DEPARTEMEN PENDIDIKAN NASIONAL
UNIVERSITAS INDONESIA
FAKULTAS EKONOMI
PROGRAM STUDI MAGISTER MANAJEMEN

KARYA AKHIR

ESTIMASI UKURAN PROYEK SISTEM INFORMASI MENGGUNAKAN *USE CASE POINT* DENGAN STUDI KASUS PENGEMBANGAN SISTEM INFORMASI PEMBUKUAN DAN SUMBER DAYA MANUSIA UNTUK PT TAWANG SWASTI RAWIKARA

Diajukan oleh

Kwarta Fitra Rachmilliza

6605521497

UNTUK MEMENUHI SEBAGIAN DARI SYARAT-SYARAT
GUNA MENCAPAI GELAR
MAGISTER MANAJEMEN
2008

UNIVERSITAS INDONESIA

FAKULTAS EKONOMI

PROGRAM STUDI MAGISTER MANAJEMEN

TANDA PERSETUJUAN KARYA AKHIR

Nama : **KWARTA FITRA RACHMILLIZA**

Nomor Mahasiswa : **6605521497**

Konsentrasi : **Manajemen Operasi**

Judul Karya Akhir : **ESTIMASI UKURAN PROYEK SISTEM INFORMASI MENGGUNAKAN USE CASE POINT DENGAN STUDI KASUS PENGEMBANGAN SISTEM INFORMASI PEMBUKUAN DAN SUMBER DAYA MANUSIA UNTUK PT TAWANG SWASTI RAWIKARA**

Ketua Program Studi

Tanggal ..... Magister Manajemen **Rhenald Kasali, Ph.D**

Tanggal ..... Pembimbing Karya Akhir : **Dr. Setyo H. Wijanto**

MAGISTER MANAJEMEN
FAKULTAS EKONOMI · UNIVERSITAS INDONESIA

BERITA ACARA
PRESENTASI KARYA AKHIR

Pada hari *MINGGU*, tanggal *06 APRIL 2008*, telah dilaksanakan presentasi Karya Akhir dari mahasiswa dengan

Nama : Kwarta Fitra Rachmilliza

No. Mhs : 6605521497

Konsentrasi : Manajemen Operasi - Malam

Presentasi tersebut diuji oleh tim penguji yang terdiri dari :

Nama :	Tanda Tangan :
1. **Dr. Mohammad Hamsal** (Ketua)	.....
2. **Yudho Giri Sucahyo, Ph.D.** (Anggota 1)	.....
3. **Dr. Setyo H. Wijanto** (Anggota 2/ Pembimbing)	.....

Mengetahui,

Ratna Wardani, MM
Kepala Bagian Administrasi Akademik

SURAT PERNYATAAN KEASLIAN KARYA AKHIR

Saya yang bertanda tangan di bawah ini:

Nama : Kwarta Fitra Rachmilliza

No. Mahasiswa : 66 05 52 14 97

Konsentrasi : Manajemen Operasi

Dengan ini menyatakan sebagai berikut:

1) Karya akhir yang berjudul: **ESTIMASI UKURAN PROYEK SISTEM INFORMASI MENGGUNAKAN *USE CASE POINT* DENGAN STUDI KASUS PENGEMBANGAN SISTEM INFORMASI PEMBUKUAN DAN SUMBER DAYA MANUSIA UNTUK PT TAWANG SWASTI RAWIKARA**

   Penelitian yang terkait dengan karya akhir ini adalah hasil dari kerja saya sendiri.

2) Setiap ide atau kutipan dari karya orang lain baik berupa publikasi atau bentuk lainnya dalam karya akhir ini, telah diakui sesuai dengan standar prosedur referensi dalam disiplin ilmu.

3) Saya juga mengakui bahwa karya akhir ini dapat dihasilkan berkat bimbingan dan dukungan penuh oleh pembimbing saya, yaitu: **Bapak Dr. Setyo H. Wijanto.**

Apabila di kemudian hari dalam karya akhir ini ditemukan hal-hal yang menunjukkan telah dilakukannya kecurangan akademik oleh saya, maka gelar akademik saya yang telah saya dapatkan akan ditarik sesuai dengan ketentuan dari Program Magister Manajemen Fakultas Ekonomi Universitas Indonesia.

Jakarta, Mei 2008

Kwarta Fitra Rachmilliza

KATA PENGANTAR

Puji syukur ke hadirat Allah SWT atas berkah dan karunia-Nya, penulis dapat menyelesaikan karya akhir ini. Penulisan karya akhir ini dimaksudkan untuk menambah wawasan bagi penulis maupun pembaca karya akhir ini mengenai penerapan salah satu metode estimasi ukuran proyek pengembangan sistem informasi manajemen untuk PT Tawang Swasti Rawikara. Semoga karya akhir ini dapat memberikan manfaat bagi pembacanya.

Penulis mengucapkan terima kasih dan penghargaan yang setinggi-tingginya kepada Bapak DR. Setyo H. Wijanto selaku pembimbing karya akhir, yang dengan sabar membimbing, memberikan konsultasi dan saran-saran yang dibutuhkan oleh penulis dalam perjuangannya menyelesaikan karya akhir ini.

Penulis juga mengucapkan terima kasih kepada Ayah, Ibu, Abang, Engah, Kiyai, Merry dan Danu atas doa, dorongan semangat dan kasih sayang yang tak henti-hentinya diberikan kepada penulis untuk menyelesaikan karya akhir ini dan meraih gelar master di MMUI.

Karya akhir ini kupersembahkan khusus untuk istriku yang tercinta, Fardila Astari Rachmilliza yang selalu sabar mendampingi dan tidak bosan-bosanya menyemangati penulis untuk menyelesaikan karya akhir ini dan untuk Shakeela Meira Rachmilliza, kehadiranmu di dunia merupakan berkah yang tak ternilai bagi penulis.

Untuk semua teman-teman di MMUI angkatan 2005, semua dosen dan staff MMUI, Adpen, perpustakaan dan pihak-pihak lain yang tidak bisa penulis sebutkan satu persatu, penulis sampaikan ucapan terima kasih sebesar-besarnya atas semua bantuan, bimbingan yang penulis terima sehingga penulis dapat mengikuti pendidikan di MMUI dengan baik. Akhir kata, mohon maaf sebesar-besarnya kepada semua

pihak atas segala kesalahan dalam perkataan dan perbuatan penulis baik disengaja maupun tidak, selama penulis mengikuti pendidikan di MMUI.

Jakarta, Mei 2008

Penulis

RINGKASAN EKSEKUTIF

Banyak proyek teknologi informasi yang dikerjakan di seluruh dunia menghadapi tiga masalah penting, yaitu: *late delivery*, *over budget*, dan *out of scope*. Hal ini memaksa para profesional di dunia teknologi informasi untuk mengkaji ulang prinsip-prinsip yang digunakan pada manajemen proyek di industri teknologi informasi.

Salah satu tahapan penting dari manajemen proyek di industri teknologi informasi adalah estimasi ukuran proyek. Estimasi terhadap ukuran proyek perlu dilakukan untuk menilai apakah *resource* yang dikeluarkan untuk mengembangkan atau membeli suatu sistem informasi sebanding dengan *value* yang diberikan sistem informasi kepada organisasi. Hasil dari estimasi akan digunakan untuk menentukan apakah proyek pengembangan atau akuisisi sistem baru layak untuk diteruskan atau tidak. Hal ini perlu dilakukan oleh organisasi untuk menghindari investasi besar yang ternyata tidak memberikan manfaat yang sebanding kepada organisasi.

Di sisi pengembang, proses estimasi digunakan selain untuk mengukur besarnya proyek yang akan dikerjakan, juga membantu pihak pengembang dalam menentukan *pricing* terhadap proyek, menghitung jangka waktu penyelesain proyek serta menjadi acuan dalam membuat perencanaan alokasi *human resources* untuk tiap proyek yang saat ini sedang dikerjakan. PT Imani Prima, tempat penulis bekerja, saat ini mendapatkan sebuah proyek pengembangan sebuah sistem informasi manajemen untuk PT Tawang Swasti Rawikara *Holding Company* yang berkantor di Jakarta Selatan. Untuk membantu mendefinisikan *pricing* dan lamanya waktu yang dibutuhkan untuk mengembangkan sistem informasi tersebut, penulis mengadakan

penelitian untuk menerapkan salah satu metode estimasi ukuran proyek *Use Case Point* pada proyek pengembangan sistem informasi manajemen ini.

Dalam penelitian ini, Penulis memilih menggunakan *tools use case* yang merupakan salah satu *tools* yang bagus dan disediakan oleh UML versi 2.0 untuk menganalisis suatu sistem informasi selain juga karena *use case points* menghitung ukuran sebuah sistem berdasarkan *use case* yang terdefinisi. Berdasarkan pengalaman penulis dalam mengembangkan beberapa sistem informasi, *use case* menawarkan analisis dan perancangan yang lebih dalam untuk memahami lingkungan sistem informasi, fungsi apa saja yang dilakukan oleh sistem informasi, siapa saja yang akan berinteraksi dengan sistem dan bagaimana interaksi sistem informasi yang baru ini kepada sistem eksternal lain yang telah diimplementasikan sebelumnya.

Estimasi dilakukan dengan menggunakan dua skenario. Skenario pertama, PT Imani Prima akan mengerjakan proyek ini dengan menggunakan tenaga *part time* atau kontrak yang belum memiliki pengalaman sama sekali dalam mengembangkan sebuah sistem informasi yang memiliki fitur cukup banyak dan terintegrasi. Skenario kedua, PT Imani Prima menugaskan pegawai tetap perusahaan yang telah memiliki pengalaman dalam mengembangkan sistem informasi dan biasa menggunakan Microsoft Visual Basic 6.0, SQL Server Express serta Rational Rose 2002 sebagai *tools* yang akan digunakan untuk mengembangkan sistem informasi tersebut. Berdasarkan hasil estimasi terhadap kedua skenario ditambah dengan faktor resiko sebesar 10%, ternyata skenario I membutuhkan waktu pengembangan selama 6,4 bulan, sedangkan skenario II hanya membutuhkan waktu pengembangan selama 3,7 bulan. Berdasarkan hasil ini dan batasan dari klien yang hanya memberikan waktu 6 bulan untuk pengembangan, Penulis memberikan rekomendasi kepada pihak manajemen untuk menggunakan skenario kedua dalam mengerjakan proyek ini.

EXECUTIVE SUMMARY

Most of Information Technology (IT) projects around the world face three main problems: late delivery, over budget and out of scope. This issue forces professionals in IT industry to redefine principles used in project management in IT industry.

One of the important phases of IT project management is estimation of project size. The estimation process needs to be done in order to justify that the organisation will get more value compare to resources used to develop or to acquire the new information system. The result will be used to decide the project should be continued or rejected by the executives of the organisation. This effort is essential for the organisation in order to neglect big investment that will bring no value for the organisation in the future.

From the developer's point of view, the estimation process is used not only to define the project size, but it can also be used to define the pricing of the project, time needed to finish the project and as a guideline for the management to make human resources allocation planning for every current projects the organisation owns. PT Imani Prima, where the author works, currently gets a project in developing a management information system for PT Tawang Swasti Rawikara Holding Company whose office at South Jakarta. This purpose of this research is to help the developer team to define the pricing and time needed to finish the project. This research uses *Use Case Points* as a project size estimation method.

This research will use UML 2.0 use case as a tool to analyze the system. Use case is a powerful tool and also because the use case points will estimate the project

based on the defined use case of the system. Based on author's experiences in developing some information systems, *use case* offers better analysist and design in order to understand how the information system works, what features are performed by the system, who will interact with the system and how the new information system interacts and communicates with the other existing systems.

The estimation process uses two scenarios. Scenario I, PT Imani Prima will assign part timer or freelances whose no experiences in developing complex and integrated information system. The scenario II, PT Imani Prima will assign its employees whose quite experiences in developing information system and familiar to use Microsoft Visual Basic 6.0, SQL Server Express and Rational Rose 2002 as tools will be used in the development. The estimation results will be added to 10% risk factor. According to the results, scenario I will take 6.4 months for the development and the scenario II will take 3.7 months. Based on the results and the customer requirement (6 months for development), this research gives recommendation to management to use the scenario II in this project.

DAFTAR ISI

KATA PENGANTAR i

RINGKASAN EKSEKUTIF iii

EXECUTIVE SUMMARY v

DAFTAR ISI vii

DAFTAR LAMPIRAN x

DAFTAR GAMBAR xi

DAFTAR TABEL xii

BAB I PENDAHULUAN 1

1.1 Latar Belakang 1

1.2 Perumusan Masalah 2

1.3 Tujuan dan Manfaat Penelitian 3

1.4 Lingkup Penelitian 4

1.5 Metodologi Penelitian 5

1.6 Sistematika Pembahasan 6

BAB II LANDASAN TEORI 8

2.1 Systems Development Life Cycle 8

2.1.1 Perencanaan 8

2.1.2 Analisis 10

2.1.3 Desain 10

2.1.4 Implementasi 10

2.2 Analisis Sistem Informasi Berbasis Objek 11

2.3 *Unified Modelling Languange* (UML) 2.0 14

2.3.1 Diagram yang digunakan 15

2.3.1.1 Diagram Struktur 15

2.3.1.2 Diagram Permodelan Tingkah Laku 17

2.4 Estimasi Ukuran Proyek 19

2.4.1 *Planning Phase Approach* 19

2.4.2 Function Point 18

2.4.2.1 Estimasi ukuran sistem 19

2.4.2.2 Estimasi *effort* yang dibutuhkan 22

2.4.2.3 Estimasi jadual 24

2.4.3 *Use Case Point* 24

2.4.3.1 Pembobotan terhadap aktor 25

2.4.3.2 Pembobotan terhadap *use case* 26

2.4.3.3 Pembobotan terhadap *technical complexity* dan *environmental complexity* 26

2.4.3.4 Penentuan *Person Hour Multiplier* (PHM) 28

2.4.3.5 Penghitungan jumlah jam kerja yang dibutuhkan 29

2.4.3.6 Lembar kerja estimasi *Use Case Point* 29

BAB III OBJEK PENELITIAN **31**

3.1 Objek Penelitian 31

3.2 Kebutuhan akan sistem yang akan dibangun 31

3.2.1 Kebutuhan Sistem Informasi Pembukuan 32

3.2.2 Kebutuhan Sistem Informasi Sumber Daya Manusia 34

BAB IV ANALISIS DAN ESTIMASI SISTEM **35**

4.1 Analisis Sistem

4.1.1 Spesifikasi Kebutuhan 35

4.1.1.1 Kebutuhan Fungsional 35

4.1.1.2 Kebutuhan Non Fungsional 36

4.1.2 Permodelan Fungsional 36

4.1.3 Daftar Paket 37

4.1.4 Daftar Aktor 38

4.1.5 Daftar *Use Case* 39

4.1.6 Diagram Aktifitas 39

4.2 Estimasi ukuran proyek menggunakan *Use Case Points* 39

4.2.1 Pengukuran *Unadjusted Use Case Points* 39

4.2.1.1 *Simple Use Case* 40

4.2.1.2 *Average Use Case* 41

4.2.1.3 *Complex Use Case* 42

4.2.2 Pengukuran kompleksitas teknis pengembangan 45

4.2.3 Pengukuran kompleksitas lingkungan pengembangan 47

4.2.4 Jumlah jam kerja orang 48

BAB V PENUTUP **51**

5.1 Kesimpulan 51

5.2 Saran 53

DAFTAR PUSTAKA **55**

LAMPIRAN

DAFTAR LAMPIRAN

Lampiran A : Use Case Description

Lampiran B : Use Case Diagram

Lampiran C : Diagram Aktifitas

DAFTAR GAMBAR

Gambar 2.1 : Urutan aktifitas pada *Function Point* 19

DAFTAR TABEL

Tabel 2.1	: Implementasi persentasi standar industri pada estimasi system informasi	18
Tabel 2.2	: Komponen pembentuk sistem informasi	20
Tabel 2.3	: Daftar bobot komponen	20
Tabel 2.4	: Daftar *Unadjusted Function Point*	20
Tabel 2.5	: Kompleksitas Keseluruhan Proses	21
Tabel 2.6	: Tabel konversi *Function Points* menjadi jumlah baris kode program	21
Tabel 2.7	: Daftar item *technical complexity*	25
Tabel 2.8	: Daftar item *environmental factor*	26
Tabel 2.9	: Lembar kerja *Unadjusted Actor Weight*	28
Tabel 2.10	: Lembar kerja *Unadjusted Use Case Weight*	28
Tabel 2.11	: Lembar kerja faktor tingkat kompleksitas teknis	28
Tabel 2.12	: Lembar kerja faktor lingkungan	29
Tabel 3.1	: Contoh Struktur Rekening PT Tawang Swasti Rawikara	32
Tabel 4.1	: Daftar Nama Paket	37
Tabel 4.2	: Daftar aktor dan perannya	38
Tabel 4.3	: Daftar *Unadjusted Actor Weight* hasil analisis	40
Tabel 4.4	: Daftar *Unadjusted Use Case Weight* hasil analisis	44
Tabel 4.5	: Daftar faktor tingkat kompleksitas teknis hasil analisis	46
Tabel 4.6	: Daftar faktor lingkungan hasil analisis skenario I	48
Tabel 4.7	: Daftar faktor lingkungan hasil analisis skenario II	50

BAB I

PENDAHULUAN

1.1 Latar Belakang

Sebagian besar proyek Teknologi Informasi (TI) yang dikerjakan di seluruh dunia menghadapi tiga masalah penting, yaitu: lewat tenggat waktu (*Late Delivery*), melebihi anggaran yang sudah ditetapkan (*Over Budget*), serta hasil akhir tidak memenuhi seluruh kebutuhan klien (*out of scope*) (Schwalbe, 2006). Tiga permasalahan ini memaksa banyak CEO di perusahaan yang bergerak di bidang TI di seluruh dunia untuk melakukan pembenahan besar-besaran. Banyak revisi dilakukan terhadap metode yang umum digunakan dalam *Software Engineering*, seperti metode *Waterfall*, *Prototyping* dan *Object Oriented Programming*.

