7.6 Layar Tambah Borang

Nama Layar : L0012

Deskripsi : Layar untuk menambah borang baru ke database

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- textfield nomor borang untuk menginputkan nomor borang yang baru
- textfield nama borang untuk menginputkan nama borang yang baru
- textfield nomor prosedur yang berisi nomor prosedur borang tersebut. Nomor prosedur ini tidak dapat diubah.
- button submit untuk menyimpan data borang yang baru ditambahkan



Gambar 17 Layar Tambah Borang

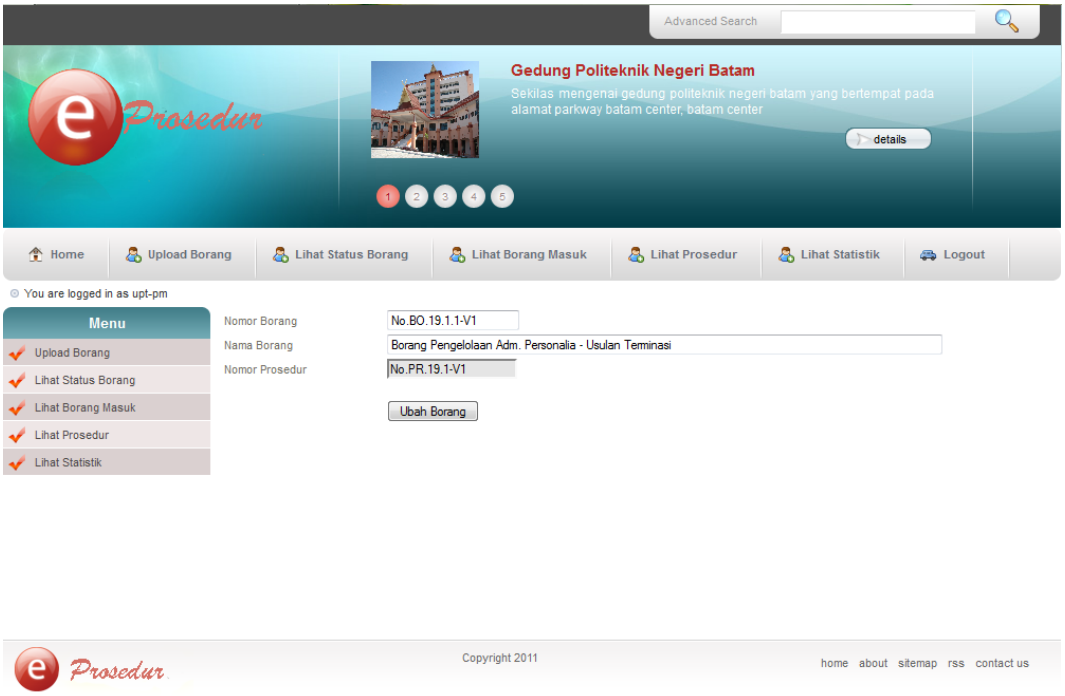
D.7.7 Layar Edit Borang

Nama Layar : L0013

Deskripsi : Layar untuk mengubah borang

Terdiri dari :

- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- textfield nomor borang untuk menginputkan nomor borang yang ingin diubah
- textfield nama borang untuk menginputkan nama borang yang ingin diubah
- textfield nomor prosedur yang berisi nomor prosedur borang tersebut. Nomor prosedur ini tidak dapat diubah.
- button Ubah Borang untuk menyimpan data borang yang baru diubah.



Gambar 18 Layar Edit Borang

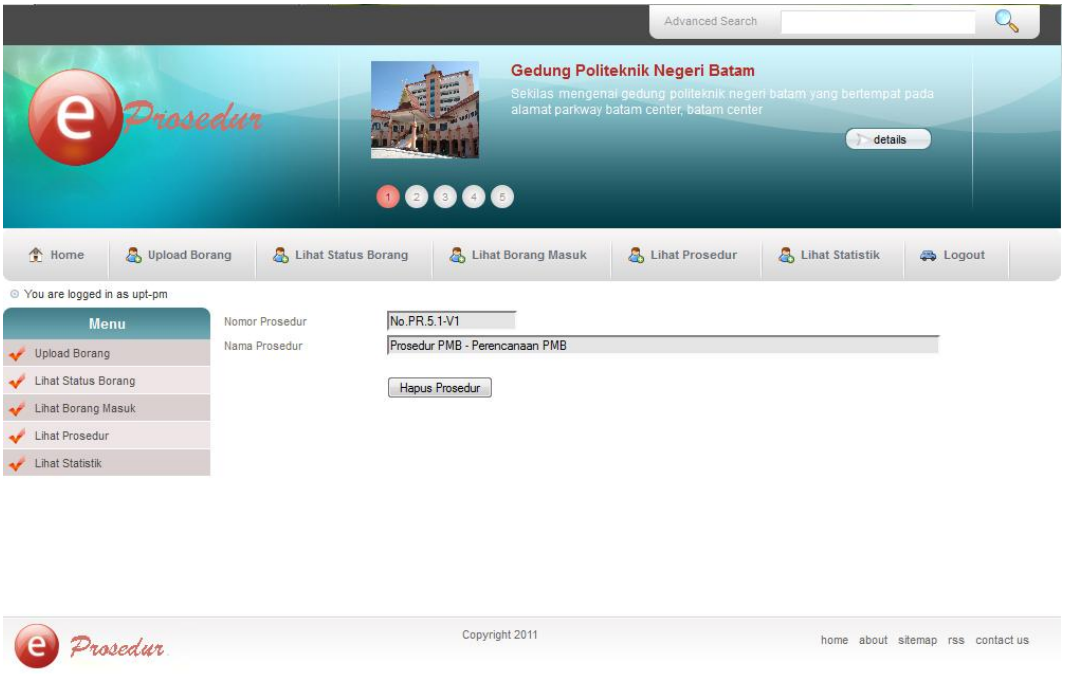
D.7.8 Layar Delete Borang

Nama Layar : L0014

Deskripsi : Layar untuk menghapus borang

Terdiri dari :