Pengembangan sistem informasi berbasis objek merupakan salah satu *Software Development Life Cycle* (SDLC) yang banyak dipakai oleh para pengembang saat ini. Menggabungkan paradigma *data centric* dan *process centric*, proses analisis dan disain pada pemrograman berbasis objek menawarkan pendekatan yang dapat lebih memahami struktur dan tingkah laku sistem. Pengembang lebih mudah menentukan apa yang dilakukan oleh sistem, interaksi antara sistem dengan pengguna, interaksi antara sistem dengan sistem eksternal lainnya dan untuk lebih memahami interaksi antar komponen sistem.

Use case diagram adalah salah satu alat bantu yang digunakan pada pemrograman berorientasi objek untuk memudahkan tim pengembang melakukan analisis terhadap aktifitas apa saja yang mungkin terjadi dalam suatu sistem serta siapa yang melakukan atau mengalami aktifitas tersebut. Dengan bantuan *use case diagram*, tim pengembang dapat memfokuskan

perhatian kepada aktifitas-aktifitas penting yang terjadi di dalam sebuah sistem informasi. Aktifitas-aktifitas yang terdefinisi inilah yang kemudian dikenal dengan sebutan *use case*.

Fase perancangan dalam rangkaian proses SDLC memegang peranan yang sangat penting dalam pengembangan sistem informasi. Pada fase inilah estimasi terhadap besar suatu proyek pengembangan sistem informasi dilakukan. Dengan melakukan estimasi, manajer proyek bisa memiliki gambaran awal tentang proyek, dan membantu merencanakan pengorganisasian anggota tim, menentukan anggaran dan yang lebih penting untuk menentukan apakah proyek pembangunan sistem informasi sebaiknya dilanjutkan atau tidak.

Saat ini, penulis sendiri bekerja di sebuah perusahaan yang bergerak di bidang TI dengan nama PT Imani Prima. Salah satu aktivitas yang akan dijalankan adalah pengembangan sistem informasi pembukuan dan sumber daya manusia pada PT Tawang Swasti Rawikara dengan memanfaatkan teknologi yang ada dan umum digunakan di pasar saat ini, *Microsoft based technology*. Melalui karya akhir ini, penulis ingin menerapkan salah satu metode estimasi ukuran proyek pada proyek pengembangan sistem informasi di PT Tawang Swasti Rawikara. Melalui estimasi yang dilakukan terhadap proyek ini, diharapkan proyek dapat berjalan lancar dan dapat memenuhi tiga kriteria proyek: tepat waktu, tepat anggaran dan memenuhi kebutuhan PT Tawang Swasti Rawikara sebagai pelanggan.

1.2 Perumusan Masalah

Kriteria *Operation Excellence* yang ingin dicapai adalah mampu mengatasi ketiga kendala yang sebelumnya disebutkan: *Late Delivery, Over Budget* dan *Out of Scope*. Dalam pembahasannya, penulis akan menekankan pada proses-proses yang dilakukan pada fase analisis terhadap proyek pengembangan sebuah sistem informasi. Setelah mendapatkan data

yang cukup terhadap perilaku sistem, karya akhir akan membahas metode estimasi terhadap sistem informasi yang akan dibangun.

Beberapa faktor penting yang perlu diperhatikan dalam mengestimasi ukuran proyek ini adalah sebagai berikut:

1. Pemilihan *Software Development Life Cycle* (SDLC) yang akan digunakan dalam pengembangan sistem informasi yang akan dibangun.
2. Pendefinisian daftar kebutuhan dari sistem informasi yang akan dibangun.
3. Pemilihan *tools* dan prosedur analisis terhadap sistem informasi yang akan dibangun.
4. Menciptakan batasan pembahasan agar hasil dapat lebih terukur dan signifikan.

1.3 Tujuan dan Manfaat Penelitian

Tujuan yang ingin dicapai penulis dalam karya akhir ini adalah sebagai berikut:

1. Melakukan kajian terhadap kebijakan-kebijakan yang sebelumnya telah dilakukan oleh perusahaan-perusahaan yang bergerak di bidang TI baik lokal maupun internasional untuk mencapai *Operations Excellence*.
2. Melakukan kajian terhadap metode ataupun model yang mungkin dapat diterapkan oleh PT Imani Prima untuk mencapai *Operations Exellence*.
3. Mendokumentasikan analisis sistem informasi pembukuan dan sumber daya manusia memanfaatkan *framework* SDLC yang terpilih.
4. Mendokumentasikan dokumen-dokumen yang terkait dalam manajemen proyek yang akan dibutuhkan pada fase perancangan dan implementasi sistem informasi pembukuan dan sumber daya manusia ini.

Adapun manfaat penelitian yang ingin dicapai oleh penulis adalah sebagai berikut:

1. Memberikan masukan bagi para pelaku bisnis dalam TI sebagai bahan pertimbangan dalam menjalankan dan mengembangkan usaha mereka di Indonesia.
2. Mengaplikasikan ilmu yang selama ini telah dipelajari oleh penulis di Magister Manajemen Universitas Indonesia (MM UI), khususnya di konsentrasi Manajemen Operasi.
3. Memberikan kontribusi bagi perkembangan ilmu pengetahuan manajemen di Indonesia.

1.4 Lingkup Penelitian

Penelitian dilakukan dalam lingkungan PT Imani Prima dengan memanfaatkan pengalaman karyawan PT Imani Prima serta semua data sekunder yang mungkin didapatkan selama penelitian ini dilakukan. Penelitian yang dilakukan hanya terbatas pada fase analisis dan proses estimasi terhadap sistem informasi yang akan dibangun dengan studi kasus Sistem Informasi Pembukuan dan Sumber Daya Manusia untuk PT Tawang Swasti Rawikara. Fase perancangan, implementasi, pengujian produk serta instalasi tidak dibahas pada karya akhir ini. Proses estimasi hanya menghitung jumlah jam kerja orang yang dibutuhkan untuk mengembangkan sebuah sistem informasi.

Penelitian menggunakan *use case* sebagai alat bantu dalam fase analisis. Notasi yang digunakan adalah yang terdefinisi pada UML versi 2.0. Walaupun *use case* umumnya digunakan dalam analisis untuk sistem berorientasi objek, fase implementasi sistem informasi manajemen ini akan menggunakan salah satu *tools* dalam *structured programming*, yaitu: Microsoft Visual Basic 6.0 dengan menggunakan basis data Microsoft SQL Server Express.

1.5 Metodologi Penelitian

Metodologi pembahasan yang dilakukan dalam proses penulisan karya akhir ini adalah:

1. Tahap persiapan

   Kegiatan yang dilakukan meliputi perencanaan pengumpulan data dan informasi, serta menentukan cara untuk mendapatkan data dan informasi berdasarkan tujuan yang ingin dicapai.

2. Penelitian kepustakaan

   Penelitian Kepustakaan dilakukan dengan membaca berbagai literatur dan referensi yang dimaksudkan untuk mendapatkan teori-teori yang relevan dengan instrumen analisis untuk membahas dan menganalisis masalah yang akan dibahas. Selain itu untuk mendapatkan data dan informasi tambahan, perlu dilakukan pencarian data sekunder dari berbagai sumber, seperti: *internet* maupun laporan-laporan yang dikeluarkan oleh perusahaan konsultan yang dapat dimanfaatkan untuk menunjang asumsi yang dipakai.

3. Pengolahan data awal

   Data dan informasi yang diperoleh harus diolah untuk mendapatkan data yang lebih relevan. Selain itu, tahap ini juga dapat membantu penulis untuk membatasi permasalahan dan memperoleh asumsi-asumsi awal yang dapat digunakan dalam penelitian ini.

4. Pemilihan metode dan model yang digunakan

   Tahap ini melakukan pemilihan terhadap model dan metode yang digunakan, metode pembobotan, dan metode penelitian terhadap aktifitas yang terdefinisi pada sistem informasi yang akan dibangun.

5. Analisis dan proses estimasi

   Tahap ini merupakan proses inti dari karya akhir. Analisis dilakukan terhadap data yang didapatkan terutama daftar kebutuhan terhadap sistem informasi yang akan dibangun. Hasil dari fase analisis akan menjadi bahan yang sangat berguna untuk proses estimasi ukuran sistem informasi yang akan dibangun. Hasil yang didapatkan dengan melakukan analisis dan estimasi diharapkan dapat menjadi pegangan dalam tahapan selanjutanya dalam SDLC terhadap sistem informasi yang akan dibangun.

6. Penulisan laporan

   Data dan informasi yang didapatkan dari karya akhir ini akan dibahas kembali untuk menyesuaikan dengan maksud dan tujuan penulisan serta untuk kemudian ditarik kesimpulan atas hasil penelitian serta jika memungkinkan untuk memberikan rekomendasi untuk perbaikan.

1.6 Sistematika Pembahasan

Karya akhir ini disusun dalam sistematika penulisan sebagai berikut:

BAB I. PENDAHULUAN

Bab ini membahas latar belakang penulisan, permasalahan, tujuan penelitian, ruang lingkup penelitian, metodologi penelitian serta sistematika penulisan karya akhir.

BAB II. LANDASAN TEORI

Bab ini berisi pembahasan ringkas mengenai konsep sistem informasi, SDLC, permodelan yang digunakan, serta metodologi estimasi yang digunakan.

BAB III. OBJEK PENELITIAN

Bab ini membahas identitas PT Tawang Swasti Rawikara sebagai perusahaan yang akan diteliti. Pembahasan juga mencakup kebutuhan PT Tawang Swasti Rawikara akan sistem informasi pembukuan dan sumber daya manusia yang akan dibangun.

BAB IV ANALISIS SISTEM

Bab ini membahas serangkaian proses analisis dalam menerjemahkan sederet kebutuhan menjadi sebuah sistem informasi pembukuan dan sumber daya manusia yang diinginkan. Pada bab ini juga dibahas proses estimasi ukuran sistem informasi pembukuan dan sistem informasi sumber daya manusia yang ingin dibangun.

BAB V KESIMPULAN DAN SARAN

Bab ini membahas kesimpulan yang berhasil didapatkan penulis serta saran-saran yang mungkin dapat diberikan sebagai sumbangsih penulis terhadap kemajuan ilmu pengetahuan.

BAB II
LANDASAN TEORI

2.1 Systems Development Life Cycle

Systems Development Life Cycle (SDLC) adalah sebuah tahapan proses yang perlu dilakukan dalam mengembangkan sebuah sistem informasi di dalam suatu organisasi. Tahapan proses ini perlu dijalankan untuk memahami kebutuhan bisnis yang akan dipenuhi oleh sistem yang akan dibangun, mendesain, membangun dan mengintegrasikan sistem yang baru kepada lingkungan organisasi. Tujuan utama SDLC adalah sistem yang dibangun dapat memberikan nilai lebih (*value*) kepada organisasi dan memenuhi batasan-batasan yang ditetapkan dalam organisasi dalam proses pengembangannya. Batasan-batasan tersebut dapat berupa lingkup fungsionalitas sistem serta sumber daya organisasi yang digunakan untuk membangun sistem yang baru.

SDLC terdiri atas beberapa fase dan diimplementasikan dalam berbagai cara yang berbeda. Meskipun demikian, ada empat fase fundamental yang dimiliki oleh tiap SDLC, yaitu:

2.1.1 Perencanaan

Fase Perencanaan adalah sebuah fase untuk memahami mengapa sebuah sistem informasi harus dibangun dan bagaimana mengkoordinasikan anggota tim untuk membangun sistem tersebut. Fase Perencanaan terbagi lagi menjadi dua bagian:

a. Inisiasi Proyek

   Inisiasi Proyek dilakukan sebelum sistem dibangun. Tahapan ini dimulai dengan mendefinisikan manfaat bisnis (*business value*) apa yang dibutuhkan oleh organisasi atau unit tertentu sebagai justifikasi bahwa sistem informasi harus dibangun. Unit

organisasi yang mendefinisikan atau menginginkan implementasi manfaat bisnis disebut dengan Sponsor Proyek.

Setelah manfaat bisnis dari proyek pembangunan sistem informasi yang baru berhasil didefinisikan, tahapan berikutnya yang perlu dilakukan adalah melakukan suatu analisis fisibilitas (*feasibility analysis*) untuk menentukan apakah manfaat bisnis yang didapatkan organisasi lebih besar dibandingkan sumber daya yang harus dikeluarkan oleh organisasi untuk membangun sistem informasi tersebut. Analisis fisibilitas mengkaji tiga hal penting: fisibilitas teknis (*technical feasibility*), fisibilitas ekonomis (*economic feasibility*) dan fisibilitas organisasi (*organizational feasibility*).

Hasil dari kedua proses di atas (*deliveries*) akan dipresentasikan kepada sebuah *steering committee* yang akan menentukan apakah sebuah proyek pembangunan sistem informasi layak untuk dilanjutkan.

b. Manajemen Proyek

Manajemen proyek adalah serangkaian proses perencanaan dan pengawasan terhadap proses pengembangan sebuah sistem agar proses tersebut memenuhi tiga kriteria, yaitu: tepat waktu, sesuai dengan anggaran dan memenuhi kebutuhan klien atau pengguna. (Dennis et.al, 2005: 85). UML 2.0 mendefinisikan tahapan Manajemen Proyek terdiri atas 4 bagian:

- mengidentifikasikan ukuran proyek, bertujuan untuk mengukur berapa besar ukuran dari sistem dan membutuhkan berapa banyak *resources* untuk membangunnya. Untuk karya akhir ini, penulis akan menggunakan metode *use case points*.
- membuat dan mengontrol rencana kerja, bertujuan untuk mendefinisikan *job role* apa saja yang dibutuhkan dalam pembangunan sistem ini dan apa saja aktivitas-aktivitas penting yang dominan dalam proyek pembangunan sistem informasi

manajemen retail ini. Hasil dari tahapan ini bisa diterjemahkan dalam *Work Breakdown Structure* (WBS) untuk melihat keseluruhan rangkaian aktivitas yang akan dilakukan dalam pembangunan sistem informasi ini. WBS dapat didefinisikan dengan bantuan *Gantt Chart*.

- merencanakan penyusunan staff, bertujuan untuk mendefinisikan kebutuhan keahlian (*skill*), pengetahuan (*knowledge*) serta *update* teknologi yang mapan disesuaikan dengan *job role* yang sebelumnya telah terdefinisi pada Gantt Chart.
- mengkoordinasikan aktifitas-aktifitas proyek, bertujuan untuk mengawasi pelaksanaan proyek agar tetap memenuhi kriteria: tepat waktu, masih dalam anggaran, dan memenuhi kebutuhan klien. Untuk mencapai tujuan ini, maka perlu didefinisikan suatu standarisasi tertentu agar proyek dapat dipantau apakah masih *on track* atau perlu diberikan usaha tambahan agar keseluruhan proyek dapat sesuai dengan batasan yang telah ditentukan.

2.1.2 Analisis

Fase analisis bertugas untuk menjawab pertanyaan siapa yang akan memanfaatkan sistem, apa yang akan dilakukan oleh sistem, dan kapan sistem akan digunakan. Selama fase ini, tim pengembang akan melakukan investigasi tentang sistem yang sudah berjalan saat ini, mengidentifikasikan kemungkinan-kemungkinan pengembangan yang dapat dilakukan, dan mengembangkan sebuah konsep untuk sebuah sistem baru.

Fase ini memiliki tiga langkah, yaitu:

a. Strategi analisis, diperlukan untuk membimbing anggota tim pengembang. Termasuk di dalamnya analisis terhadap sistem saat ini dan permasalahannya serta mencari cara untuk mendisain sistem baru.

b. Mengumpulkan kebutuhan, melalui wawancara atau *questionnaire*. Analisis dari informasi yang didapatkan diselaraskan dengan masukan dari sponsor proyek dan pihak lain yang dapat mengarahkan kepada konsep pengembangan sebuah sistem baru. Konsep ini yang kemudian digunakan sebagai dasar untuk mengembangkan seperangkat permodelan analisis bisnis yang mendeskripsikan bagaimana sistem bekerja apabila sistem jadi diimplementasikan.

c. Membuat sebuah proposal, hasil analisis, konsep dan model dibuat menjadi sebuah dokumen yang disebut dengan *system proposal* yang kemudian akan dikirimkan kepada sponsor proyek dan para pengambil keputusan yang lain untuk diputuskan apakah proyek layak untuk dilanjutkan atau tidak.

2.1.3 Desain

Fase desain yang memutuskan bagaimana sistem bekerja dan diterjemahkan menjadi sekumpulan dokumen yang disebut dengan *system spesification*. *System spesification* terdiri atas desain arsitektur sistem, desain antarmuka, spesifikasi basis data dan *file*, dan desain program.