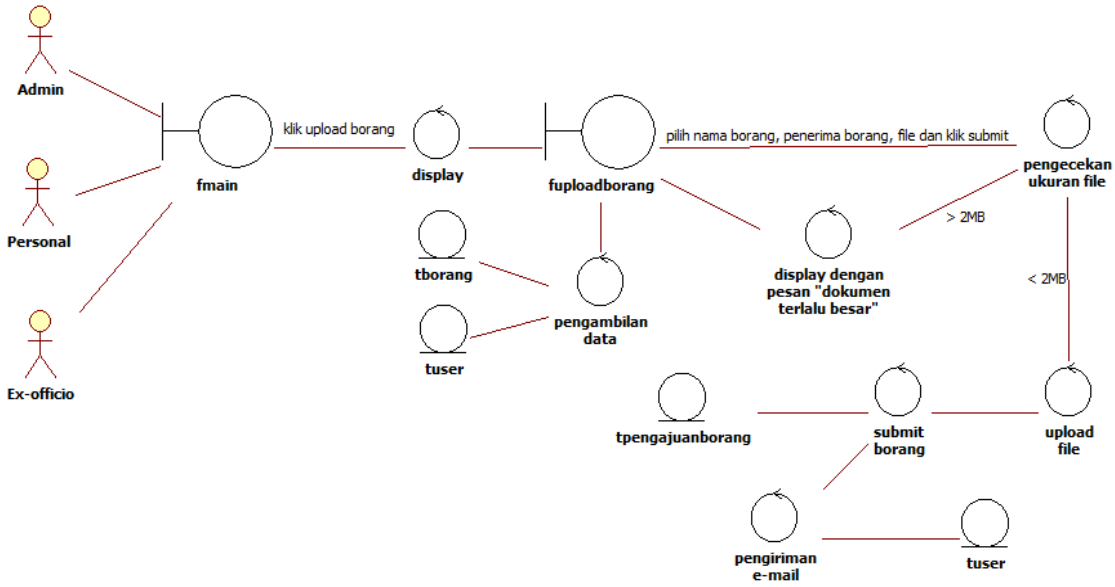
- menu-menu sesuai hak akses pengguna yang terletak di sebelah kiri layar
- textfield nomor borang yang berisi nomor borang yang ingin dihapus
- textfield nama borang yang berisi nama borang yang ingin dihapus
- textfield nomor prosedur yang berisi nomor borang yang ingin dihapus
- button Hapus Borang untuk menghapus borang di database



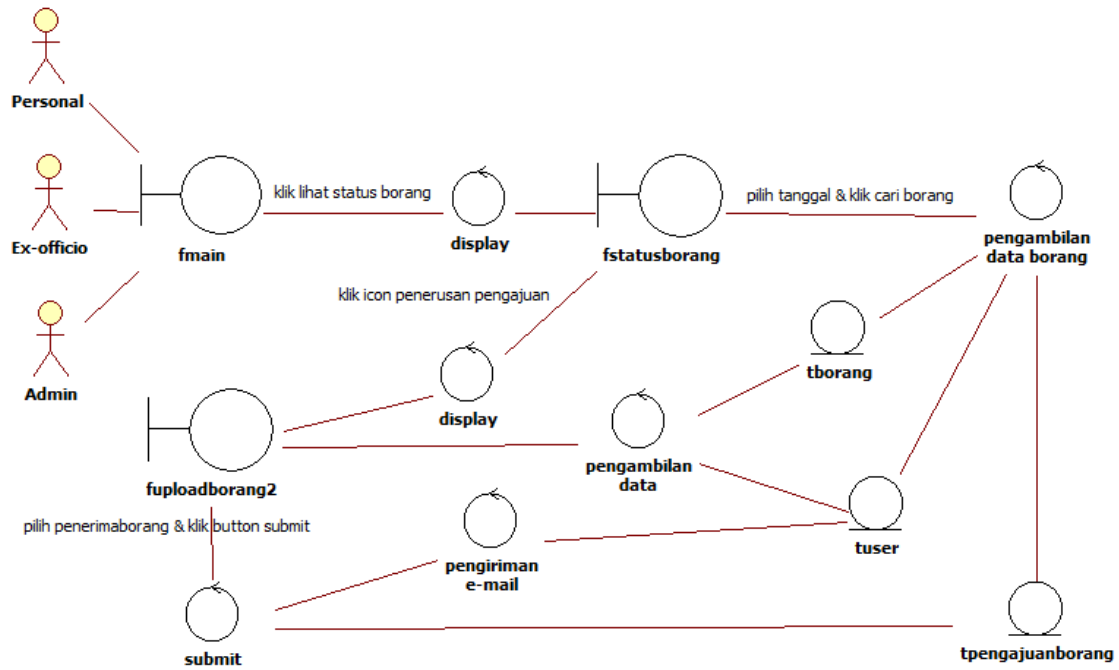
Gambar 19 Layar Delete Borang

Lampiran E Robustness Diagram

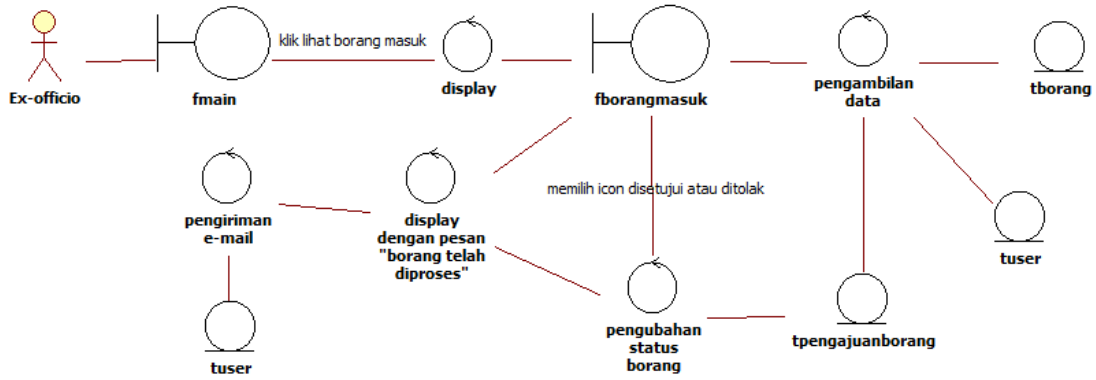
E.1 Robutness Diagram Pengajuan Borang



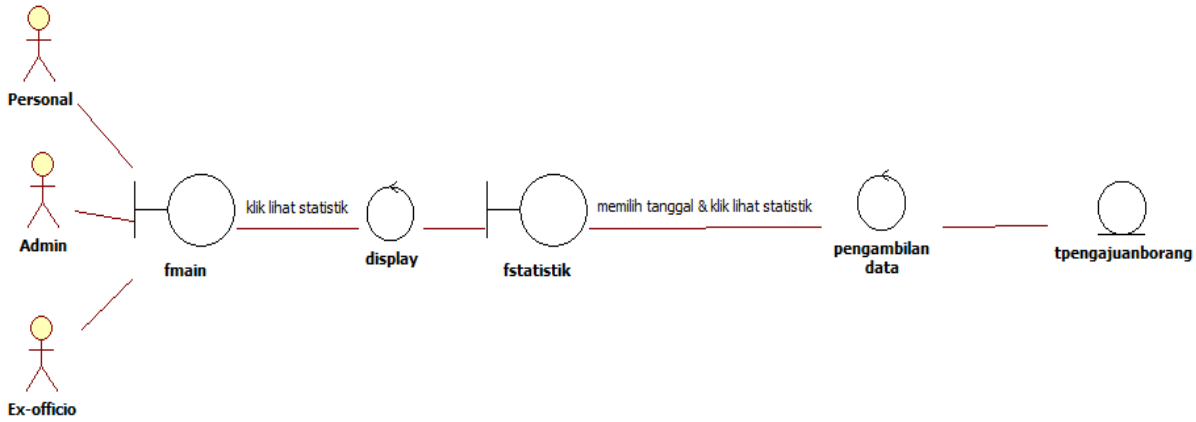
E.2 Robustness Diagram Pengecekan Status