2.1.4 Implementasi

Fase implementasi merupakan fase akhir dari SDLC. Pada fase inilah, sistem akhirnya dibangun. Biasanya fase ini yang menggunakan waktu paling lama dan menghabiskan biaya paling besar. Fase ini terdiri atas 3 bagian penting:

a. Konstruksi sistem, dimana sistem dibangun dan dites untuk memastikan bahwa sistem yang dibangun sesuai dengan spesifikasi kebutuhan yang telah didefinisikan sebelumnya oleh klien.

b. Instalasi sistem, dimana sistem yang lama diganti dengan yang baru. Berbagai pendekatan dapat digunakan pada tahapan ini mulai dari yang paling radikal adalah

direct cutover (sistem lama langsung digantikan oleh sistem baru), *parallel conversion* (sistem yang lama dan yang baru sama-sama berjalan dan pelan-pelan fungsi dari sistem yang lama akan digantikan oleh sistem yang baru), *phased conversion* (sistem yang baru diimplementasikan pada suatu bagian dari organisasi sebelum diimplementasikan pada bagian lain dari organisasi). Satu hal lagi yang penting dari fase ini adalah *training*, dimana pengguna diajarkan bagaimana memanfaatkan sistem yang baru untuk membantu pekerjaannya.

c. Pemeliharaan sistem atau mekanisme *support*, termasuk mungkin melakukan perubahan besar atau kecil pada sistem setelah sistem diinstalasi di lingkungan klien.

2.2 Analisis Sistem Informasi Berbasis Objek

Sistem Berorientasi Objek merupakan salah satu SDLC yang menterjemahkan struktur dan tingkah laku suatu sistem informasi menjadi modul-modul kecil yang menggambarkan data dan proses apa saja yang dilakukan dan terjadi di dalam sistem. Modul kecil ini dikenal dengan konsep objek. SDLC berbasis objek melakukan analisis terhadap sistem dengan melakukan analisis terhadap objek apa saja yang mungkin terlibat dalam suatu sistem informasi dan analisis terhadap interaksi yang terjadi antar objek di dalam sistem. SDLC berbasis objek memiliki karakteristik dasar yang terdiri atas: *classes*, *objects*, *methods*, *encapsulation*, *information hiding*, *inheritance*, *polymorphism* dan *dynamic binding*.

2.3 *Unified Modelling Language* (UML) 2.0

Awalnya konsep objek sangat populer tapi diimplementasikan dalam berbagai cara dan oleh berbagai pengembang. Tiap pengembang memiliki metodologi dan notasi-notasi tertentu, seperti: Booch, Coad, Moses, OMT, OOSE, dan SOMA. Pada tahun 1995, Rational

Software mengajak tiga besar dalam industri objek untuk mengembangkan suatu pendekatan baru dalam metode pengembangan aplikasi berbasis objek. Bersama dengan Grady Booch, Ivar Jacobson dan James Rumbaugh, Rational Software mengembangkan suatu konsep baru yang kemudian disebut dengan *Unified Modelling Language*. (Dennis et.al, 2005)

Objektif yang ingin dicapai oleh UML adalah menyediakan sebuah bahasa yang sama dalam terminologi pemrograman berbasis objek dan dan teknik diagram yang cukup kaya untuk memodelkan proyek pengembangan sistem apapun mulai tahap analisis sampai implementasi. Pada November 1997, *Object Management Group* (OMG) secara formal menetapkan UML sebagai standar untuk semua pengembang berbasis objek. Saat ini UML telah mencapai versi 2.0 yang ditetapkan pada 2003. (Dennis et.al, 2005)

2.3.1 Diagram yang digunakan

UML versi 2.0 mendefinisikan 14 diagram untuk memodelkan suatu sistem informasi. Diagram-diagram ini terbagi menjadi dua grup besar: untuk memodelkan struktur sistem dan untuk memodelkan tingkah laku sistem.

2.3.1.1 Diagram Struktur

Diagram struktur menyajikan cara merepresentasikan data dan hubungan statik yang ada di sistem informasi. Berikut ini adalah jenis-jenis diagram yang merupakan bagian dari diagram struktur:

1. Diagram *Class*

   Diagram *Class* merepresentasikan benda-benda, ide atau konsep yang ada pada aplikasi serta hubungan antara *class* yang terdefinisi. Diagram ini juga mendefinisikan apa yang menjadi sifat dari *class* serta apa saja yang dapat dilakukan oleh *class* yang bersangkutan di dalam sistem.

2. Diagram *Object*

Diagram *Object* merupakan diagram yang menggambarkan perwujudan dari *class* yang sebelumnya telah terdefinisi pada diagram *class*. Dengan menggambarkan *object* dari perwujudan *class,* diharapkan pengembang dapat lebih memahami karakteristik dari *object* pada saat dibangkitkan dan menjalankan fungsinya di dalam aplikasi.

3. Diagram *Package*

Fungsi utama dari Diagram *Package* adalah mengumpulkan elemen-elemen UML diagram ke dalam elemen lain yang memiliki satu level lebih tinggi. Diagram *Package* adalah Diagram *class* yang hanya menggambarkan *package* dan hubungan yang dimiliki antar *package*.

4. Diagram *Deployment*

Diagram ini digunakan untuk merepresentasikan hubungan antara komponen perangkat keras yang digunakan oleh infrastruktur fisik pada suatu sistem informasi. Sebagai contoh, pada disain sistem informasi terdistribusi yang menggunakan *Wide Area Network*, diagram ini dapat digunakan untuk menggambarkan komunikasi antar node jaringan. Diagram ini juga dapat digunakan untuk merepresentasikan komponen perangkat lunak dan bagaimana mereka berjalan di atas arsitektur fisik atau infrastruktur sebuah sistem informasi. Pada kasus ini, diagram ini merepresentasikan lingkungan eksekusi suatu perangkat lunak.

5. Diagram *Component*

Diagram ini digunakan untuk memodelkan hubungan fisik antara modul fisik kode program. Apabila diagram ini dikombinasikan dengan diagram *deployment*, dapat menggambarkan *class* atau *package of classes* yang ada di node jaringan dan mana yang ada di server.

6. Diagram *Composite Structure*

Diagram ini adalah diagram baru yang disediakan oleh UML memodelkan hubungan antara bagian-bagian dalam kelas. Diagram ini digunakan apabila struktur internal suatu *class* sangat rumit. Diagram ini memiliki 3 *subclass* penting, yaitu: *header, footer* dan detil baris program. Diagram ini juga dapat digunakan untuk memodelkan struktur internal dari sebuah komponen pada sistem berbasis komponen.

2.3.1.2 Diagram Permodelan Tingkah Laku

Diagram Permodelan Tingkah Laku menggambarkan hubungan dinamis antar instansiasi atau *object* yang merepresentasikan sebuah sistem informasi. Diagram ini juga dapat memodelkan kelakuan dinamis objek individual selama masa hidup mereka dalam suatu sistem informasi. Berikut ini adalah diagram-diagram yang merupakan bagian dari Diagram Permodelan Tingkah Laku:

1. Diagram Aktifitas

   Diagram ini dipakai untuk menjelaskan proses dalam suatu sistem informasi. Diagram ini juga dapat digunakan untuk memodelkan *workflow*, *individual use cases*, lojik keputusan yang terkandung di dalam suatu metode individual. Diagram ini juga dapat digunakan untuk memodelkan proses paralel bagian yang membantu terwujudnya sistem informasi.

2. Diagram *Sequence*

   Diagram ini menggambarkan interaksi dinamis antara *object* dalam sistem informasi. Diagram ini menekankan pada urutan waktu aktifitas dari sekumpulan *object* yang berkolaborasi dan dapat digunakan untuk menggambarkan interaksi lojik dan fisik antar *object*.

3. Diagram Komunikasi

Diagram ini menawarkan alternatif lain dalam menggambarkan interaksi dinamis antar *object* dalam sistem informasi. Diagram ini berfokus pada seperangkat pesan yang saling dikirimkan oleh para *object* yang berkolaborasi.

4. Diagram *Interaction Overview*

Diagram ini memberikan *overview* kepada aliran proses kontrol. Diagram ini dapat dianggap sebagai Diagram Aktifitas dengan menambahkan beberapa *sequence fragments* dari Diagram *sequence*. Keuntungan utama dari diagram ini adalah pengembang dapat menambahkan aliran *sequence* alternatif dengan mudah.

5. Diagram Pewaktuan

Diagram ini menggambarkan interaksi antara *objects* dalam sumbu axis waktu. Tujuan utama dari diagram ini adalah untuk menunjukkan perubahan *state* suatu *object* dalam merespon suatu *event* dalam urutan waktu. Diagram ini sangat berguna dalam mengembangkan sistem *real time* atau *embedded.*

6. *Behavioral State Machine*

Diagram ini menyediakan sebuah metode dalam memodelkan *states* yang berbeda atau sekumpulan *values* yang dapat diturunkan dari suatu *class* sepanjang hidupnya. Misalnya, suatu *class* pasien dapat berubah dari *Pasien Baru* menjadi *Pasien Sekarang* untuk kemudian akhirnya berubah menjadi *Pasien Sebelumnya.*

7. *Protocol State Machine*

Diagram ini mendukung analis dalam mendisain ketergantungan antar elemen pada antarmuka *class*. Sebagai contoh, pengguna harus membuka *file* atau basis data sebelum melakukan *query* atau update terhadapnya. Berbeda dengan *Behavioral State Machine*, *Protocol State Machine* dapat diasosiasikan dengan *port* komponen atau antarmuka *class*.

8. Diagram *Use Case*

Diagram ini digunakan untuk memodelkan interaksi antara sebuah sistem informasi dengan lingkungannya. Lingkungan yang dimaksud dapat berupa pengguna ataupun sistem eksternal lainnya yang berinterkasi dengan sistem informasi. Fungsi utama dari diagram ini adalah sebagai sarana untuk mendokumentasikan dan memahami kebutuhan dibangunnya sistem informasi.

2.4 Estimasi Ukuran Proyek

Estimasi adalah sebuah proses yang membuat proyeksi nilai terhadap waktu dan usaha yang akan dikeluarkan dalam proyek pembangunan sebuah sistem informasi. Standarisasi nilai yang digunakan dalam estimasi diambil dari berbagai sumber, seperti: berasal dari metodologi yang digunakan, berasal dari proyek sebelumnya yang menggunakan *task* dan teknologi yang sama, atau disediakan oleh pengembang yang berpengalaman. Di bawah ini akan dibahas beberapa metode estimasi ukuran proyek.

2.4.1 *Planning Phase Approach*

Pendekatan ini menetapkan waktu yang digunakan pada fase perencanaan (*Planning phase*) sebagai acuan dalam menentukan waktu yang akan digunakan pada fase-fase selanjutnya dalam SDLC. Ide dasar dari pendekatan ini adalah "proyek sederhana memerlukan waktu yang sedikit untuk perencanaan dan sebuah proyek yang kompleks akan memerlukan perencanaan yang lebih matang dan waktu yang lebih lama." (Dennis et.al, 2005). Berdasarkan ide dasar ini, cukup beralasan untuk menggunakan waktu yang digunakan pada fase perencanaan untuk mengestimasi kebutuhan waktu untuk seluruh proyek.

Metode ini membutuhkan dua jenis data: data waktu yang dibutuhkan pada fase perencanaan, data yang kedua adalah persentase standar industri (atau persentase berdasarkan

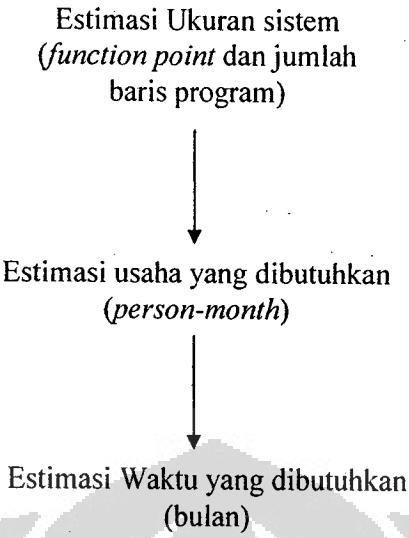
pengalaman organisasi) untuk menghitung estimasi fase lain pada SDLC. Standar industri menganjurkan bahwa sistem aplikasi bisnis pada umumnya menghabiskan 15% usaha pada fase perencanaan, 20% pada fase analisis, 35% pada fase desain, dan 30% pada fase implementasi. Untuk lebih jelasnya, dapat dilihat pada tabel 2.1

Tabel 2.1 Implementasi persentase standar industri pada estimasi sistem informasi

	Perencanaan (*person-month*)	Analisis (*person-month*)	Desain (*person-month*)	Implementasi (*person-month*)
Standar industri untuk aplikasi bisnis biasa	15%	20%	35%	30%
Estimasi berdasarkan data fase analisis	Actual : 4	Estimasi: 5.33	Estimasi: 9.33	Estimasi: 8

2.4.2 *Function Point*

Konsep ini dikembangkan oleh Allen Albrecht dari IBM pada tahun 1979. *Function Point* adalah pengukuran ukuran program yang berdasarkan pada jumlah sistem dan kompleksitas *input*, *ouput*, *queries*, *files* dan antarmuka program. Konsep ini terbagi menjadi 3 langkah penting, yaitu: manajer proyek mengestimasi ukuran proyek dalam jumlah baris program yang dibutuhkan oleh sistem baru, mengonversi estimasi ukuran tadi menjadi satuan *person-month*, untuk kemudian dikonversi menjadi estimasi jadual dalam jumlah bulan dari awal sampai akhir masa pengembangan sistem.



Gambar 2.1 Urutan aktifitas pada *Function Point*

2.4.2.1 Estimasi ukuran sistem

Komponen-komponen pembentuk sistem informasi didata dalam sebuah lembar kerja untuk merepresentasikan elemen dominan dari sistem. Sebagai contoh, layer masukan data adalah jenis *input*, laporan adalah jenis *output*, dan *query* ke basis data adalah jenis dari *query* (Lihat gambar 2.2). Manajer proyek menyimpan jumlah total setiap komponen yang menjadi bagian dari sistem untuk kemudian digolongkan kembali menjadi komponen yang memiliki kompleksitas rendah, sedang dan tinggi. Tiap golongan dan komponen memiliki bobot yang berbeda. Hasil dari perkalian antara tabel bobot dan tabel komponen menghasilkan suatu nilai yang kemudian ditotalkan dan disebut sebagai *Total Unadjusted Function Point* (TUFP).

Tabel 2.2 Komponen pembentuk sistem informasi

Deskripsi		Kompleksitas		
	Jumlah	Rendah	Sedang	Tinggi
Input	6	3	2	1
Output	19	4	10	5
Queries	10	7	0	3
Files	15	0	15	0
Program Interfaces	3	1	0	2

Tabel 2.3 Daftar bobot komponen

Deskripsi	Kompleksitas		
	Rendah	Sedang	Tinggi
Input	3	4	6
Output	4	5	7
Queries	3	4	6
Files	7	10	15
Program Interfaces	5	7	10

Tabel 2.4 Daftar *Unadjusted Function Point*

Deskripsi	*Unadjusted Function Point* (UFP)
Input	23
Output	101
Queries	39
Files	150
Program Interfaces	25
Total UFP (TUFP)	**338**

Kompleksitas keseluruhan sistem lebih besar dibandingkan jumlah dari keseluruhan komponen pembentuknya. *Familiarity* tim proyek terhadap area bisnis dan teknologi yang akan digunakan untuk diimplementasikan pada proyek juga berpengaruh cukup besar terhadap kompleksitas proyek. Nilai bobot untuk kompleksitas keseluruhan sistem berkisar antara 0-3. "0" berarti tidak mempengaruhi kompleksitas pemrosesan, "3" berarti daftar komponen berpengaruh sangat besar terhadap kompleksitas pemrosesan. Daftar item pembentuk kompleksitas keseluruhan sistem dapat dilihat pada table di bawah ini.

Tabel 2.5 Kompleksitas Keseluruhan Proses

Nomor	Deskripsi	Bobot
1	Data communication	3
2	Heavy use configuration	0
3	Transaction rate	0
4	End user efficiency	0
5	Complex processing	0
6	Installation ease	0
7	Multiple sites	0
8	Performance	0
9	Distributed functions	2
10	Online data entry	2
11	Online update	0
12	Reusability	0
13	Operational ease	0
14	Extensibility	0
	Total Processing Complexity (PC)	7

Berdasarkan table di atas dapat dihitung *Adjusted Processing Complexity* (APC) yang dimiliki oleh sistem informasi yang akan dibangun: 0,65 + (0,01 x 7) = **0,72**

Langkah berikutnya adalah menghitung *Total Adjusted Function Points* (TAFP) yang merupakan perkalian dari APC dan TUFP. Sehingga TAFP = 0.72 x 338 = **243**

Langkah terakhir adalah menghitung jumlah baris kode program berdasarkan TAFP yang baru didapatkan. Di bawah ini adalah tabel perkiraan jumlah baris kode program untuk setiap *function point* yang terdefinisi. Data diambil dari Capers Jones, Software Productivity Research, http://www.spr.com

Tabel 2.6 Tabel konversi *Function Points* menjadi jumlah baris kode program

Language	Approximate Number of Lines of Codes per Function Point
	130
C	110
COBOL	55
Java	50
C++	50
Turbo Pascal	50
Visual Basic	30
Power Builder	15

Language	Approximate Number of Lines of Codes per Function Point
HTML	15
Packages (Access, Excel)	10-40

Menggunakan nilai TAFP=243, maka apabila pengembang ingin menggunakan bahasa pemrograman Visual Basic untuk mengembangkan aplikasi, maka jumlah baris program yang akan dihasilkan adalah sebesar = 243 x 30 = 7.290 baris program.