E.3 Robustness Diagram Pemrosesan Borang

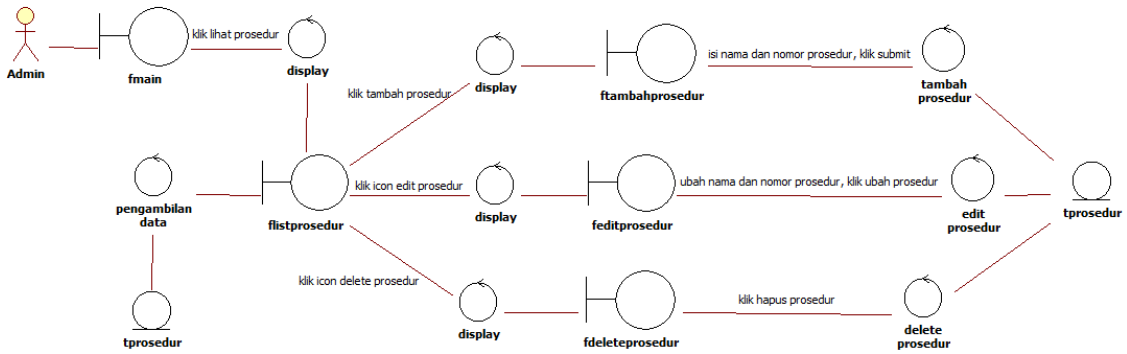


E.4 Robustness Diagram Pengecekan Statistik

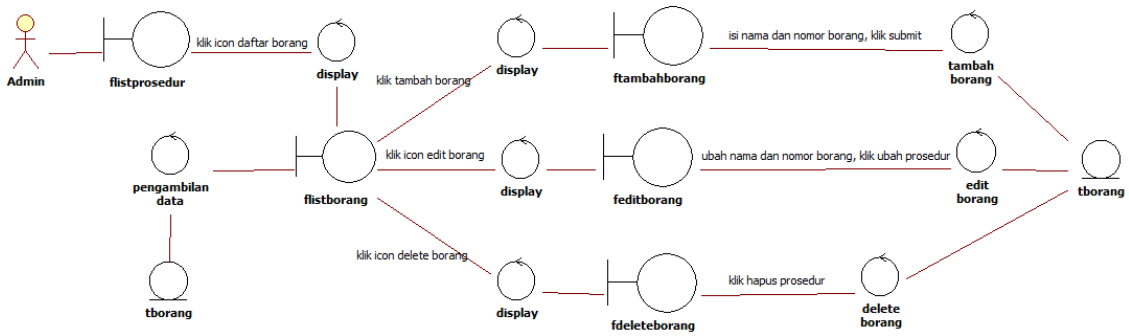


E.5 Robustness Diagram Pengaturan Prosedur dan Borang

E.5.1 Robustness Diagram Pengaturan Prosedur

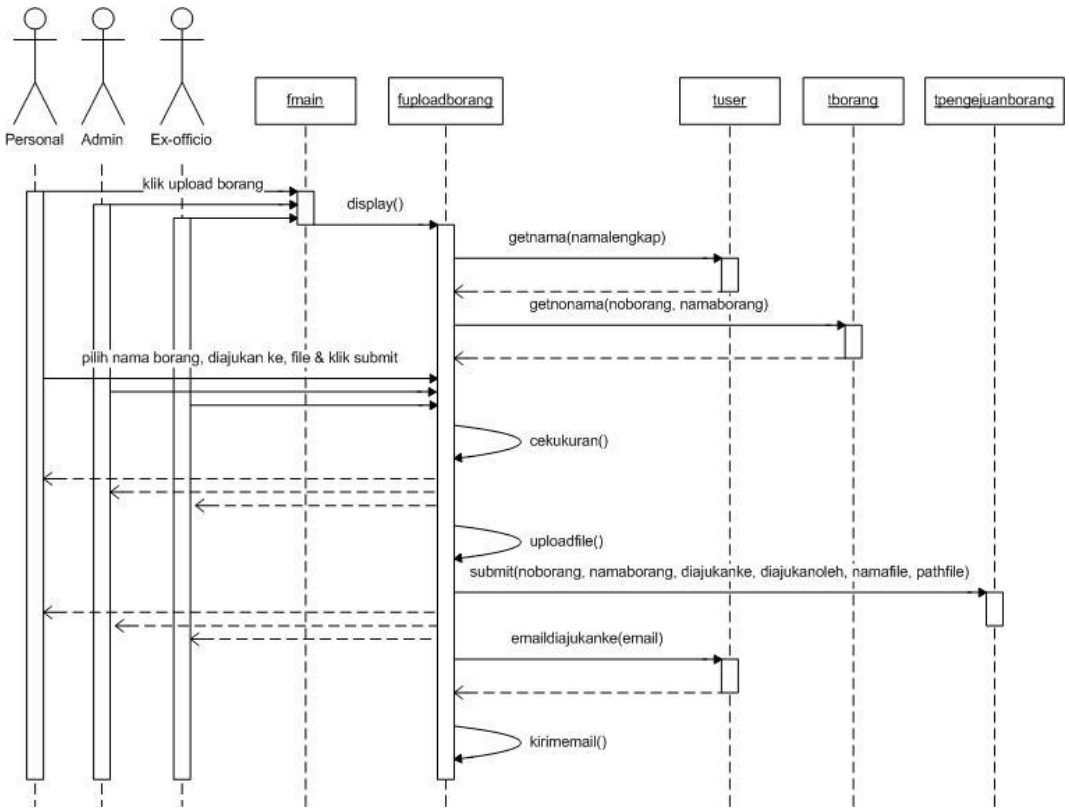


E.5.2 Robustness Diagram Pengaturan Borang