2.4.2.2 Estimasi *effort* yang dibutuhkan

Berdasarkan data jumlah baris program di atas, maka dapat ditentukan besarnya usaha dalam satuan *person-month* untuk mengembangkan aplikasi. Metode untuk menghitung *efforts* adalah dengan mengalikan jumlah baris program dengan *production rates* (berapa besar pekerjaan dapat diselesaikan oleh seseorang dalam satu waktu tertentu). Banyak metode yang dapat digunakan untuk menghitung rasio produksi aplikasi. Salah satu yang paling terkenal adalah COCOMO yang diperkenalkan oleh Barry W. Boehm. Saat ini banyak versi permodelan COCOMO yang dibuat berdasarkan kompleksitas aplikasi, ukuran sistem, pengalaman pengembang, dan tipe aplikasi yang sedang dikembangkan. Misalnya untuk proyek aplikasi ukuran kecil ke moderat, permodelan biasanya menggunakan

Effort (dalam *person-month*) = 1,4 x seribu baris program.

Untuk kasus di atas, maka *effort* yang dibutuhkan adalah = 1,4 x 7,29 = 10,206 bulan.

2.4.2.3 Estimasi jadual

Setelah *effort* berhasil ditetapkan, maka jadwal optimal dari proyek dapat diestimasi. Data historis dapat digunakan sebagai acuan atau bisa menggunakan salah satu rumus yang telah ditetapkan oleh para ahli:

$$\textit{Scheduled time}\ (\textit{months}) = 3{,}0 \times \textit{person months}^{1/3}$$

Persamaan di atas banyak digunakan untuk melakukan estimasi. Menggunakan data *effort* selama 10,206 bulan, persamaan di atas menghasilkan 6,507 bulan. Hasil di atas memiliki arti bahwa proyek pengembangan sistem informasi seharusnya dijadualkan akan menggunakan waktu kurang lebih 6,5 bulan. Waktu ini adalah hanya untuk fase analisis, desain dan implementasi, tidak termasuk perancangan.

2.4.3 *Use Case Point*

Analisis *Use Case Point* (UCP) pertama kali dikembangkan oleh Gustav Karner pada tahun 1993 dan diperbaiki oleh Schneider dan Winter pada tahun 1998. Teknik ini melakukan estimasi terhadap proyek yang menggunakan permodelan *use case* dalam pengembangan aplikasi perangkat lunaknya.

UCP mencoba menggabungkan proses estimasi yang dipakai pada *Function Point* dan menerapkannya untuk melakukan estimasi terhadap pemodelan *Object Oriented* yang menggunakan pemodelan *use case*. Sebelum melakukan estimasi, pengembang aplikasi perangkat lunak harus terlebih dahulu mendefinisikan *use case* yang akan dibangun serta membuat diagramnya. UCP menggunakan diagram dan keterangan untuk tiap *use case* yang sebelumnya didefinisikan untuk menentukan bobot proyek yang akan dikerjakan.

Dalam mengestimasi suatu proyek, ada 6 (enam) proses utama yang perlu dilakukan secara berurutan. Keenam proses tersebut adalah:

1. Pembobotan terhadap *actor* yang terlibat dalam sistem yang akan dibangun
2. Pembobotan terhadap *use case* yang ada dalam sistem
3. Pembobotan terhadap *technical complexity*
4. Pembobotan terhadap *environmental*
5. Penentuan *person hour multiplier* (PHM)
6. Menentukan jumlah jam kerja yang diperlukan

2.4.3.1 Pembobotan terhadap aktor

Aktor, yang terdaftar pada diagram *use case* yang telah dibuat, diklasifikasikan ke dalam 3 kategori: *simple*, *average*, *complex*. Penjelasan ketiganya ada di bawah ini:

a. *simple actor*: sistem terpisah yang berkomunikasi dengan sistem yang akan dibangun menggunakan sebuah *application programming interface* (API) yang telah terdefinisi dengan baik dan mudah digunakan. Aktor ini memiliki bobot 1 (satu).

b. *average actor*: sistem terpisah yang berkomunikasi dengan sistem yang akan dibangun melalui protokol komunikasi standar, seperti: TCP/IP, FTP, HTTP atau basis data eksternal yang dapat diakses mengunakan SQL standar. Aktor ini memiliki bobot 2 (dua).

c. *complex actor*: pengguna akhir yang berkomunikasi dengan sistem menggunakan *graphical user interface* (GUI). Aktor ini memiliki bobot 3 (tiga).

Penghitungan bobot total dilakukan dengan menjumlahkan total perkalian antara bobot dan jumlah aktor yang ada pada klasifikasinya masing-masing. Proses ini akan menghasilkan nilai bobot yang dikenal dengan *Unadjusted Actor Weight Total* (UAW).

2.4.3.2 Pembobotan terhadap *use case*

Proses pembobotan terhadap *use case* hampir sama dengan proses pembobotan terhadap aktor. Perbedaan terletak pada nilai bobot untuk tiap kelas serta kriteria yang ditetapkan untuk tiap klasifikasi. Penentuan kriteria klasifikasi berdasarkan jumlah transaksi unik yang harus dijelaskan oleh *use case* yang bersangkutan. Klasifikasi *use case* adalah sebagai berikut:

a. *simple use case* (1-3 transaksi – bobot 5)

b. *average use case* (4-7 transaksi – bobot 10)

c. *complex use case* (>7 transaksi – bobot 15)

Proses penghitungan sama dengan proses pembobotan terhadap actor. Nilai bobot akhir yang dihasilkan dikenal dengan *Unadjusted Use Case Weight Total* (UUCW).

2.4.3.3 Pembobotan terhadap *technical complexity* dan *environmental complexity*

Proses pembobotan terhadap *technical complexity* dan *environmental complexity* dimaksudkan untuk memperhitungkan kompleksitas aplikasi yang akan dibangun serta pengalaman yang dimiliki oleh para anggota tim pengembang. Kedua faktor di atas ikut mempengaruhi besarnya usaha yang harus dilakukan dalam mengembangkan sistem yang diinginkan sehingga hasil proses estimasi diharapkan dapat mendekati realisasinya pada saat proyek diimplementasikan.

Technical complexity memiliki 13 *item* yang perlu diperhatikan, sedangkan *environmental* hanya memiliki 8 *item* yang perlu diperhatikan. Semua *item* diberikan memiliki bobot yang telah terdefinisi. Manajer proyek harus memberikan sebuah nilai tertentu (0-5) untuk semua *item* dengan 0 (nol) berarti *item* tidak relevan terhadap sistem yang akan dibangun, dan 5 (lima) berarti *item* merupakan factor yang penting bagi sistem yang akan dibangun. Total dari proses perkalian antara bobot dan nilai yang diisi oleh manajer proyek menghasilkan dua buah nilai yang disebut dengan *Technical Factor Value* (TFaktor) dan *Environmental Factor Value* (EFaktor). Tabel di bawah ini mendefinisikan *item* yang diperhitungkan pada *technical complexity*.

Tabel 2.7 Daftar item *technical complexity*

Kode	Item	Deskripsi
T1	Sistem Terdistribusi	Apakah sistem akan dibangun menjadi sebuah sistem terdistribusi
T2	Waktu tanggap atau kebutuhan akan tingkat performansi lewatan (*troughput*)	Pentingnya response time
T3	Efisiensi user online	Tingkat efisiensi dari end user saat menggunakan sistem

Kode	Item	Deskripsi
T4	Proses internal yang kompleks	Kompleksitas proses internal sistem
T5	Tingkat *reusability* kode program	Pentingnya *code reuse*
T6	Mudah diinstall	Seberapa mudah proses instalasi sistem dilakukan
T7	Mudah digunakan	Seberapa penting sistem mudah digunakan
T8	Portabilitas	Seberapa penting sistem bisa diimplementasikan pada platform lain
T9	Mudah dilakukan perubahan	Apakah perawatan sistem sangat penting
T10	Konkurensi	Apakah sistem akan menangani proses yang parallel dan konkuren
T11	Kebutuhan tingkat keamanan khusus	Tingkat keamanan special yang dibutuhkan
T12	Akses langsung untuk pihak ketiga	Tingkat aksesibilitas sistem oleh pihak ketiga
T13	Butuh pelatihan khusus bagi pengguna	Apakah diperlukan pelatihan khusus bagi *end user* untuk dapat memanfaatkan sistem

Tabel 2.8 Daftar item *environmental factor*

Kode	Item	Deskripsi
E1	Familiar dengan proses pengembangan sistem yang digunakan	Tingkat pengalaman yang dimiliki anggota tim pengembang terhadap proses dan metodologi yang digunakan
E2	Pengalaman Aplikasi	Aplikasi yang sedang dikembangkan
E3	Pengalaman *Object-oriented*	Tingkat pengalaman dengan metode object-oriented
E4	Kemampuan memimpin analisis	Tingkat kemampuan pimpinan analis
E5	Motivasi	Tingkat motivasi anggota tim pengembang untuk menyelesaikan sistem
E6	Stabilitas Kebutuhan	Stabilitas kebutuhan pengguna akan sistem
E7	Staff *part time*	Apakah diperlukan pegawai *part-time* untuk membantu tim pengembang
E8	Tingkat kesulitan	Tingkat kesulitan dari bahasa pemrograman yang digunakan untuk mengembangkan sistem

2.4.3.4 Penentuan *Person Hour Multiplier* (PHM)

PHM didefinisikan sebagai jumlah jam kerja yang dibutuhkan untuk membangun sistem. Dalam bukunya mengenai UCP, Karner menganjurkan menggunakan nilai 20 untuk PHM. Tetapi berdasarkan pengalaman menggunakan UCP dalam berbagai proyek, para ahli

menyarankan dua buah angka yang digunakan, yaitu: 20 dan 28. Karya akhir ini menggunakan asumsi yang terakhir.

Aturan penentuan nilai PHM dapat dijabarkan sebagai berikut:

Jika jumlah dari (Nilai Efactor E1 sampai E6 *assigned value* < 3) dan
(Nilai Efactor E7 dan E8 *assigned value* > 3) <=2
PHM = 20

Jika Sum of (Nilai Efactor E1 sampai E6 *assigned value* < 3) and
(Nilai Efactor E7 dan E8 *assigned value* > 3) = 3 atau 4
PHM = 28

Selain itu, proyek patut dipertimbangkan ulang, memiliki resiko tinggi untuk gagal.

2.4.3.5 Penghitungan jumlah jam kerja yang dibutuhkan

Proses penghitungan jumlah jam kerja yang dibutuhkan didapatkan melalui rumus:

Besar Usaha dalam Jumlah Jam Kerja Orang = UCP * PHM

2.4.3.6 Lembar kerja estimasi *Use Case Point*

Berikut ini adalah lembar kerja yang digunakan dalam melakukan estimasi dengan menggunakan metode *use case point*. Nilai dalam faktor pembobotan merupakan nilai yang digunakan oleh standar industri (Dennis et.al, 2005), sedangkan nilai yang di-*assign* adalah nilai yang diisi oleh penulis berdasarkan pengamatan terhadap kebutuhan teknis dan lingkungan pengembangan sistem informasi yang akan dibangun.

Tabel 2.9 Lembar kerja *Unadjusted Actor Weight*

Tipe Aktor	Deskripsi	Faktor Pembobotan	Jumlah	Total
Simple	Sistem eksternal dengan API yang terdefinisi dengan baik	1		
Average	Sistem eksternal menggunakan antarmuka berbasis protokol serperti: HTTP, TCP/IP atau basisdata	2		
Complex	Manusia	3		
			Unadjusted Actor Weight Total **(UAW)**	

Tabel 2.10 Lembar kerja *Unadjusted Use Case Weight*

Tipe Use Case	Deskripsi	Faktor Pembobotan	Jumlah	Total
Simple	1 – 3 transaksi	5		
Average	4 – 7 transaksi	10		
Complex	> 7 transaksi	15		
			Unadjusted Use Case Weight Total **(UUCW)**	

Unadjusted Use Case Points **(UUCP) = UAW + UUCW**

Tabel 2.11 Lembar kerja faktor tingkat kompleksitas teknis

Faktor	Deskripsi	Bobot	Assigned Value (0 – 5)	Weighted Value	Notes
T1	Sistem Terdistribusi	2			
T2	Waktu tanggap atau kebutuhan akan tingkat performansi lewatan (*troughput*)	1			
T3	Efisiensi user online	1			
T4	Proses internal yang kompleks	1			
T5	Tingkat *reusability* kode program	1			
T6	Mudah diinstall	0.5			
T7	Mudah digunakan	0.5			
T8	Portabilitas	2			
T9	Mudah dilakukan perubahan	1			

Faktor	Deskripsi	Bobot	Assigned Value (0 – 5)	Weighted Value	Notes
T10	Konkurensi	1			
T11	Kebutuhan tingkat keamanan khusus	1			
T12	Akses langsung untuk pihak ketiga	1			
T13	Butuh pelatihan khusus bagi pengguna	1			
Nilai Faktor Teknis (TFactor)					

Technical Complexity Factor **(TCF) = 0.6 + (0.01 * TFactor)**

Nilai *assign value* berkisar antara 0-5 dengan "0" berarti faktor teknis tidak dibutuhkan oleh sistem dan "5" berarti faktor teknis sangat dibutuhkan oleh sistem dan menjadi *mandatory*.

Tabel 2.12 Lembar kerja faktor lingkungan

Faktor	Deskripsi	Bobot	Nilai yang diassign(0 – 5)	Nilai Bobot	Catatan
E1	Familiar dengan proses pengembangan sistem yang digunakan	1.5			
E2	Pengalaman Aplikasi	0.5			
E3	Pengalaman *Object-oriented*	1			
E4	Kemampuan memimpin analisis	0.5			
E5	Motivasi	1			
E6	Stabilitas Kebutuhan	2			
E7	Staff *part time*	-1			
E8	Tingkat kesulitan	-1			
Nilai Faktor Lingkungan (EFactor)					

Environmental Factor **(EF) = 1.4 + (-0.03 * EFactor)**

Adjusted Use Case Points **(UCP) = UUCP * TCF * ECF**

Besar Usaha dalam Jumlah Jam Kerja Orang = UCP * PHM

BAB III
OBJEK PENELITIAN

3.1 Objek Penelitian

Objek penelitian karya akhir ini adalah sistem informasi pembukuan dan sumber daya manusia di PT Tawang Swasti Rawikara. PT Tawang Swasti Rawikara adalah sebuah *holding company* yang berkedudukan di Gedung Aspine Lt. 2, Jl. RS Fatmawati 29, Jakarta Selatan.

PT Tawang Swasti Rawikara sendiri berdiri sejak tahun 1992 dan saat ini memiliki banyak anak perusahaan yang bergerak di berbagai bidang seperti: Property, Manufacturing, Trading, Services, Publishing, Information Technology. Saat ini kurang lebih ada 18 anak perusahaan aktif di bawah *holding*. PT Imani Prima tempat penulis bekerja merupakan salah satu anak perusahaan dari PT Tawang Swasti Rawikara yang bergerak di bidang Information Technology.

3.2 Kebutuhan akan sistem yang akan dibangun

PT Tawang Swasti Rawikara memerlukan suatu sistem informasi manajemen yang diperlukan saat ini adalah untuk mengakomodir kebutuhan pembukuan dan sumber daya manusia. Saat ini kedua sistem tersebut dibuat secara manual dengan menggunakan aplikasi Microsoft Excel.

Perkembangan perusahaan dengan bertambahnya jumlah anak perusahaan mengakibatkan bertambahnya aktivitas di PT Tawang Swasti Rawikara, sehingga kebutuhan untuk mengimplementasikan sebuah sistem informasi yang cukup serius dan sesuai dengan kebijakan perusahaan menjadi sebuah kebutuhan yang tidak dapat ditunda lagi.