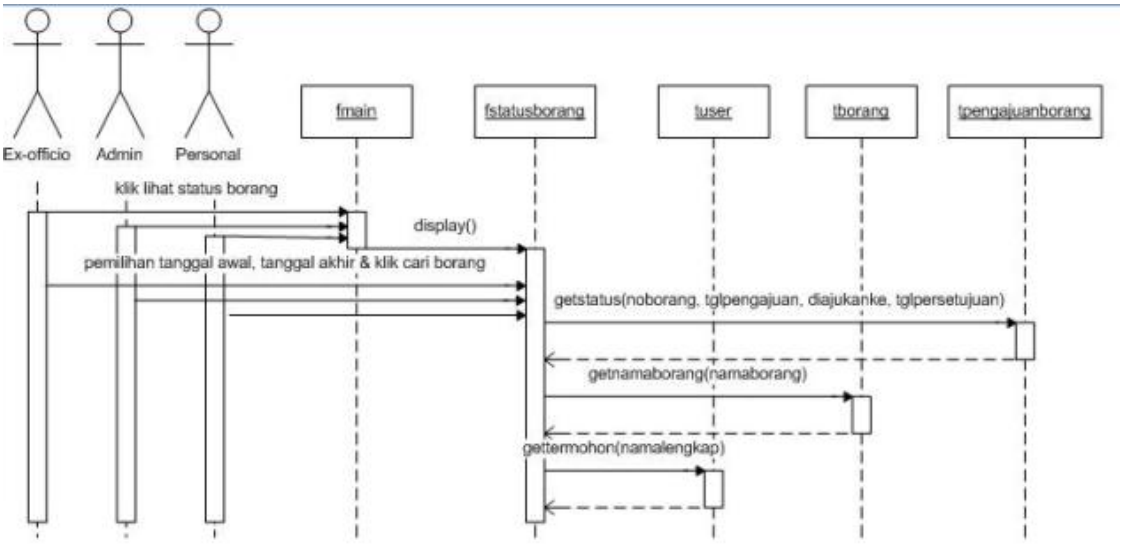


Lampiran F Sequence Diagram

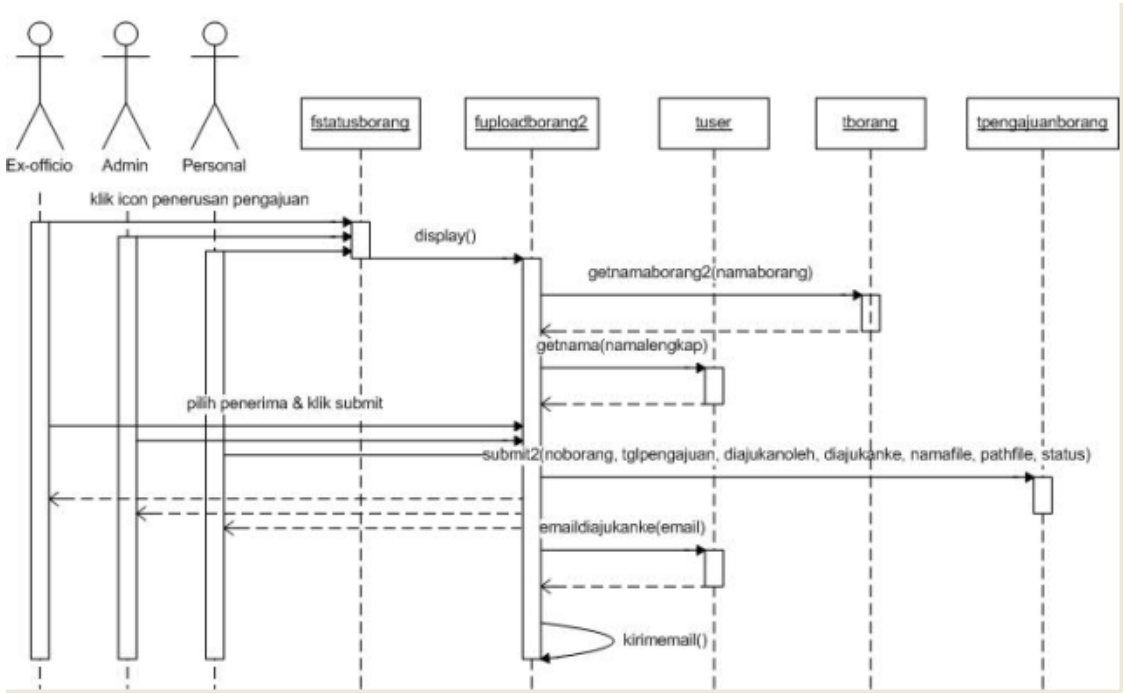
F.1 Sequence Diagram untuk Use Case Pengajuan Borang



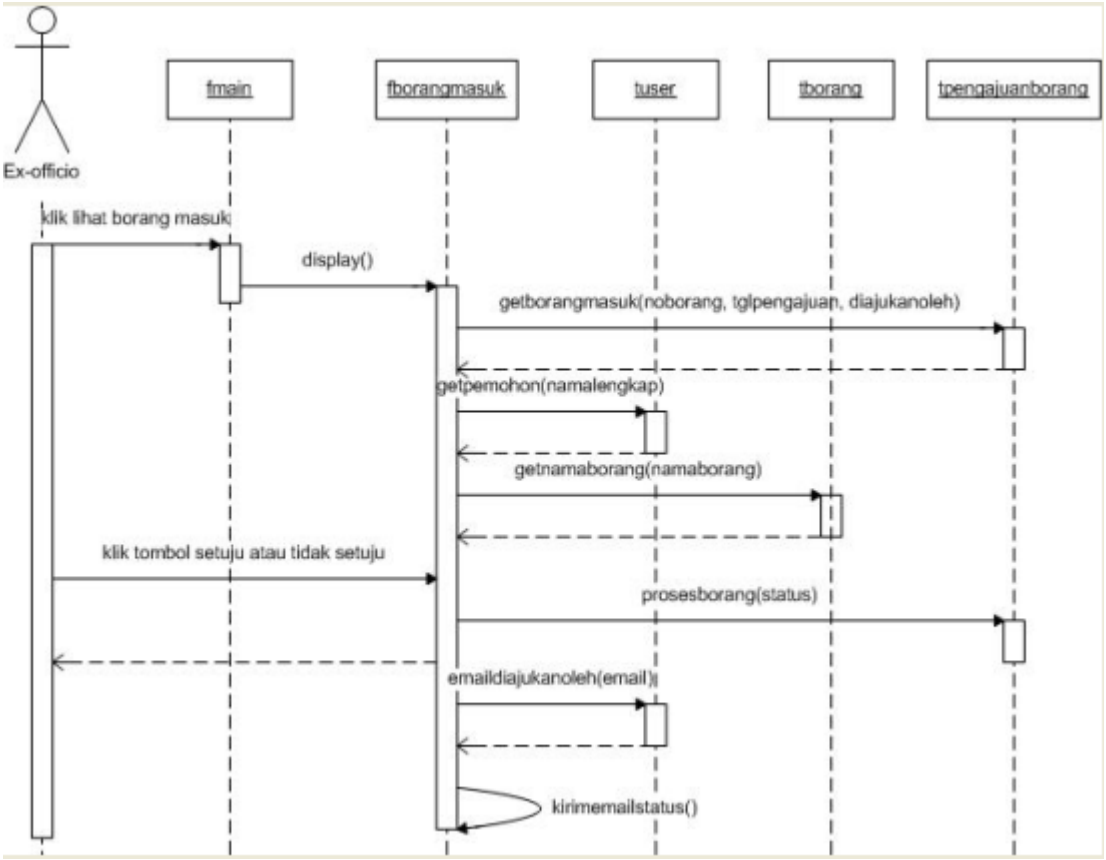
F.2 Sequence Diagram untuk Use Case Pengecekan Status Borang



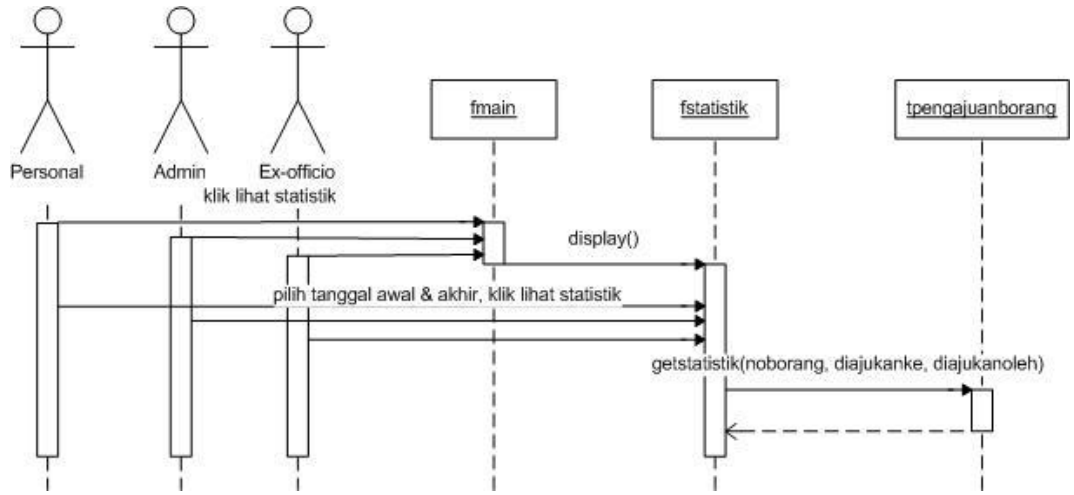
F.3 Sequence Diagram untuk Use Case Penerusan Pengajuan



F.4 Sequence Diagram untuk Use Case Pemrosesan Borang

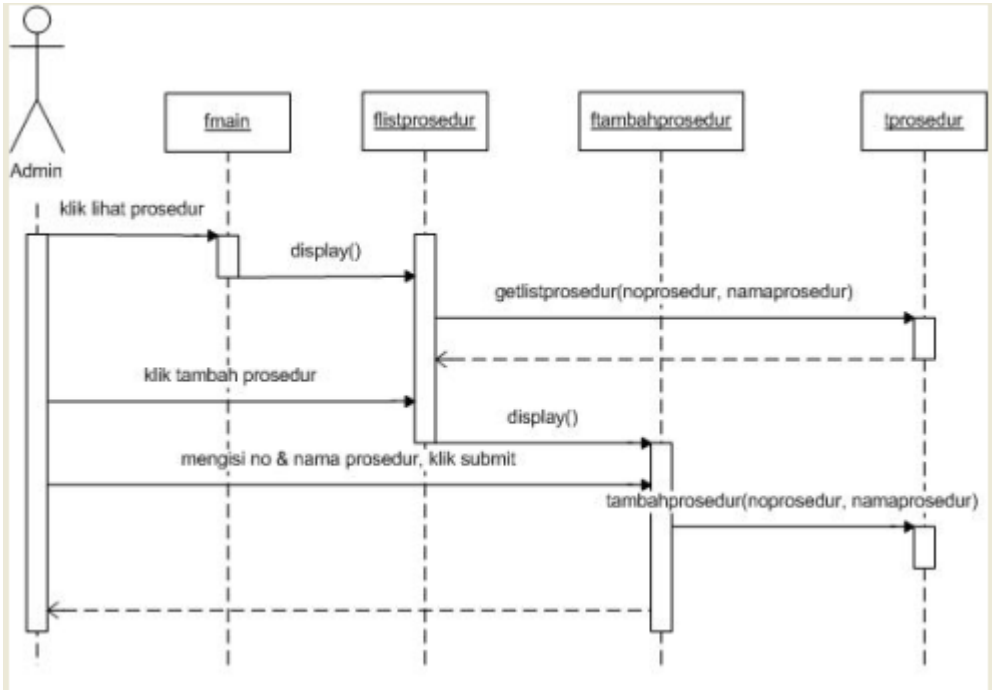


F.5 Sequence Diagram untuk Use Case Pengecekan Statistik

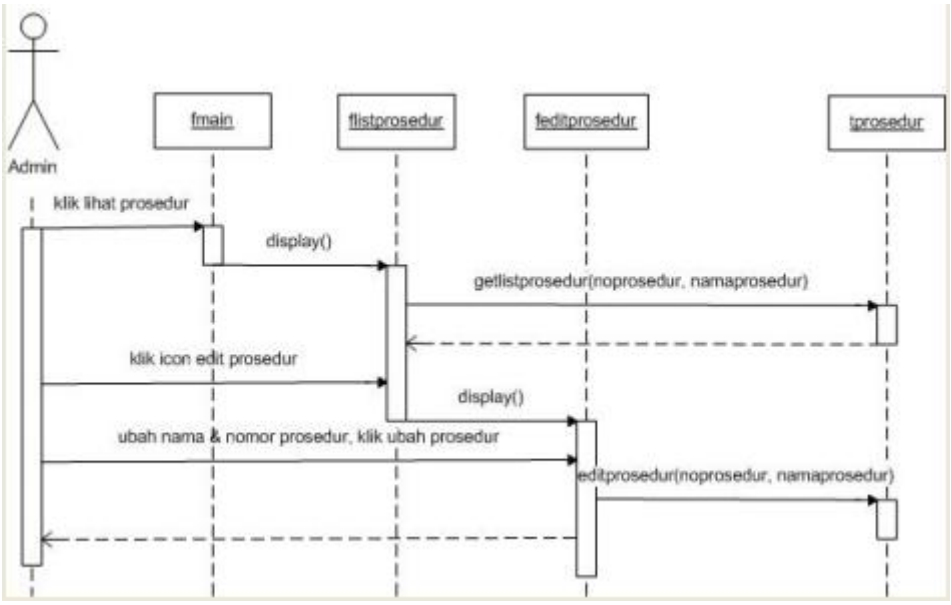


F.7 Sequence Diagram untuk Use Case Pengaturan Prosedur dan Borang

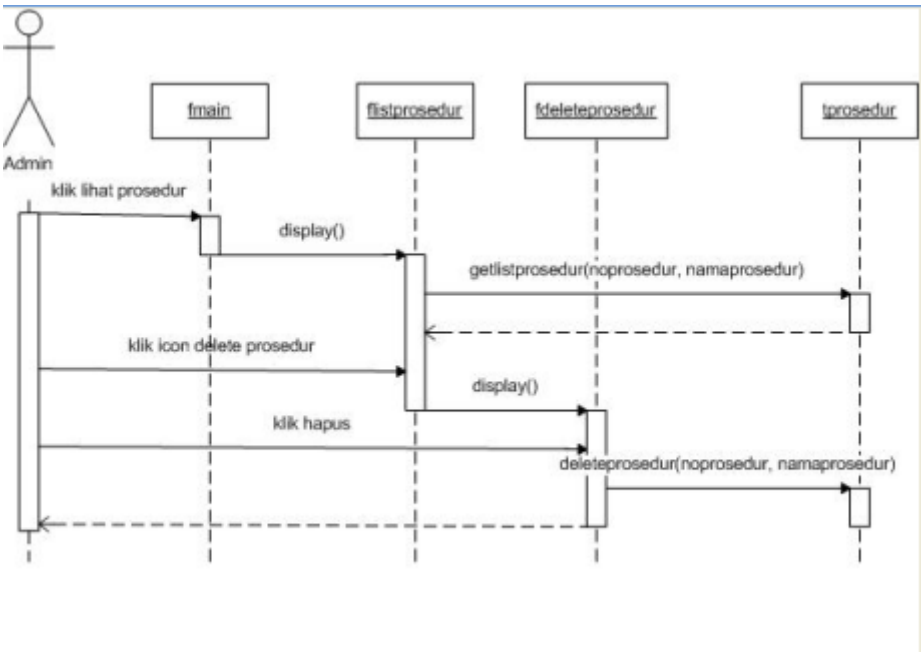
F.7.1 Sequence Diagram untuk Use Case Tambah Prosedur