3.2.1 Kebutuhan Sistem Informasi Pembukuan

Di bawah ini adalah daftar kebutuhan yang dikirimkan oleh manajemen PT Tawang Swasti Rawikara kepada PT Imani Prima sebagai pengembang aplikasi sistem informasi yang akan dibangun:

1. Semua jenis transaksi akan dibukukan pada rekening yang terkait dengan menggunakan nomor rekening atas dasar nota dan bukti pembayaran yang telah disyahkan oleh pejabat yang berwenang.
2. Selain transaksi, nota-nota pembukuan terkait penyusutan aktiva tetap, pencadangan, dan fungsi lainnya akan dibukukan setelah disyahkan oleh pejabat yang berwenang.
3. Nota-nota tersebut di atas diinput ke dalam komputer oleh petugas
4. Setiap menjelang akhir hari kerja setelah semua nota diinput, akan dilakukan "Run end of day", yaitu menutup pembukuan transaksi pada hari itu, dan selanjutnya dapat dicetak bila perlu:
    a. Daftar Rekapitulasi Harian semua transaksi debit dan kredit.
    b. Buku Besar (*General Ledger*) masing-masing rekening khususnya yang mengalami mutasi karena adanya pembukuan transaksi.
5. Dilakukan pengecekan kembali kebenaran data yang terinput dengan nota pembukuannya.
6. Perlu dilakukan koreksi entry/jurnal bila ditemukan adanya kesalahan input setelah mendapat pengesahan dari pejabat yang berwenang.
7. Program memberikan fleksibilitas dalam mencetak batch dan laporan-laporan yang diperlukan setiap saat selain mencetak Laporan Keuangan (*Financial Reports*) bulanan dan akhir tahun yang meliputi:
    a. Neraca (*Balance Sheet*)
    b. Laporan Rugi Labat (*Profit and Loss*)

8. Disusun daftar rekening (*Chart of Account*) yang dikelompokkan menurut jenis aktiva dan pasiva serta rekening pendapatan dan biaya. Misal, kelompok aktiva (01) dan pasiva (02) terdiri dari:

Tabel 3.1 Contoh Struktur Rekening PT Tawang Swasti Rawikara

No Rekening	Nama Rekening
Aktiva (01)	
Aktiva Lancar (011)	
0110100	Kas
0110200	Bank
0110201	Bank Mandiri
0120202	BCA
0110300	Persediaan
0110301	Persediaan bahan mentah
0110302	Persediaan barang setengah jadi
0110303	Persediaan barang jadi

9. Pengelompokan rekening menurut jenis dan nomor rekening perlu memperhatikan fleksibilitas sehingga bisa juga digunakan oleh anak-anak perusahaan/perusahaan afiliasi walau berbeda *line of business*. Namun dengan keseragaman jenis dan nomor rekening utama dalam *Balance Sheet* dan *Profit & Loss Statement* memungkinkan PT Tawang Swasti Rawikara melakukan konsolidasi laporan keuangan anak-anak perusahaan di kemudian hari.
10. Program untuk sistem akuntansi yang akan dikembangkan adalah untuk multi user.
11. Pengoperasian program memerlukan password sesuai dengan jenjang otorisasi, misal untuk Administrator, Manajer dan Operator. Masing-masing pengguna memiliki kewenangan yang berbeda dalam melakukan akses ke sistem.

3.2.2 Kebutuhan Sistem Informasi Sumber Daya Manusia

Di bawah ini adalah kebutuhan akan sistem informasi sumber daya manusia PT Tawang Swasti Rawikara:

1. Daftar pegawai yang lengkap dengan data pribadi dan data keluarga serta id pengguna untuk akses ke sistem informasi
2. Daftar tunjangan dan potongan yang dimiliki oleh tiap pegawai sehingga proses penghitungan *payroll* menjadi otomatis
3. Daftar absensi yang mencatat jumlah jam kerja dan jam lembur karyawan.
4. Sistem harus dapat mencatat berapa sisa cuti yang dimiliki oleh tiap pegawai dan dapat menolak ijin cuti apabila melebihi jumlah cuti yang boleh didapatkan.
5. Sistem dapat membuat daftar *payroll* yang otomatis langsung menghitung seluruh jam lembur, tunjangan dan potongan.
6. Sistem dapat menghitung pajak penghasilan dari tiap pegawai dan dapat langsung memposting pengeluaran gaji ke sistem pembukuan.
7. Sistem dapat mencetak history gaji, absensi, daftar cuti, daftar pajak yang dibayarkan.

BAB IV
ANALISIS DAN ESTIMASI SISTEM

4.1 Analisis Sistem

Analisis sistem berdasarkan UML 2.0 dibagi lagi menjadi 4 sub bagian, yaitu:

1 Spesifikasi Kebutuhan

2 Permodelan Fungsional

3 Permodelan Struktural

4 Permodelan Tingkah Laku (*behavioral*)

Karya akhir ini hanya akan membahas tahap analisis sampai pada permodelan fungsional.

4.1.1 Spesifikasi Kebutuhan

Spesifikasi kebutuhan dari sistem informasi manajemen ini didefinisikan oleh PT Imani Prima sebagai pengembang bersama dengan PT Tawang Swasti Rawikara sebagai klien. Kedua pihak ini selanjutnya disebut dengan sponsor proyek. Adapun spesifikasi kebutuhan yang didefinisikan oleh sponsor proyek ini adalah sebagai berikut:

4.1.1.1 Kebutuhan Fungsional

Daftar kebutuhan fungsional akan sistem informasi manajemen retail terdefinisi di bawah ini:

1. Sistem memiliki fungsi untuk menangani proses pembukuan (*General Ledger*).
2. Sistem memiliki fungsi untuk mengatur Sumber Daya Manusia (SDM).
3. Sistem memiliki fungsi untuk mengatur daftar Aktiva Tetap yang dimiliki perusahaan.
4. Sistem memiliki fungsi untuk membuat laporan keuangan yang dibutuhkan oleh perusahaan.

4.1.1.2 Kebutuhan Non Fungsional

Daftar kebutuhan non fungsional akan sistem informasi manajemen terdefinisi di bawah ini:

1. Kebutuhan Operasional
   a. Sistem dibuat dengan menggunakan aplikasi *client server.*
   b. Sistem dapat beroperasi di lingkungan Windows.
   c. Sistem dapat berjalan di lingkungan LAN.
   d. Sistem dapat melakukan fungsi ekspor atau impor data dari *file* data Microsoft Office (Word dan Excel).
2. Kebutuhan Performansi
   a. Waktu tunggu sistem untuk suatu aksi tidak boleh melebihi 5 detik.
   b. Kemudahan membuat tipe pengguna dan membagi hak akses untuk tiap pengguna terhadap setiap fungsi yang dimiliki oleh sistem.
3. Kebutuhan Pengamanan
   a. Sistem tidak dapat diakses oleh pengguna yang tidak memiliki otoritas.
   b. Akses terhadap fungsionalitas sistem ditentukan oleh jenis pengguna yang mengakses sistem.
4. Kebutuhan Budaya dan Politik

   Tidak terdefinisi

4.1.2 Permodelan Fungsional

Permodelan Fungsional menggambarkan proses bisnis dan interaksi antara sistem informasi dengan lingkungannya (Dennis et.al, 2005:163). Pengembangan sistem dengan

bantuan *use case* menggunakan dua tipe permodelan untuk mendefinisikan fungsionalitas dari sebuah sistem, yaitu: diagram *use case* dan *activity diagram* (diagram aktifitas).

Diagram aktifitas menggambarkan permodelan lojik dari proses bisnis dan *workflows* (alur kerja), sedangkan *use case* menggambarkan fungsi dasar dari sebuah sistem informasi. Keduanya dapat digunakan untuk mendefinisikan sistem yang saat ini telah berjalan dalam suatu organisasi (*as is system*) dan sistem yang akan dikembangkan (*to-be system*).

Pada permodelan fungsional yang menggunakan diagram *use case*, daftar kebutuhan fungsional yang terdefinisi di atas dimodelkan menjadi paket-paket sistem yang kemudian didekomposisi lagi menjadi beberapa *use case*. Interaksi antara *use case* dan *actor* membentuk sebuah diagram *use case*. Dalam karya akhir ini, penulis membuat sejumlah diagram *use case* yang merupakan dekomposisi dari paket yang telah didefinisikan sebelumnya.

4.1.3 Daftar Paket

Sebuah paket (*package*) merupakan salah satu elemen permodelan yang dapat mengandung elemen-elemen permodelan lain sehingga membentuk suatu kumpulan ataupun unit yang spesifik di dalam permodelan UML (Dennis et.al : 2005). Sebuah paket dapat memiliki elemen paket lain di dalamnya, bahkan dapat memiliki banyak elemen lain di dalamnya yang bisa dianggap sebagai sebuah sistem tersendiri. Elemen paket yang seperti ini disebut dengan *high level package*.

Karya akhir ini membagi spesifikasi kebutuhan PT Tawang Swasti Rawikara menjadi 3 buah paket. Daftar paket yang didefinisikan pada sistem informasi ini adalah sebagai berikut:

Tabel 4.1 Daftar Nama Paket

No	Nama Paket	Keterangan
1	Pengaturan Sistem	Paket yang bertugas mengatur *user* dan hak akses *user*
2	Buku Besar	Paket yang bertugas mengatur proses yang terkait dengan pembukuan
3	Sumber Daya Manusia	Paket yang bertugas mengatur struktur organisasi, data pegawai dan melakukan proses penggajian pegawai

4.1.4 Daftar Aktor

Aktor merepresentasikan pelaku yang berinteraksi dengan *use case* pada suatu sistem. Aktor dapat membantu membatasi dan memperjelas apa yang seharusnya dilakukan oleh sistem. (Suhendar: 2002)

Definisi Aktor di dalam UML ver 2.0 adalah seseorang atau sesuatu yang :

1. berinteraksi dengan sistem atau yang menggunakan sistem
2. melakukan *input* atau menerima informasi dari sistem
3. sistem eksternal dan tidak memiliki kontrol terhadap *use case*.

Dibantu dengan definisi di atas, sebuah aktor dapat diidentifikasi dengan mengecek:

1. Siapa yang langsung menggunakan sistem
2. Siapa yang bertanggung jawab memelihara sistem
3. *hardware* eksternal yang digunakan oleh sistem
4. sistem lain yang berinteraksi dengan sistem

Daftar aktor yang didefinisikan pada sistem informasi manajemen ini beserta perannya adalah sebagai berikut:

Tabel 4.2 Daftar aktor dan perannya

No	Nama Aktor	Peran
1	Administrator	1. Membuat daftar user 2. Membuat daftar hak akses 3. Mendefinisikan hak akses user
2	Manajer Pembukuan	1. Mengawasi proses pembukuan yang dilakukan oleh Staff Pembukuan 2. Melakukan koreksi terhadap proses pembukuan yang dilakukan oleh Staff Pembukuan
3	Staff Pembukuan	1. Melakukan aktivitas pembukuan 2. Membuat laporan pembukuan 3. Memberikan laporan pembukuan kepada Manajer Pembukuan
4	Staff Keuangan	1. Menerima daftar gaji yang harus dibayarkan
5	Staff HRD	1. Mengatur daftar departemen 2. Mengatur daftar pegawai 3. Mendata jam kerja pegawai 4. Melakukan proses payroll 5. Melakukan transfer gaji 6. Menghitung pajak penghasilan dan melaporkan pajak kepada Staff Pembukuan
6	Karyawan	1. Mengisi jam kerja, lembur 2. Mengisi data karyawan 3. Menerima gaji

4.1.5 Daftar *Use Case*

Daftar *use case* yang didefinisikan pada sistem informasi manajemen retail ini merupakan hasil dekomposisi dari paket sistem yang telah terdefinisi sebelumnya. Deskripsi *use case* dapat dilihat pada lampiran A, sedangkan diagram *use case* dapat dilihat pada lampiran B.

4.1.6 Diagram Aktifitas

Diagram Aktifitas memodelkan aliran kerja dari sebuah bisnis proses. Diagram ini digunakan untuk menggambarkan urutan aktivitas yang terjadi atau dilakukan oleh sistem

untuk mencapai suatu objektif tertentu. Diagram aktifitas juga dapat digunakan untuk memodelkan proses-proses yang berjalan paralel, konkuren dan proses yang kompleks. (Dennis et.al: 2005)

Diagram aktifitas dapat menggambarkan *high level business worklfow,* yang melibatkan banyak *use case,* sampai kepada detil dari tiap *use case* bahkan sampai ke tingkat *method.* Pada karya akhir ini, diagram aktifitas digunakan untuk menggambarkan urutan aktifitas dalam suatu *use case*. Diagram aktifitas untuk tiap *use case* dapat dilihat pada Lampiran C.

4.2 Estimasi ukuran proyek menggunakan *Use Case Points*

Setelah mendefinisikan semua *actor* dan *use case* yang terlibat dalam sistem, tahapan selanjutnya adalah menghitung pembobotan bagi *actor* dan *use case* yang terdefinisi. Semua nilai faktor pembobotan menggunakan nilai standarisasi *use case point* (Dennis et.al, 2005). Sedangkan untuk nilai yang diassign adalah asumsi yang dipakai oleh penulis berdasarkan pengalaman proyek sebelumnya.

4.2.1 Pengukuran *Unadjusted Use Case Points*

Unadjusted Use Case Point diukur dengan melakukan pembobotan terhadap *actor* dan *use case* yang terdefinisi. Hasil analisis terhadap sistem informasi manajemen yang akan dikembangkan menunjukkan bahwa ada 6 (enam) *actor* yang saling berinteraksi dan semuanya dikategorikan sebagai *complex actor*, karena semua *actor* adalah manusia. Tabel 4.3 mendefinisikan hasil pembobotan terhadap *actor* yang terlibat.

Tabel 4.3 Daftar *Unadjusted Actor Weight* hasil analisis

Tipe Aktor	Deskripsi	Faktor Pembobotan	Jumlah	Total
Simple	Sistem eksternal dengan API yang terdefinisi dengan baik	1	0	0
Average	Sistem eksternal menggunakan antarmuka berbasis protokol serperti: HTTP, TCP/IP atau basisdata	2	0	0
Complex	Manusia	3	6	18
		***Unadjusted Actor Weight Total* (UAW)**		**18**

Use case yang berhasil didefinisikan berjumlah 18 *use case*. Semua *use case* yang berhasil didefinisikan dikategorikan menjadi 3 tipe, yaitu: *simple*, *average*, *complex*. Pengkategorian *use case* ditentukan berdasarkan jumlah transaksi penting yang dilakukan oleh *use case* yang bersangkutan. Transaksi dapat diartikan sebagai sebuah proses yang dilakukan atau dialami oleh *actor*.

4.2.1.1 Simple Use Case

Use case dikategorikan sebagai *simple* apabila hanya mengandung 1-3 transaksi penting. Hasil analisis menunjukkan bahwa ada 8 (delapan) *simple use case* dari 18 *use case* yang berhasil didefinisikan pada fase analisis. Kedelapan *use case* tersebut adalah:

1. Edit Info Perusahaan
2. Edit Mata Uang
3. Cetak Laporan Keuangan
4. Edit Departemen
5. Edit Tunjangan
6. Edit Potongan
7. Edit Pengguna
8. Edit Hak Akses

Kedelapan *use case* hanya mengandung transaksi sederhana yang mengakomodasi interaksi sederhana dengan pengguna dan melakukan *query* yang sederhana dengan basis data. Operasi yang dilakukan hanya pembacaan data dan proses *update* yang hanya melibatkan satu buah *table* di basis data. Deskripsi *use case* dapat dilihat pada lampiran A.

4.2.1.2 *Average Use Case*

Use case dikategorikan sebagai *average* apabila hanya mengandung 4-7 transaksi penting. Hasil analisis menunjukkan bahwa ada 4 (empat) *average use case* dari 18 *use case* yang berhasil didefinisikan pada fase analisis. Keempat *use case* tersebut adalah:

1. Edit Aktiva Tetap

   Pada *use case* ini, aktifitas tidak hanya mengkomodasi interaksi antara pengguna dengan sistem dan *query* yang melibatkan lebih dari satu tabel di basis data, tetapi ada beberapa transaksi seperti *assign* kode akun tertentu untuk jurnal otomatis terhadap tiap aktiva tetap yang baru, kode akun untuk penyusutan dan kode akun untuk biaya penyusutan. Selain itu, *use case* ini juga mendefinisikan data-data yang diperlukan untuk menghitung penyusutan nilai bukunya. Validasi terhadap data masukan juga perlu dilakukan.

2. Simulasi Penyusutan Aktiva Tetap

   Aktifitas yang cukup kompleks dan cukup memakan waktu dalam pengembangan program untuk *use case* ini adalah membuat rutin program untuk tiap tipe penyusutan yang terdefinisi, yaitu: *Straight Line*, *Units of Production*, *Declining Balance*. Selain itu, *use case* ini juga mengakomodasi rutin program untuk proses penghitungan jumlah hari yang digunakan dalam proses penghitungan penyusutan serta menampilkan hasil penghitungan simulasi penyusutan.

3. Edit Pegawai

   Use case ini memiliki beberapa transaksi yang cukup rumit terutama dalam *assign* daftar tunjangan, daftar potongan, *user id* serta data-data keluarga untuk tiap pegawai yang akan digunakan dalam proses penghitungan pajak penghasilan terhadap pegawai yang bersangkutan.

4. Assign Hak Akses ke Pengguna

   Use case ini memiliki beberapa transaksi yang rumit terutama dalam melakukan *assign* hak akses ke tiap *user id* yang terdefinisi untuk masing-masing pegawai. *Use case* ini juga dapat mendefinisikan *group* hak akses yang berisi beberapa hak akses tertentu yang akan di-*assign* ke beberapa tipe *user id.* Selain itu, *use case* ini juga mengakomodasi pendefinisian hak akses yang unik untuk tiap *user id.* Tiap usaha yang dilakukan oleh tiap *user* juga akan disimpan dalam sebuah *log* tertentu.

4.2.1.3 *Complex Use Case*

Use case yang dikategorikan ke dalam *complex* apabila *use case* ini banyak mengandung rutin program validasi, melibatkan banyak *query* yang *complex* yang melibatkan banyak tabel serta memiliki banyak kondisi pencabangan yang harus diakomodasi oleh sistem. Ada 6 (enam) buah *complex use case* yang berhasil didefinisikan pada sistem informasi manajemen ini, yaitu:

1. Edit Tabel Akun

   Use case ini tidak hanya mendefinisikan aktifitas penambahan, *update*, dan hapus tabel akun, tetapi memiliki beberapa rutin program untuk menangani struktur tabel akun (*parent-child*), penggolongan akun (*asset*, *liability*, *equity*, *income*, *expense*), hitung total saldo untuk *parent* yang memiliki banyak *child*, beberapa rutin validasi (cek id akun, posisi normal saldo tiap akun, tampilan di tabel akun).