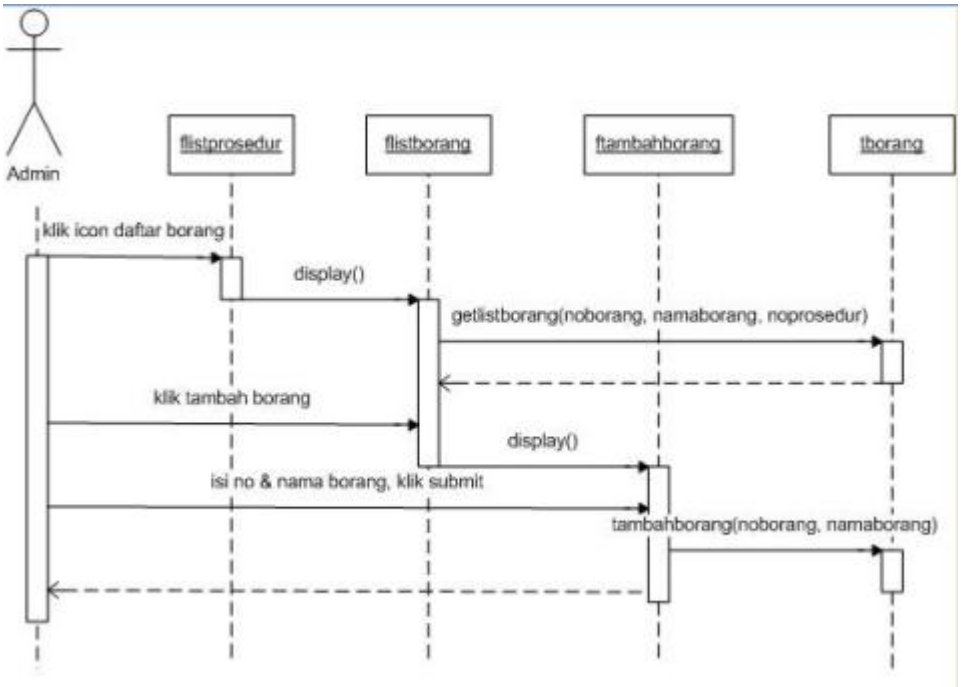
F.7.2 Sequence Diagram untuk Use Case Ubah Prosedur



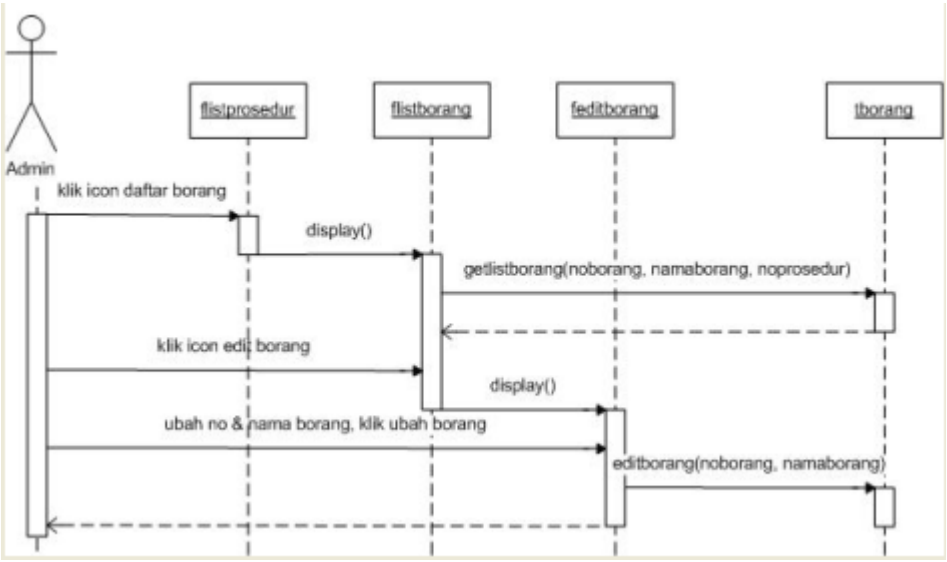
F.7.3 Sequence Diagram untuk Use Case Delete Prosedur



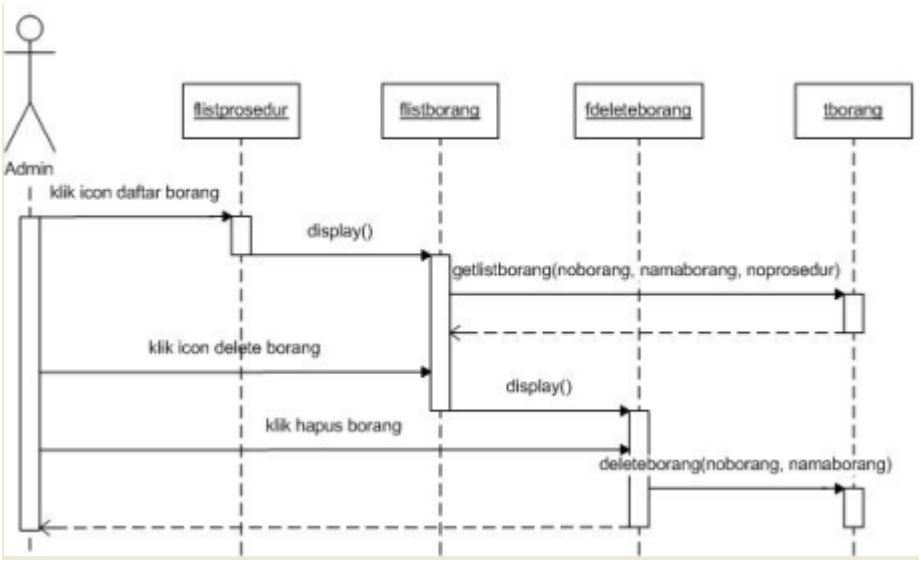
F.7.4 Sequence Diagram untuk Use Case Tambah Borang



F.7.5 Sequence Diagram untuk Use Case Ubah Borang



F.7.6 Sequence Diagram untuk Use Case Hapus Borang



Lampiran G Diagram Kelas

G.1 kelas GUI pengajuan borang

Fuploadborang
+display() +cekukuran() +uploadfile() +kirimemail()

G.2 kelas GUI pengecekan status borang

Fstatusborang
+display()

G.3 kelas GUI borang masuk

Fborangmasuk
+display() +kirimemailstatus()

G.4 kelas GUI penerusan pengajuan borang

Fuploadborang2
+display() +kirimemail()

G.5 kelas GUI lihat statistik pengajuan borang

Fstatistik
+display()

G.6 kelas GUI lihat daftar prosedur

Flistprosedur
+display()

G.7 kelas GUI tambah prosedur

Ftambahprosedur
+display()

G.8 kelas GUI edit prosedur

Feditprosedur
+display()

G.9 kelas GUI delete prosedur

Fdeleteprosedur
+display()

G.9 kelas GUI lihat daftar borang

Flistborang
+display()

G.10 kelas GUI edit borang

Feditborang
+display()

G.11 kelas GUI delete borang

Fdeleteborang
+display()

G.12 kelas pengajuan borang

Tpengajuanborang
+getstatus() +getdatamasuk() +getstatistik() +submit()

+submit2() +prosesborang()

G.13 kelas pengguna

Tuser
+emaildiajukanke() +emaildiajukanoleh() +getpemohon() +gettermohon() +getnama()

G.14 kelas borang

Tborang
+tambahborang() +editborang() +deleteborang() +getnonama() +getnamaborang() +getnamaborang2() +getlistborang()

G.15 kelas prosedur

Tprosedur
+tambahprosedur() +editprosedur() +deleteprosedur() +getlistprosedur()

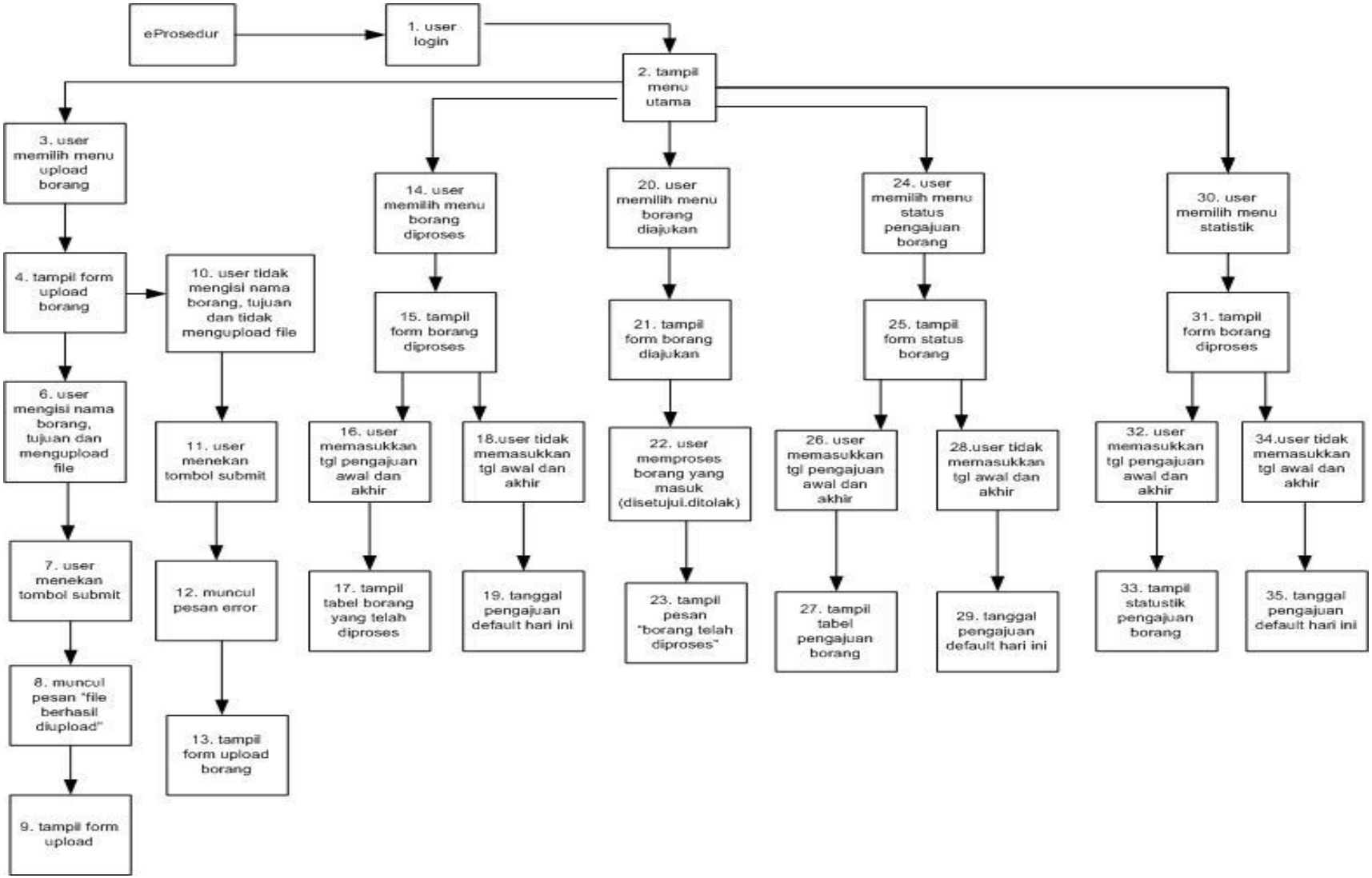
Lampiran H Source Code

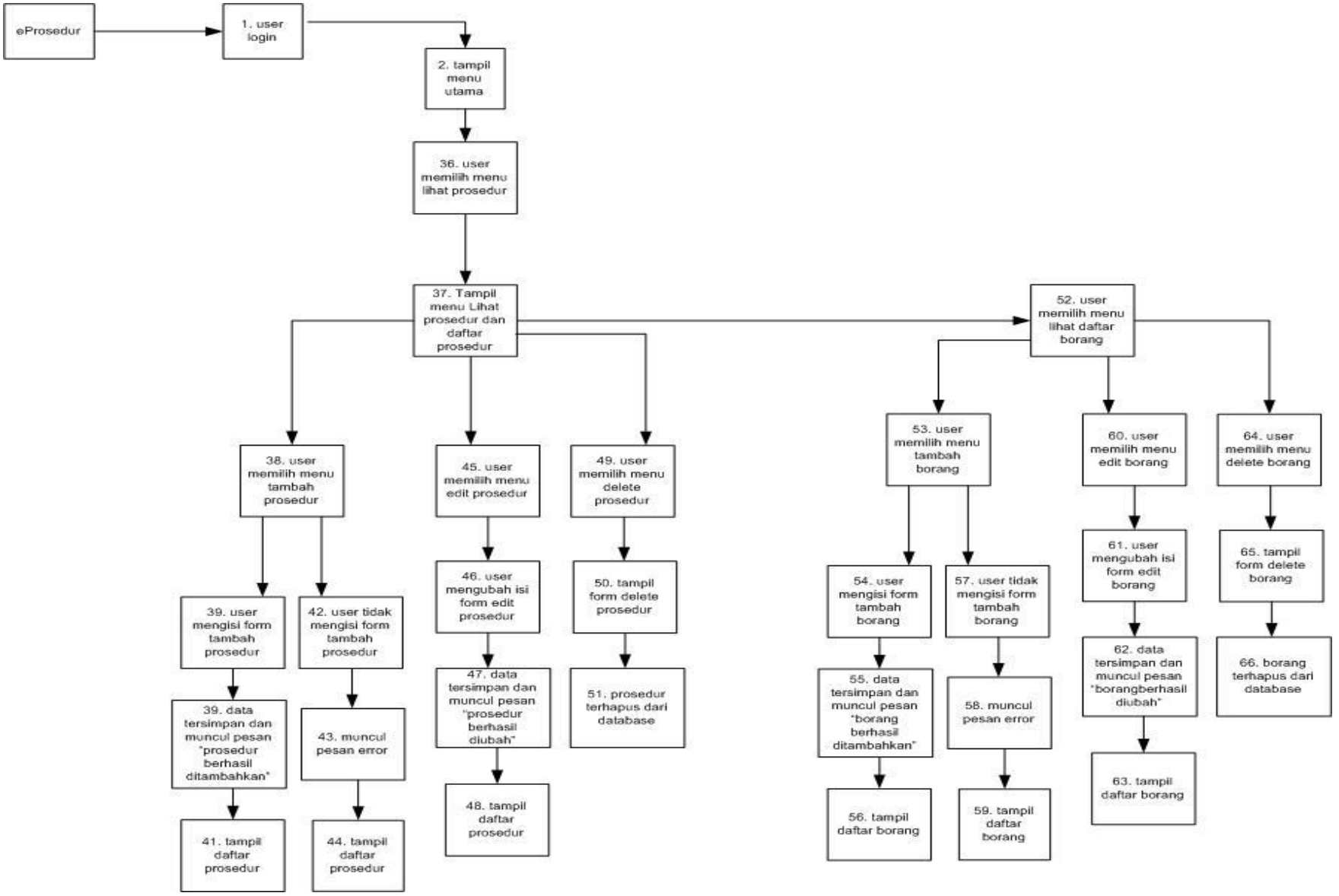
```php
<?php

// starting a session
session_start();
include "koneksi.php";

class fuploadborang
{

public function display() {

if (!isset($_SESSION['username'])) {
    echo    "<script>alert('Maaf, Anda harus login terlebih dahulu untuk masuk ke halaman ini');
        parent.location='index.html';
        </script>";
    exit;
}

$username = $_SESSION['username'];

?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252" />
<title>eProsedur</title>
<link rel="stylesheet" type="text/css" href="style.css" />
<!--[if IE 6]>
<link rel="stylesheet" type="text/css" href="iecss.css" />
<![endif]-->
<script type="text/javascript" src="js/boxOver.js"></script>
</head>
<body>
```

```
// display nomor borang
$query = "SELECT noborang,namaborang FROM tborang ORDER BY noborang";
$result = pg_query($query);

?>
                    <select name="noborang">
                    <option value="" selected>Pilih nama borang</option>
                    <?
                        while($row=pg_fetch_array($result)) {
                            $a = $row['noborang'];
                            $b = $row['namaborang'];
                            echo "<option value=\"$a\">$b</option>\n";
                        }
                    ?>
                </td>
            </tr>
            <tr>
                <td width="150">Diajukan kepada</td>
                <td>
<?php

// display nama lengkap user yang merupakan jabatan struktural
$query = "SELECT * FROM tuser WHERE struktural = '1' ORDER BY namalengkap";
$result = pg_query($query);

?>
                    <select name="struktural">
                    <option value="" selected>Pilih yang dituju</option>
                    <?
                        while($row=pg_fetch_array($result)) {
                            $a = $row['username'];
                            $b = $row['namalengkap'];
                            echo "<option value=\"$a\">$b</option>\n";
                        }

class up{

public function sub(){

//This function separates the extension from the rest of the file name and returns it
function findexts ($filename) {
    $filename = strtolower($filename);
    $exts = split("[/\\.]", $filename);
    $n = count($exts)-1;
    $exts = $exts[$n];
    return $exts;
}

if (!isset($_SESSION['username'])) {
    echo    "<script>alert('Maaf, Anda harus login terlebih dahulu untuk masuk ke halaman ini');
        parent.location='index.html';
        </script>";
    exit;
}

$username = $_SESSION['username'];

// nama direktori
//$date = date('dMY');
//$dir="./".$_SESSION['username']."/".$date;
$dir="./dok/".$_SESSION['username'];
if(!is_dir($dir)) {
    mkdir($dir,0777);
}

$action = $_POST["action"];
$max_size = "20000000";
```

```php
$action = $_POST["action"];