2. Edit Jurnal

Use case ini melakukan proses edit transaksi jurnal yang melibatkan akun-akun yang terdefinisi pada tabel akun. Proses lain yang diakomodasi oleh *use case* ini adalah *posting* dan *unposting* terhadap jurnal, menampilkan *ledger* tiap akun, menghitung total mutasi *debet* maupun *kredit* tiap akun, dan rutin program validasi (jurnal harus sudah *balance* sebelum disimpan ke dalam sistem, akun yang terlibat harus sudah terdaftar di tabel akun, tidak boleh ada akun yang berstatus "tidak aktif" terlibat dalam pembuatan jurnal).

3. Edit Koreksi Jurnal

Use case ini melibatkan dua jenis pengguna, staff keuangan dan manajer keuangan. Apabila ada kesalahan terhadap jurnal yang sudah terlebih dulu di-*posting*, maka staff keuangan tidak berhak untuk melakukan perubahan atau membuat jurnal balik. Kesalahan harus dilaporkan dulu kepada manajer keuangan. Manajer keuangan yang akan melakukan jurnal balik. Kompleksitas *use case* ada pada pembagian wewenang pengguna, pembuatan *reminder* otomatis kepada manajer keuangan, pembuatan jurnal balik otomatis dan pengisian log kesalahan *input* jurnal di sistem.

4. Tutup Buku

Use case ini adalah *use case* yang paling rumit di antara semua *use case* yang terdefinisi. Beberapa aktivitas penting yang dilakukan oleh *use case* ini, antara lain: proses menghitung penyusutan dari seluruh aktiva tetap yang dimiliki oleh perusahaan, membuat jurnal otomatis untuk penyusutan aktiva tetap, menghitung *Net Income* perusahaan dengan cara mencari selisih antara total *income* dan total *expense*, membuat jurnal otomatis untuk transaksi ubah komposisi modal, mengubah parameter bulan dan tahun berjalan jika di akhir tahun, menampilkan data-data penting hasil dari proses tutup buku.

5. Buat Laporan Keuangan

   Fokus *use case* ini lebih kepada pembuatan *query* untuk mengumpulkan data-data yang berasal dari berbagai macam tabel di basis data. Data-data tersebut disusun sesuai dengan *template* laporan keuangan standar (laporan rugi laba, laporan perubahan modal, neraca, neraca percobaan, *general ledger*). Aktifitas yang memakan *resource* cukup banyak adalah pengumpulan data dan pembuatan *layout* laporan.

6. Buat Daftar Payroll

   Use case ini juga termasuk *use case* yang memiliki transaksi yang paling banyak. *Use case* ini mengakomodasi pengisian jam kerja karyawan baik jam kerja normal, cuti, ijin, sakit maupun lembur. Sistem akan menghitung secara otomatis berapa nominal yang harus dibayarkan kepada pegawai sebagai ganti jam kerja dan jam lembur tiap karyawan. *Use case* ini juga menghitung secara otomatis berapa nominal pajak penghasilan yang harus dibayarkan, serta membuat surat permohonan dana untuk pembayaran gaji pegawai dan pajak penghasilan kepada staff keuangan. *Use case* juga membuat laporan *payslip* yang nantinya akan ditandatangani oleh pegawai yang bersangkutan sebagai tanda bahwa dia telah menerima gaji dari perusahaan.

Tabel 4.4 mendefinisikan hasil pembobotan terhadap seluruh *use case* yang berhasil didefinisikan. Penjumlahan antara *Unadjusted Use Case Weight* dan *Unadjusted Actor Weight* menghasilkan *Unadjusted Use Case Points*.

Tabel 4.4 Daftar *Unadjusted Use Case Weight* hasil analisis

Tipe Use Case	Deskripsi	Faktor Pembobotan	Jumlah	Total
Simple	1 – 3 transaksi	5	8	40
Average	4 – 7 transaksi	10	4	40
Complex	> 7 transaksi	15	6	90
			Unadjusted Use Case Weight Total **(UUCW)**	**170**

Unadjusted Use Case Points **(UUCP) = UAW + UUCW = 18 + 170 = 188**

4.2.2 Pengukuran kompleksitas teknis pengembangan

Kompleksitas teknis pengembangan yang didefinisikan pada proyek pengembangan sistem informasi manajemen ini dibahas secara detil sebagai berikut:

T1. Sistem terdiri atas sebuah *server* dan beberapa *client* yang terhubung dengan *server.* Walaupun sistem terdistribusi, tetapi sistem informasi manajemen menggunakan basis data yang sama dan tidak ada proses replika basis data, sehingga untuk T1, penulis memberikan nilai sebesar 3 (tiga).

T2. Klien sangat mementingkan waktu tanggap sistem yang cepat. Sehingga nilai untuk T2 adalah 5 (lima).

T3. Tingkat efisiensi pengguna dalam menggunakan sistem tidak terlalu diperlukan tetapi juga tidak boleh terlalu lambat dan tidak efisien, sehingga penulis memilih nilai 3 (tiga) untuk faktor ini.

T4. Sistem informasi manajemen yang dikembangkan hanya memiliki cukup banyak proses dengan kompleksitas yang tinggi dibandingkan semua *use case* yang terdefinisi. Jumlah *average* dan *complex use case* adalah 10 (sepuluh) dibandingkan dengan jumlah *simple use case* yang hanya berjumlah 8 (delapan), sehingga penulis memilih nilai 4 (empat) untuk faktor ini.

T5. Sistem informasi manajemen yang dikembangkan tidak memiliki fitur yang sangat besar tetapi karena waktu pengembangan yang diberikan tidak terlalu lama, teknik *reusability* kode program tetap penting untuk digunakan. Penulis memberikan nilai 4 (empat) untuk faktor ini.

T6. Klien menuntut sistem agar mudah untuk diinstalasi. (Nilai =5)

T7. Klien menuntut sistem mudah digunakan. (Nilai=5)

T8. Klien menuntut sistem dapat diinstall di semua jenis sistem operasi windows yang digunakan di lingkungan PT Tawang Swasti Rawikara. Walaupun terjadi perbedaan

yang cukup signifikan antara lingkungan di Windows XP dan Windows versi di bawahnya, tetapi karena masih dalam satu jenis sistem operasi, maka penulis memberikan nilai 3(tiga) untuk T8.

T9. Klien menginginkan kemudahan dalam mengubah sistem untuk mengakomodasi perubahan regulasi di Indonesia yang sangat cepat dan tidak pasti. (Nilai=5)

T10. Konkurensi tidak dibutuhkan oleh sistem. (Nilai=0)

T11. Sistem membutuhkan keamanan untuk akses data, tetapi tidak memerlukan akses khusus yang berlebih karena pengguna yang akan *login* ke sistem hanya pengguna tertentu saja. (Nilai=3)

T12. Sistem tidak menyediakan akses langsung ke sistem oleh pihak ketiga. (Nilai=0)

T13. Karena pengguna sudah terbiasa dengan aplikasi excel dan tidak terbiasa dengan sistem informasi manajemen, maka pelatihan khusus bagi pengguna memegang peranan yang signifikan. (Nilai=5)

Nilai bobot faktor teknis sistem informasi yang akan dibangun dapat dilihat pada Tabel 4.5.

Tabel 4.5 Daftar faktor tingkat kompleksitas teknis hasil analisis

Faktor	Deskripsi	Bobot	Nilai yang diassign(0 – 5)	Nilai Bobot
T1	Sistem Terdistribusi	2	3	6
T2	Waktu tanggap atau kebutuhan akan tingkat performansi lewatan (*troughput*)	1	5	5
T3	Efisiensi user online	1	3	3
T4	Proses internal yang kompleks	1	4	4
T5	Tingkat *reusability* kode program	1	4	4
T6	Mudah diinstall	0.5	5	2.5
T7	Mudah digunakan	0.5	5	2.5
T8	Portabilitas	2	3	6
T9	Mudah dilakukan perubahan	1	5	5
T10	Konkurensi	1	0	0
T11	Kebutuhan tingkat keamanan khusus	1	3	3

Faktor	Deskripsi	Bobot	Nilai yang diassign(0 – 5)	Nilai Bobot
T12	Akses langsung untuk pihak ketiga	1	0	0
T13	Butuh pelatihan khusus bagi pengguna	1	5	5
Nilai Faktor Teknis (TFactor)				**46**

***Technical Complexity Factor* (TCF) = 0,6 + (0,01 * TFactor)= 0,6 + (0,01 * 40) = 1,06**

4.2.3 Pengukuran kompleksitas lingkungan pengembangan

Kompleksitas lingkungan pengembangan yang didefinisikan pada proyek pengembangan sistem informasi manajemen ini dibahas secara detil sebagai berikut:

E1. Karena sebagian besar dari *programmer* menggunakan *staff part time*, maka hanya sedikit anggota tim yang familiar dengan proses pengembangan sistem yang digunakan. Tapi peran dari *programmer* yang berpengalaman cukup dominan, sehingga bobot untuk E1 penulis memberikan nilai=3.

E2. Tidak semua anggota tim yang berpengalaman dalam mengembangkan aplikasi cukup memiliki pengetahuan dan pengalaman dalam mengembangkan sistem informasi yang akan dibangun, sehingga penulis memberi nilai 2(dua) untuk faktor lingkungan E2.

E3. Tidak semua anggota tim memiliki pengalaman dalam mengembangkan sebuah sistem informasi berdasarkan *Object-oriented*, sehingga untuk faktor lingkungan E3, penulis memberi nilai 2(dua).

E4. Kemampuan pemimpin analis cukup tinggi untuk memimpin anggota tim dan telah memiliki banyak pengalaman dalam memimpin *programmer* dalam mengembangkan sebuah sistem informasi. Penulis memberikan nilai cukup tinggi, yaitu 4 (empat).

E5. Motivasi sangat tinggi dari seluruh anggota tim. (nilai=5)

E6. Stabilitas kebutuhan akan sistem informasi berada pada posisi tengah, artinya klien tidak terlalu sering mengubah *requirement* tetapi perubahan yang dilakukan ternyata melibatkan cukup banyak usaha untuk mengakomodir keinginan klien tadi. (nilai=3)

E7. Ketergantungan terhadap *staff part time* sangat tinggi, sehingga nilai yang diberikan adalah 5 (lima).

E8. Microsoft Visual Basic 6.0 memiliki nama yang cukup terkenal dan banyak digunakan. Sebagian besar *programmer* pernah menggunakan MS VB 6.0, sehingga untuk E8, penulis memberikan nilai 2 (dua).

Daftar faktor lingkungan dapat dilihat pada Tabel 4.6 di bawah ini.

Tabel 4.6 Daftar faktor lingkungan hasil analisis skenario I

Faktor	Deskripsi	Bobot	Nilai yang diassign(0 – 5)	Nilai Bobot
E1	Familiar dengan proses pengembangan sistem yang digunakan	1,5	2	3
E2	Pengalaman Aplikasi	0,5	2	1
E3	Pengalaman *Object-oriented*	1	2	2
E4	Kemampuan memimpin analisis	0,5	4	2
E5	Motivasi	1	5	5
E6	Stabilitas Kebutuhan	2	3	6
E7	Staff *part time*	-1	5	-5
E8	Tingkat kesulitan	-1	2	-2
			Nilai Faktor Lingkungan (Efactor)	**12**

***Environmental Factor* (EF) = 1,4 + (-0,03 * EFactor) = 1,4 + (-0,03*12) = 1,04**

4.2.4 Jumlah jam kerja orang

Berdasarkan nilai-nilai yang diperoleh di atas, maka dapat ditentukan nilai *Adjusted Use Case Points* (UCP) seperti terlihat di bawah ini:

(UCP) = UUCP * TCF * ECF = 188 * 1,04 * 1,04 = 203,3408

Setelah mendapatkan nilai UCP, tahapan berikutnya yang perlu dilakukan adalah menentukan nilai PHM yang akan digunakan untuk menghitung jumlah jam kerja orang. Langkah menghitung PHM:

1. Hitung jumlah banyaknya Efactor (E1-E6) yang lebih kecil dari 3. Berdasarkan data, ada 3 Efactor yang lebih kecil dari 3, yaitu: E1, E2 dan E3.
2. Hitung jumlah banyak Efactor (E7 dan E8) yang lebih besar dari 3 adalah E7. Jadi jumlahnya hanya satu.
3. Jumlahkan hasil dari nomor 1 dan nomor 2. Hasilnya adalah 4 (empat).
4. Berarti Nilai PHM yang digunakan adalah **28**

Berdasarkan hasil di atas, maka Besar Usaha dalam Jumlah Jam Kerja Orang = 203,3408 * 28 = **5.693,5424 jam**.

Hasil penghitungan di atas sangat tinggi untuk pembuatan aplikasi ini. Apabila SDM yang di-*assign* untuk proyek ini adalah 6 orang dan dengan asumsi satu minggu adalah 40 jam kerja, maka waktu yang dibutuhkan untuk membangun sistem informasi manajemen ini adalah 23,72 minggu atau sekitar **5,9 bulan**.

Hasil di atas akan berbeda apabila tim yang akan mengerjakan memiliki kondisi seperti di bawah ini:

1. SDM yang digunakan adalah SDM internal PT Imani Prima (bukan *part time*). (E7=1).
2. Memiliki pengalaman dan pengetahuan yang memadai terhadap aplikasi yang mirip dengan sistem informasi yang akan dibangun. Dalam proyek ini, tim seharusnya memiliki pengalaman dan pengetahuan dalam mengembangkan aplikasi pembukuan dan aplikasi sumber daya manusia atau penggajian. (E2=3)
3. Memiliki pengalaman dan pengetahuan yang memadai dalam pengembangan aplikasi berbasis objek. (E3=3)

Jika skenario di atas berlaku, maka nilai E1 dan E2 bernilai lebih besar dari 3 dan E7 bernilai lebih kecil dari 3. Dengan kondisi yang seperti ini, maka Efactor menjadi

Tabel 4.7 Daftar faktor lingkungan hasil analisis skenario II

Faktor	Deskripsi	Bobot	Nilai yang diassign(0 – 5)	Nilai Bobot
E1	Familiar dengan proses pengembangan sistem yang digunakan	1,5	3	4,5
E2	Pengalaman Aplikasi	0,5	3	1,5
E3	Pengalaman *Object-oriented*	1	3	3
E4	Kemampuan memimpin analisis	0,5	4	2
E5	Motivasi	1	5	5
E6	Stabilitas Kebutuhan	2	3	6
E7	Staff *part time*	-1	1	-1
E8	Tingkat kesulitan	-1	2	-2
Nilai Faktor Lingkungan (EFactor)				**19**

Environmental Factor (EF) = 1.4 + (-0.03 * EFactor) = 1.4 + (-0.03*19) = **0,83**

Adjusted Use Case Points (UCP) = UUCP * TCF * ECF = 188 * 1,04 * 0,83 = **162,2816**

PHM yang digunakan dalam penghitungan adalah 20, sehingga Besar Usaha dalam Jumlah Jam Kerja Orang = 162,2816 * 20= 3.245,632 jam. Dengan asumsi jumlah anggota tim adalah 6 orang, dan waktu kerja adalah 40 jam seminggu, maka waktu yang dibutuhkan untuk mengembangkan sistem informasi ini adalah: 13,5 minggu atau sekitar **3,375 bulan**. Hasil estimasi bisa diterima.

Berdasarkan hasil penelitian di atas, maka skenario yang digunakan untuk pengembangan sistem informasi pembukuan dan sumber daya manusia untuk PT Tawang Swasti Rawikara adalah skenario kedua.

BAB V
PENUTUP

5.1 Kesimpulan

Estimasi ukuran proyek pengembangan sebuah sistem informasi merupakan salah satu aktifitas yang sangat penting dari rangkaian aktifitas dalam manajemen proyek teknologi informasi. Dengan mengetahui berapa banyak *resource* yang dibutuhkan untuk mengembangkan sebuah sistem informasi di tahap awal manajemen proyek, seorang manajer proyek dapat membuat sebuah perencanaan yang lebih baik untuk menghindari tiga kriteria kegagalan sebuah proyek teknologi informasi(*On Time Delivery*, *Out of Scope*, *Over Budget*).

Use case point merupakan salah satu metode estimasi yang mudah digunakan dan menawarkan sebuah *framework* yang cukup jelas dan terstruktur untuk melakukan proses estimasi ukuran proyek. Di awal tender atau proyek, seringkali pengembang menghadapi kesulitan dalam mendefinisikan berapa lama waktu yang dibutuhkan untuk menyelesaikan sebuah proyek sistem informasi, mengatur jumlah *programmer* yang terlibat, serta kualifikasi *skill* apa yang dibutuhkan, dan apa dampaknya terhadap proyek. Dengan melakukan eksperimen terhadap nilai faktor teknis dan faktor lingkungan pada *use case point*, tim pengembang dapat langsung mengetahui apa pengaruh perubahan nilai tadi terhadap keseluruhan proyek. Hal ini dapat membantu tim pengembang untuk menentukan batasan-batasan yang akan digunakan dalam manajemen proyek tersebut.

Dalam melakukan estimasi ukuran proyek, tim pengembang perlu mewaspadai terhadap beberapa faktor kompleksitas teknis dan faktor kompleksitas lingkungan yang memiliki bobot yang cukup besar. Faktor kompleksitas teknis yang signifikan adalah T1 (sistem terdistribusi) dan T8 (portabilitas sistem). Sedangkan faktor kompleksitas lingkungan yang signifikan adalah E1 (*familiar* dengan proses pengembangan yang dipakai) dan E6 (stabilitas kebutuhan pengguna akan sistem). Tim pengembang harus waspada apabila nilai

yang diberikan untuk keempat faktor di atas besar, bisa dipastikan bahwa ukuran proyek akan menjadi sangat besar dan menggunakan *resources* yang banyak.