$max_size = "20000000";

if ($action == 'upload') {
    if ($_FILES["filename"]["size"] > $max_size)
        die ("<b>Dokumen terlalu besar! Silahkan coba lagi...</b>");
    $noborang=$_POST["noborang"];
    $struktural=$_POST["struktural"];
    $pengirim = $_SESSION['username'];
    $tanggal = date ("d F Y h:i:s A");
    $namaFile = $_FILES["filename"]["name"];
    $namaFile = str_replace(" ","_","$namaFile");
    $date = date('dMY');
    $namaFile = $date."-".$namaFile;
    $uploadfile = $dir."/".$namaFile;
    move_uploaded_file($_FILES["filename"]["tmp_name"],$uploadfile);
    $simpan="INSERT INTO tpengajuanborang(noborang,tglpengajuan,diajukanoleh,diajukanke,namafile,pathfile,status)
            VALUES ('$noborang','$tanggal','$pengirim','$struktural','$namaFile','$dir','0')";
    pg_query($simpan);

    echo "<script>alert('File Berhasil diUpload.');
        parent.location='fuploadborang.php';
        </script>";

}

pg_close();
}}
$up = new up();
$up->sub();
?>
```

Lampiran I Skenario Pengujian




eProsedur
1. user login
2. tampil menu utama
36. user memilih menu lihat prosedur
37. Tampil menu Lihat prosedur dan daftar prosedur
38. user memilih menu tambah prosedur
39. user mengisi form tambah prosedur
39. data tersimpan dan muncul pesan "prosedur berhasil ditambahkan"
41. tampil daftar prosedur
42. user tidak mengisi form tambah prosedur
43. muncul pesan error
44. tampil daftar prosedur
45. user memilih menu edit prosedur
46. user mengubah isi form edit prosedur
47. data tersimpan dan muncul pesan "prosedur berhasil diubah"
48. tampil daftar prosedur
49. user memilih menu delete prosedur
50. tampil form delete prosedur
51. prosedur terhapus dari database
52. user memilih menu lihat daftar borang
53. user memilih menu tambah borang
54. user mengisi form tambah borang
55. data tersimpan dan muncul pesan "borang berhasil ditambahkan"
56. tampil daftar borang
57. user tidak mengisi form tambah borang
58. muncul pesan error
59. tampil daftar borang
60. user memilih menu edit borang
61. user mengubah isi form edit borang
62. data tersimpan dan muncul pesan "borangberhasil diubah"
63. tampil daftar borang
64. user memilih menu delete borang
65. tampil form delete borang
66. borang terhapus dari database

DAFTAR PUSTAKA

Hanya berisi daftar karya orang lain yang diacu dalam dokumen TA

1. Gannod, c k., Sudindranath, Gora., Fagnani, Mark E., dan Cheng, Betty H.C., "Packrat: A Software Reengineering Case Study", 1998
2. Frakes, B. William.,Kulczcky, Gregory., and Mooadliar, Natasha.," An Empirical Comparison of Method for Reengineering Procedural Software System to Object-Oriented System", 2008.
3. Ying Zou, "Technique and Methodologies for the Migration of Legacy Sytem to Object Oriented Platform", 2003.
4. Chikofsky and J.H. Cross, "Reverse Engineering and Design Recover: A Taxonomy," IEEE Software, vol. 7, pp.13-17, January 1990.
5. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, "Perspectivr on Legacy System Engineering", 1995.
6. Karona Consulting Ltd, "Use Case Includes and Extends V0.4.doc".
7. Kusuma, Wahyu Andika, S.kom, "Object Oriented Reengineering Patterns and Techniques".
8. Zandstra, Matt, "PHP Objects, Patterns, and Practice", 2008.