Karya akhir ini membahas dua skenario yang digunakan untuk mengembangkan sistem informasi manajemen untuk PT Tawang Swasti Rawikara. Skenario pertama, sebagian besar anggota tim terdiri atas pegawai kontrak (*part time*) dan rata-rata belum memiliki pengalaman dan pengetahuan yang memadai untuk membangun sebuah sistem informasi. Skenario kedua, anggota tim terdiri atas karyawan tetap PT Imani Prima yang memiliki pengalaman pemrograman selama kurang lebih 2 tahun dan sudah sering menggunakan metodologi *object oriented programming*. Faktor teknis dari aplikasi tidak berubah dan tidak mungkin berubah karena sudah menjadi standarisasi dari klien sendiri.

Tim pengembang dari PT Imani Prima yang akan membangun sistem informasi manajemen untuk PT Tawang Swasti Rawikara awalnya cukup yakin dengan kemampuan tim untuk menyelesaikan proyek ini dalam waktu 4 (empat) bulan dengan menggunakan skenario pertama. Proses estimasi memberikan hasil dengan perbedaan yang cukup signifikan untuk kedua skenario di atas. Skenario pertama ternyata membutuhkan waktu pengembangan selama 5,9 bulan. Sedangkan skenario kedua ternyata membutuhkan waktu pengembangan selama 3,375 bulan. Hasil kedua skenario sebenarnya masih memenuhi waktu yang diberikan oleh klien, tetapi skenario pertama sangat rentan terhadap *Late Delivery*. Apabila ditambahkan faktor risiko sebesar 10%, maka waktu yang dibutuhkan oleh skenario pertama menjadi 6,4 bulan (sudah tidak lagi memenuhi batasan dari klien) dan skenario kedua menjadi 3,7 bulan. Tim pengembang akhirnya memutuskan untuk menggunakan skenario kedua.

5.2 Saran

Setelah melakukan penelitian ini, berikut ini adalah beberapa saran yang perlu dilakukan untuk memperbaiki penelitian yang telah dilakukan:

1. Dalam pemrograman berbasis objek (*object-oriented*), fase analisis seharusnya melibatkan tidak hanya *functional modelling*, tetapi juga sampai kepada *structural modelling* dan *behavioral modelling*. Apabila proyek benar-benar akan diimplementasikan, sebaiknya tim pengembang melanjutkan proses analisis dengan membangun diagram-diagram lainnya untuk lebih memahami proses bisnis yang ada pada sistem informasi yang akan dibangun. Walaupun proses estimasi dengan menggunakan *use case point* hanya memanfaatkan hasil dari *use case diagram*, tetapi dengan mendefinisikan semua diagram-diagram yang lain, tim pengembang dapat memiliki informasi yang cukup banyak dan lebih rinci dalam menentukan kompleksitas tiap *use case* yang berhasil didefinisikan. Diharapkan dengan ini, nilai yang terdefinisi pada *adjusted use case point* lebih mendekati bobot sebenarnya pada manajemen proyek.
2. Perlu dilakukan kajian ulang terhadap *technical factor* dan *environment factor* pada *use case point*, apakah sudah mewakili seluruh aspek yang mungkin mempengaruhi proses pengembangan sebuah sistem informasi manajemen. Sebagai contoh untuk T1 pada faktor kompleksitas teknis (sistem terdistribusi). *Use case point* memberikan bobot sebesar 2 (dua) untuk T1. Padahal saat ini untuk membuat sebuah sistem terdistribusi sederhana tidak membutuhkan *effort* yang banyak. Beberapa *tools* pemrograman telah menyediakan objek-objek yang dapat langsung digunakan untuk menjamin bahwa sistem yang dibangun dapat menjadi sebuah sistem terdistribusi. Contoh lain adalah T6 pada faktor kompleksitas teknis (kemudahan instalasi). Walaupun *use case point* memberikan bobot yang tidak besar tetapi dengan banyaknya *tools* untuk membuat *installer* yang ada saat ini, bisa jadi

faktor kemudahan instalasi tidak lagi relevan untuk diikutkan dalam daftar kompleksitas teknis pembangunan sistem.

3. *Use case point* memiliki kelemahan dalam mendistribusikan jam kerja kepada tiap *job role* tim pengembang. Hal ini terjadi karena untuk tiap *use case* yang terdefinisi, peran serta kuantitas jam kerja *job role* berbeda-beda. *Use case point* memberikan bobot pekerjaan yang sama untuk semua *complex use case*. Beberapa *complex use case* memiliki perbedaan yang signifikan dinilai dari segi banyaknya *form* yang harus dibuat, *query* ke basis data serta banyaknya jumlah validasi. Untuk *simple* dan *average use case*, perbedaan antar *use case* tidak terlalu signifikan, sehingga dapat diabaikan. Tetapi *complex use case*, perbedaan antar *use case* bisa menjadi sangat jauh. Salah satu solusi yang mungkin dapat dilakukan adalah dengan melakukan dekomposisi terhadap *use case* yang masih sangat *complex*. Saat ini penulis pun masih belum meneliti sampai dimana batas atas kompleksitas suatu *use case* yang menandakan bahwa *use case* tersebut layak untuk didekomposisi menjadi beberapa *use case* yang lebih *simple*.

DAFTAR PUSTAKA

Schwalbe, Kathy (2006). *Information Technology Project Management*, Canada, Thomson Course Technology

Dennis, Alan; & Wixom, Barbara H.; & Tegarden, David (2005). *System Analysis and Design with UML Version 2.0: An Object Oriented Approach*, New Jersey, John Wiley & Sons, Inc

Suhendar, A; Gunadi, Hariman (2002). *Visual Modelling Menggunakan UML dan Rational Rose*, Bandung, Informatika Bandung.

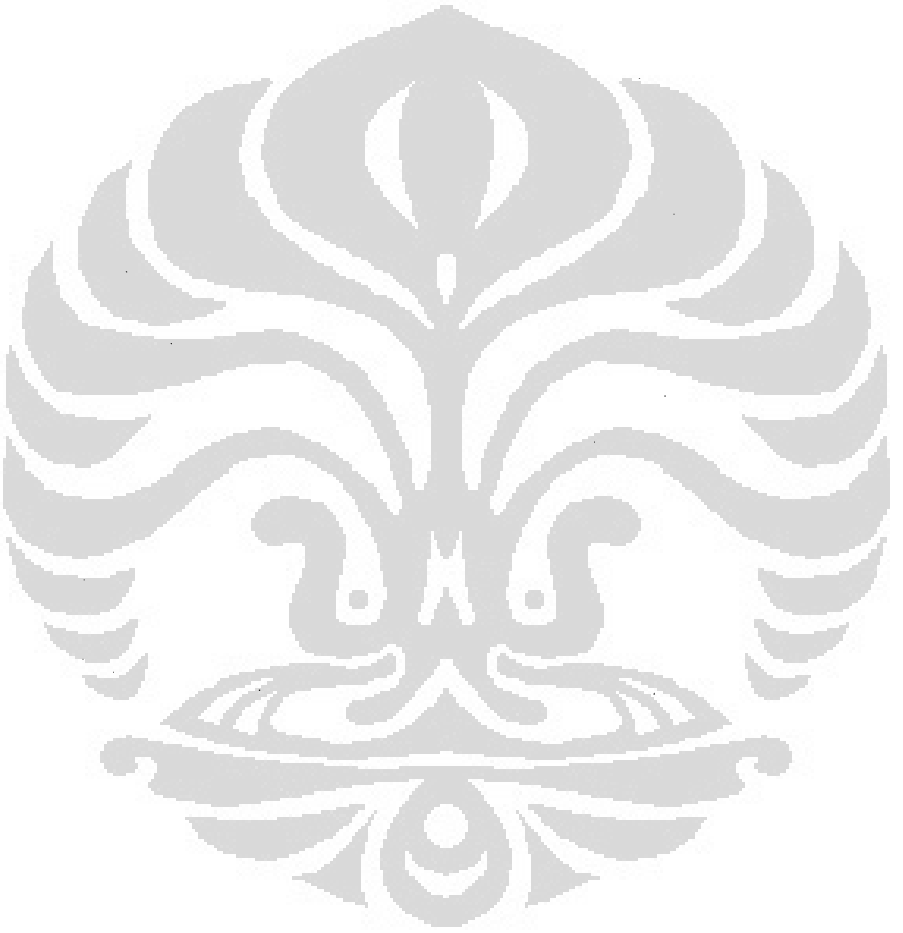
Jaya, Andri Kusuma (2005). *Analisis Perancangan Sistem Informasi Akademis dengan Orientasi Objek Menggunakan Unified Modelling Language Studi Kasus Sekolah Dedikasi Edukasi Kualiva*, Jakarta, Magister Akuntansi Universitas Indonesia

Kirsten, Ribu (2001). *Estimating Object-Oriented Software Projects with Use Cases, Master of Science Thesis*, University of Oslo, Department of Informatics

Meredith, J.P.; & Mantel Jr., S.J.; (2006). *Project Management: A Managerial Approach.* New Jersey, John Wiley & Sons, Inc.

Wikipedia Foundation, Inc. (2006). **Rapid Application Development.** http://en.wikipedia.org/wiki/Rapid_application_development, September 13, 2006.

Wikipedia Foundation, Inc. (2006). **Agile Software Development.** http://en.wikipedia.org/wiki/Agile_software_development, September 12, 2006.



LAMPIRAN A

USE CASE DESCRIPTION

Use Case Description

Nama Use Case: Edit Informasi Perusahaan	ID: 1.1	Tingkat Penting: Tinggi
Aktor Utama: Staff Pembukuan	Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan:		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff Pembukuan mengisi data informasi umum perusahaan.		
Pemicu: Staff Pembukuan memilih menu Edit Info Perusahaan Tipe: Internal		
Relationships: Association: Staff Pembukuan Include: Extend: Generalization:		
Aliran Kejadian Normal: 1. Staff Pembukuan memasukkan informasi umum perusahaan (nama, alamat, telpon, fax, website dan info umum lainnya. 2. Staff Pembukuan memilih info currency acuan yang dipakai 3. Staff Pembukuan memasukkan data-data yang terkait dengan pajak dan data tahun fiskal perusahaan.		
SubFlows: S-1.		
Alternate/Exceptional Flows: 2.a Staff Pembukuan memasukkan *currency* baru		

Nama Use Case: Edit Tabel Akun	ID: 1.2	Tingkat Penting: Tinggi
Aktor Utama: Staff Pembukuan	Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajer Pembukuan		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff Pembukuan membuat tabel akun dan melakukan setting terhadap tiap akun yang dibuat.		
Pemicu: Staff Pembukuan memilih menu Edit Tabel Akun Tipe: Internal		
Relationships: Association: Staff Pembukuan Include: Extend: Generalization:		
Aliran Kejadian Normal: 1. Staff Pembukuan mengisi data akun baru 2. Staff Pembukuan menekan tombol OK		

SubFlows: S1. Staff Pembukuan memilih klasifikasi akun S2. Staff Pembukuan menentukan hirarki akun
Alternate/Exceptional Flows: S.1.a Staff Pembukuan membuat klasifikasi akun baru S.2.a Staff Pembukuan membatalkan pembuatan akun baru

Nama Use Case: Edit Kurs Mata Uang		ID: 1.3	Tingkat Penting: Tinggi
Aktor Utama: Staff Pembukuan	Tipe Use Case: Detil		
Stakeholders dan Pihak lain yang berkepentingan: 1. Staff Pembukuan			
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff Pembukuan membuat daftar mata uang beserta nilai kursnya terhadap mata uang *default* yang digunakan.			
Pemicu: Staff Pembukuan memilih menu Edit Kurs Mata Uang Tipe: Internal			
Relationships: Association: Staff Pembukuan Include: Extend: Generalization:			
Aliran Kejadian Normal: 1. Staff Pembukuan memasukkan informasi kurs mata uang baru 2. Staff Pembukuan menentukan nilai mata uang tersebut terhadap mata uang *default* yang dipakai 3. Staff Pembukuan menekan tombol OK			
SubFlows:			
Alternate/Exceptional Flows: 3.a. Staff Pembukuan membatalkan Edit Kurs Mata Uang			

Nama Use Case: Edit Jurnal		ID: 1.4	Tingkat Penting: Tinggi
Aktor Utama: Staff Pembukuan	Tipe Use Case: Detil		
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajer Pembukuan			
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff Pembukuan membuat jurnal baru berdasarkan transaksi.			
Pemicu: Staff Pembukuan memilih menu Edit Jurnal Tipe: Internal			

Relationships: Association: Staff Pembukuan Include: Extend: Koreksi Jurnal Generalization:
Aliran Kejadian Normal: 1. Staff Pembukuan memasukkan informasi *header* jurnal (deskripsi jurnal, tanggal transaksi, dll) 2. Staff Pembukuan memilih akun yang terlibat dalam jurnal 3. Staff Pembukuan memasukkan nominal transaksi 4. Staff Pembukuan menekan tombol OK
SubFlows: S.1 Sistem mengecek apakah total nominal *balance* atau tidak.
Alternate/Exceptional Flows: 4.a Staff Pembukuan membatalkan Edit Jurnal

Nama Use Case: Tutup Buku		ID: 1.5	Tingkat Penting: Tinggi
Aktor Utama: Staff Pembukuan		Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajer Pembukuan			
Penjelasan Singkat:	Use Case ini menjelaskan bagaimana Staff Pembukuan melakukan proses Tutup Buku untuk bulan yang berjalan.		
Pemicu: Staff Pembukuan memilih menu Tutup Buku Tipe: Internal			
Relationships: Association: Staff Pembukuan Include: Extend: Generalization:			
Aliran Kejadian Normal: 1. Staff Pembukuan menekan tombol OK untuk melakukan proses Tutup Buku			
SubFlows: S1. Hitung Penyusutan Aktiva Tetap S2. Buat jurnal otomatis untuk penyusutan Aktiva Tetap S3. Hitung Net Income S4. Buat jurnal otomatis ubah komposisi modal S5. Ubah bulan berjalan			
Alternate/Exceptional Flows: S.4.a Buat jurnal otomatis penjumlahan nominal akun Laba Rugi Tahun Berjalan kepada akun Laba Rugi Tahun Lalu S.5.a Ubah tahun dan bulan berjalan			

Nama Use Case: Tampil Laporan Keuangan	ID: 1.6	Tingkat Penting: Tinggi
Aktor Utama: Staff Pembukuan	Tipe Use Case: Detil	
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajer Pembukuan		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff Pembukuan membuat laporan keuangan yang diperlukan untuk pelaporan kepada pihak manajemen dan eksternal perusahaan.		
Pemicu: Staff Pembukuan memilih menu Buat Laporan Keuangan Tipe: Internal		
Relationships: Association: Staff Pembukuan Include: Extend: Generalization:		
Aliran Kejadian Normal: 1. Staff Pembukuan memilih jenis laporan keuangan yang ingin ditampilkan 2. Staff Pembukuan mencetak laporan keuangan 3. Staff Pembukuan menutup form laporan keuangan		
SubFlows:		
Alternate/Exceptional Flows:		

Nama Use Case: Edit Aktiva Tetap	ID: 1.7	Tingkat Penting: Tinggi
Aktor Utama: Staff Pembukuan	Tipe Use Case: Detil	
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajer Pembukuan		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff Pembukuan membuat daftar Aktiva Tetap yang digunakan.		
Pemicu: Staff Pembukuan memilih menu Edit Aktiva Tetap Tipe: Internal		
Relationships: Association: Staff Pembukuan Include: Extend: Generalization:		
Aliran Kejadian Normal: 1. Staff Pembukuan memasukkan informasi umum Aktiva Tetap (ID Aktiva Tetap, deskripsi, tanggal akuisisi, tanggal pemakaian, estimasi masa hidup, metode penyusutan) 2. Staff Pembukuan memilih akun yang digunakan oleh Aktiva Tetap yang bersangkutan (akun aktiva, akun penyusutan dan akun akumulasi penyusutan) 3. Staff Pembukuan menekan tombol OK		

SubFlows: S.1. Staff Pembukuan mengisi data pengeluaran lainnya terkait dengan aktiva tetap
Alternate/Exceptional Flows: 3.a Staff Pembukuan membatalkan proses Edit Aktiva Tetap

Nama Use Case:	Simulasi Penyusutan	ID: 1.8	Tingkat Penting: Tinggi
Aktor Utama: Staff Pembukuan		Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajer Pembukuan			
Penjelasan Singkat:	Use Case ini menjelaskan bagaimana Staff Pembukuan menampilkan Simulasi Penyusutan Aktiva Tetap yang dimiliki.		
Pemicu: Staff Pembukuan memilih menu Simulasi Penyusutan Tipe: Internal			
Relationships: Association: Staff Pembukuan Include: Extend: Generalization:			
Aliran Kejadian Normal: 1. Staff Pembukuan mengisi filter tanggal simulasi 2. Staff Pembukuan memilih tipe Aktiva Tetap yang akan disimulasikan 3. Staff Pembukuan menekan tombol OK			
SubFlows:			
Alternate/Exceptional Flows: 3.a Staff Pembukuan membatalkan proses Simulasi Penyusutan			

Nama Use Case:	Edit Departemen	ID: 2.1	Tingkat Penting: Tinggi
Aktor Utama: Staff HRD		Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajemen Perusahaan			
Penjelasan Singkat:	Use Case ini menjelaskan bagaimana Staff HRD membuat daftar Departemen yang ada di organisasi atau perusahaan.		
Pemicu: Staff HRD memilih menu Edit Departemen Tipe: Internal			
Relationships: Association: Staff HRD Include: Extend: Generalization:			

Aliran Kejadian Normal: 1. Staff HRD memasukkan informasi tentang Departemen(id, nama departemen, deskripsi) 2. Staff HRD memilih apakah Departemen merupakan sub dari departemen yang lain atau tidak 3. Staff Penjualan menekan tombol OK
SubFlows: S.1. Staff HRD memilih Departemen yang akan menjadi *parent* dari Departemen yang akan dibuat ini
Alternate/Exceptional Flows: 3.a Staff HRD membatalkan proses Edit Departemen

Nama Use Case: Edit Tunjangan	ID: 2.2	Tingkat Penting: Tinggi
Aktor Utama: Staff HRD	Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajemen Perusahaan		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff HRD membuat Daftar Tunjangan untuk para pegawai perusahaan.		
Pemicu: Staff HRD memilih menu Edit Tunjangan Tipe: Internal		
Relationships: Association: Staff HRD Include: Extend: Generalization:		
Aliran Kejadian Normal: 1. Staff HRD memasukkan informasi tentang Tunjangan (id, nama tunjangan, deskripsi, nominal tunjangan, status) 2. Staff HRD menekan tombol OK		
SubFlows:		
Alternate/Exceptional Flows: 2.a Staff HRD membatalkan proses Edit Tunjangan		

Nama Use Case: Edit Potongan	ID:2.3	Tingkat Penting: Tinggi
Aktor Utama: Staff HRD	Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajemen Perusahaan		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff HRD membuat Daftar Potongan untuk para pegawai perusahaan.		
Pemicu: Staff HRD memilih menu Edit Potongan Tipe: Internal		

Relationships: Association: Staff HRD Include: Extend: Generalization:
Aliran Kejadian Normal: 1. Staff HRD memasukkan informasi tentang Potongan (id, nama potongan, deskripsi, nominal potongan, status) 2. Staff HRD menekan tombol OK
SubFlows:
Alternate/Exceptional Flows: 2.a Staff HRD membatalkan proses Edit Potongan

Nama Use Case: Edit Pegawai	ID:2.4	Tingkat Penting: Tinggi
Aktor Utama: Staff HRD	Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajemen Perusahaan		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff HRD membuat daftar Pegawai yang ada di organisasi atau perusahaan.		
Pemicu: Staff HRD memilih menu Edit Pegawai Tipe: Internal		
Relationships: Association: Staff HRD Include: Extend: Generalization:		
Aliran Kejadian Normal: 1. Staff HRD memasukkan informasi umum tentang Pegawai (data pribadi pegawai, data keluarga) 2. Staff HRD mengisi informasi terkait perusahaan (id pegawai, id user, departemen, jabatan, jam kerja email) 3. Staff HRD mengisi informasi gaji (gaji dasar, tunjangan dan potongan yang diterima, nomor rekening) 4. Staff HRD menekan tombol OK		
SubFlows: S.1. Staff HRD melakukan iterasi dalam proses mengeset tunjangan dan potongan untuk pegawai yang bersangkutan.		
Alternate/Exceptional Flows: 4.a Staff HRD membatalkan proses Edit Pegawai		

Nama Use Case: Buat Daftar Payroll	ID:2.5	Tingkat Penting: Tinggi
Aktor Utama: Staff HRD	Tipe Use Case: Detil	
Stakeholders dan Pihak lain yang berkepentingan: 1. Manajemen Perusahaan		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Staff HRD membuat daftar payroll untuk pegawai yang ada di organisasi atau perusahaan.		
Pemicu: Staff HRD memilih menu Buat Daftar Payroll Tipe: Internal		
Relationships: Association: Staff HRD Include: Buat Daftar Transfer Gaji Extend: Generalization:		
Aliran Kejadian Normal: 1. Staff HRD memasukkan bulan dan tahun untuk daftar payroll yang akan dibuat 2. Staff HRD mengisi jam kerja tiap pegawai beserta lembur dan cuti potong gaji jika ada 3. Staff HRD menekan tombol Save (menyimpan jam kerja dan jam lembur untuk bulan yang bersangkutan) 4. Sistem meng-*generate* daftar payroll berdasarkan informasi jam kerja dan data pegawai 5. Staff HRD mengecek daftar dan melakukan koreksi jika perlu 6. Staff HRD menekan tombol Save (menyimpan daftar payroll untuk bulan yang bersangkutan) 7. Staff HRD keluar dari form		
SubFlows: S.1. Sistem melakukan penghitungan jumlah jam lembur S.2. Sistem melakukan penghitungan nominal pajak penghasilan yang harus dibayarkan S.3. Sistem membuat daftar transfer gaji, daftar pajak penghasilan S.4. Sistem membuat surat permohonan dana untuk pembayaran gaji pegawai dan pajak penghasilan kepada Staff Keuangan		
Alternate/Exceptional Flows: 3.a Staff HRD membatalkan proses penyimpanan jam kerja pegawai 6.a Staff HRD membatalkan proses penyimpanan daftar payroll pegawai 7.a Staff HRD mencetak daftar payroll 7.b Staff HRD mencetak daftar transfer gaji beserta pajak penghasilan tiap pegawai yang harus dibayarkan 7.c Staff HRD mencetak surat permohonan dana untuk pembayaran gaji dan pajak penghasilan kepada Staff Keuangan		

Nama Use Case: Edit Pengguna	ID:3.1	Tingkat Penting: Tinggi
Aktor Utama: Administrator	Tipe Use Case: Detil	
Stakeholders dan Pihak lain yang berkepentingan:		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Administrator menambah pengguna baru untuk akses ke sistem.		

Pemicu: Administrator memilih menu Edit Pengguna Tipe: Internal
Relationships: Association: Administrator Include: Extend: Generalization:
Aliran Kejadian Normal: 1. Administrator memasukkan data umum pengguna 2. Administrator menekan tombol OK
SubFlows:
Alternate/Exceptional Flows: 2.a Administrator membatalkan proses dit Pengguna

Nama Use Case: Tambah Hak Akses	ID:3.2	Tingkat Penting: Tinggi
Aktor Utama: Administrator	Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan:		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Administrator membuat daftar hak akses ke sistem.		
Pemicu: Administrator memilih menu Tambah Hak Akses Tipe: Internal		
Relationships: Association: Administrator Include: Extend: Generalization:		
Aliran Kejadian Normal: 1. Administrator memasukkan data umum hak akses 2. Administrator menekan tombol OK		
SubFlows:		
Alternate/Exceptional Flows: 1.a Administrator membatalkan proses Tambah Hak Akses		

Nama Use Case: Assign Hak Akses ke Pengguna	ID:3.3	Tingkat Penting: Tinggi
Aktor Utama: Administrator	Tipe Use Case:	Detil
Stakeholders dan Pihak lain yang berkepentingan:		
Penjelasan Singkat: Use Case ini menjelaskan bagaimana Administrator melakukan assign hak akses ke pengguna.		
Pemicu: Administrator memilih menu Assign Hak Akses Tipe: Internal		
Relationships: Association: Adminstrator Include: Extend: Generalization:		
Aliran Kejadian Normal: 1. Administrator memilih user dari daftar user 2. Administrator memilih menu edit user 3. Administrator mengedit hak akses yang dimiliki oleh user yang bersangkutan 4. Administrator menekan tombol OK 5. Sistem melakukan perubahan pada basis data 6. Administrator keluar dari form		
SubFlows:		
Alternate/Exceptional Flows: 4.a Administrator membatalkan proses edit hak akses bagi user		

LAMPIRAN B

USE CASE DIAGRAM

Diagram *use case* Buku Besar

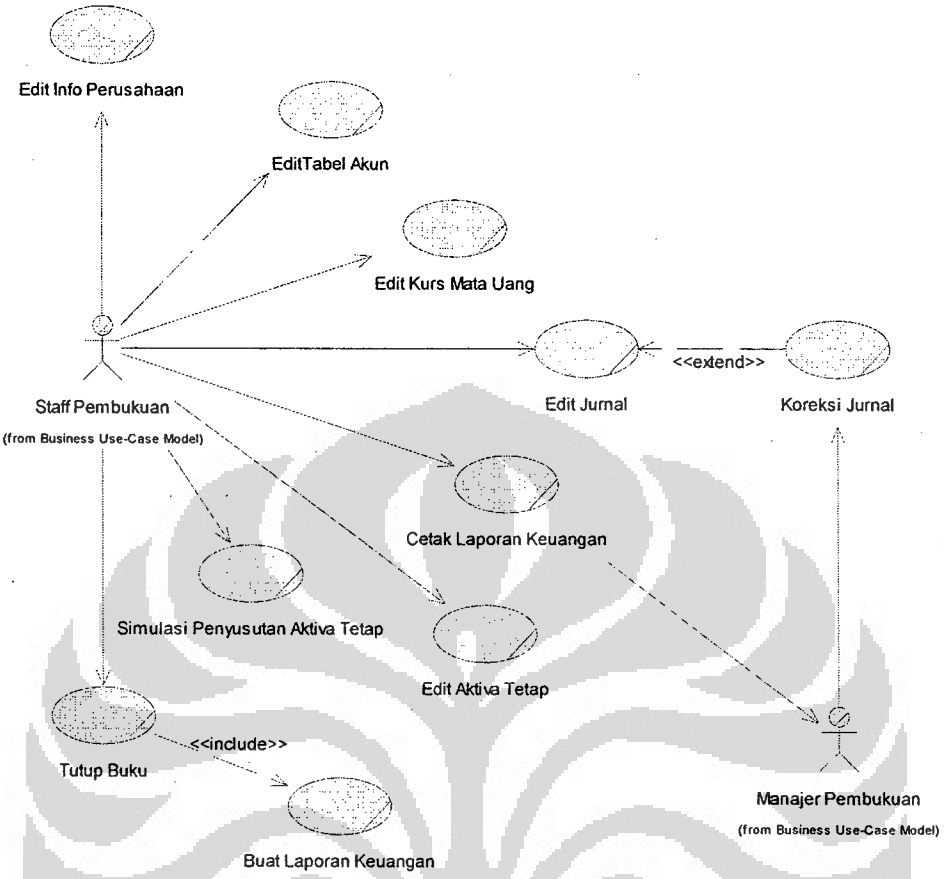


Diagram *use case* Sumber Daya Manusia

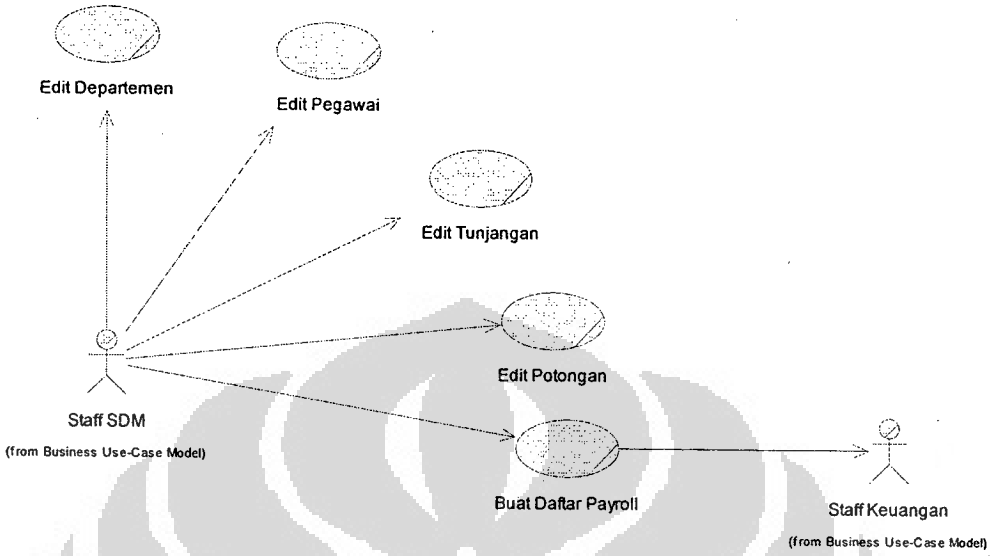
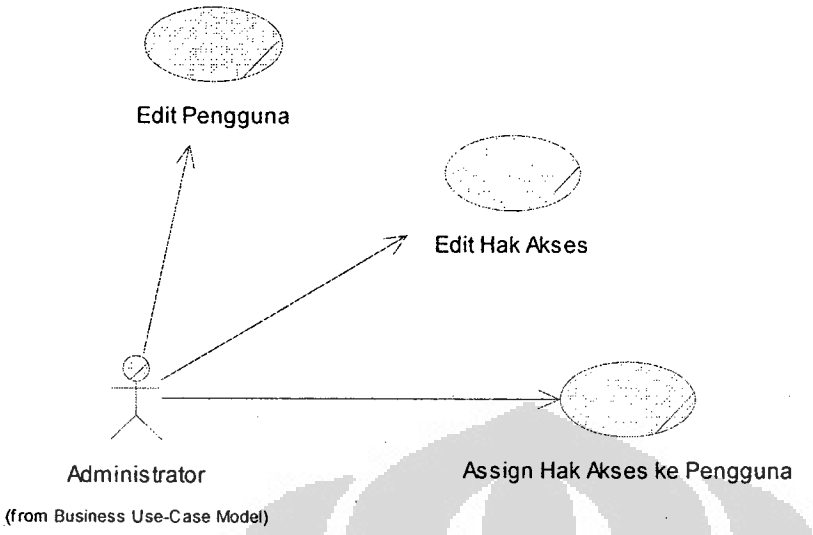


Diagram *use case* Pengaturan Sistem



LAMPIRAN C

DIAGRAM AKTIFITAS

Diagram Aktifitas untuk *use case* Edit Info Perusahaan

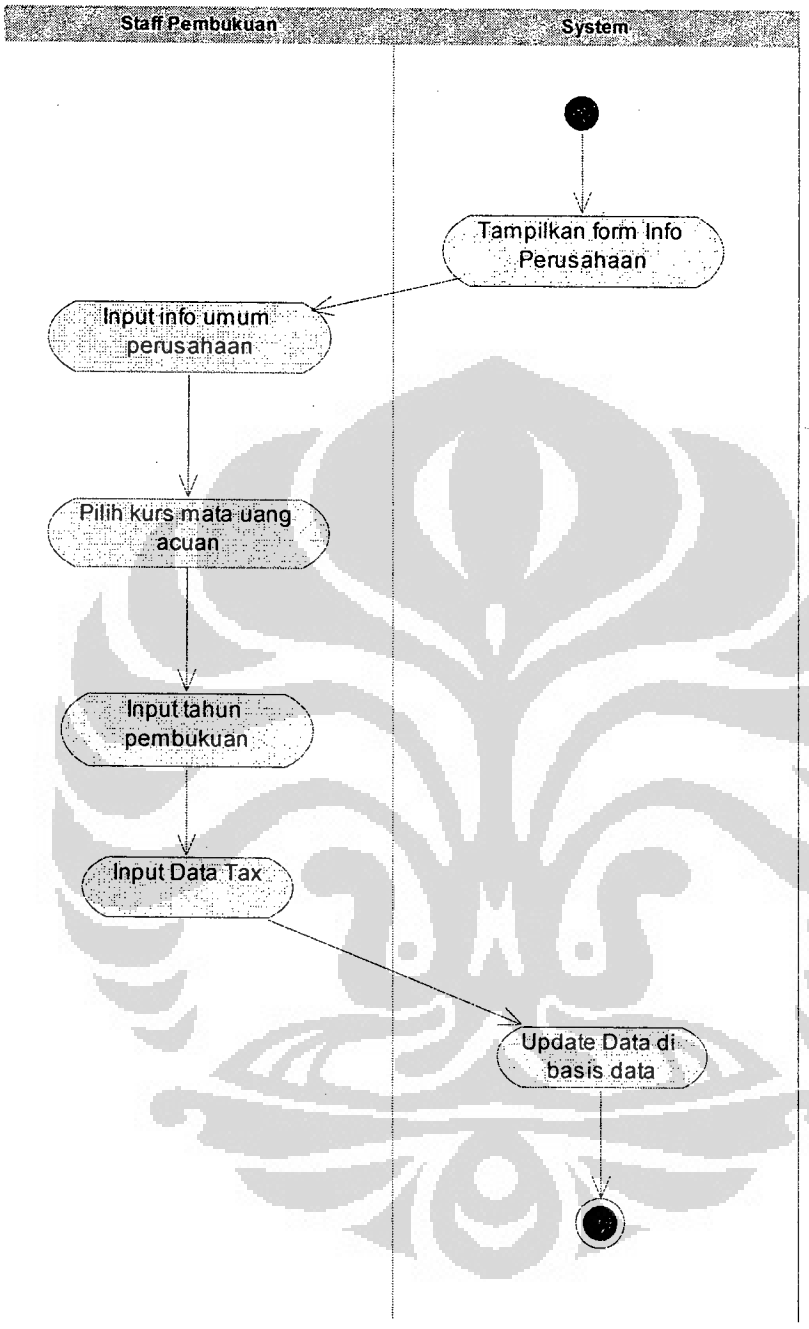


Diagram Aktifitas untuk *use case* Edit Tabel Akun

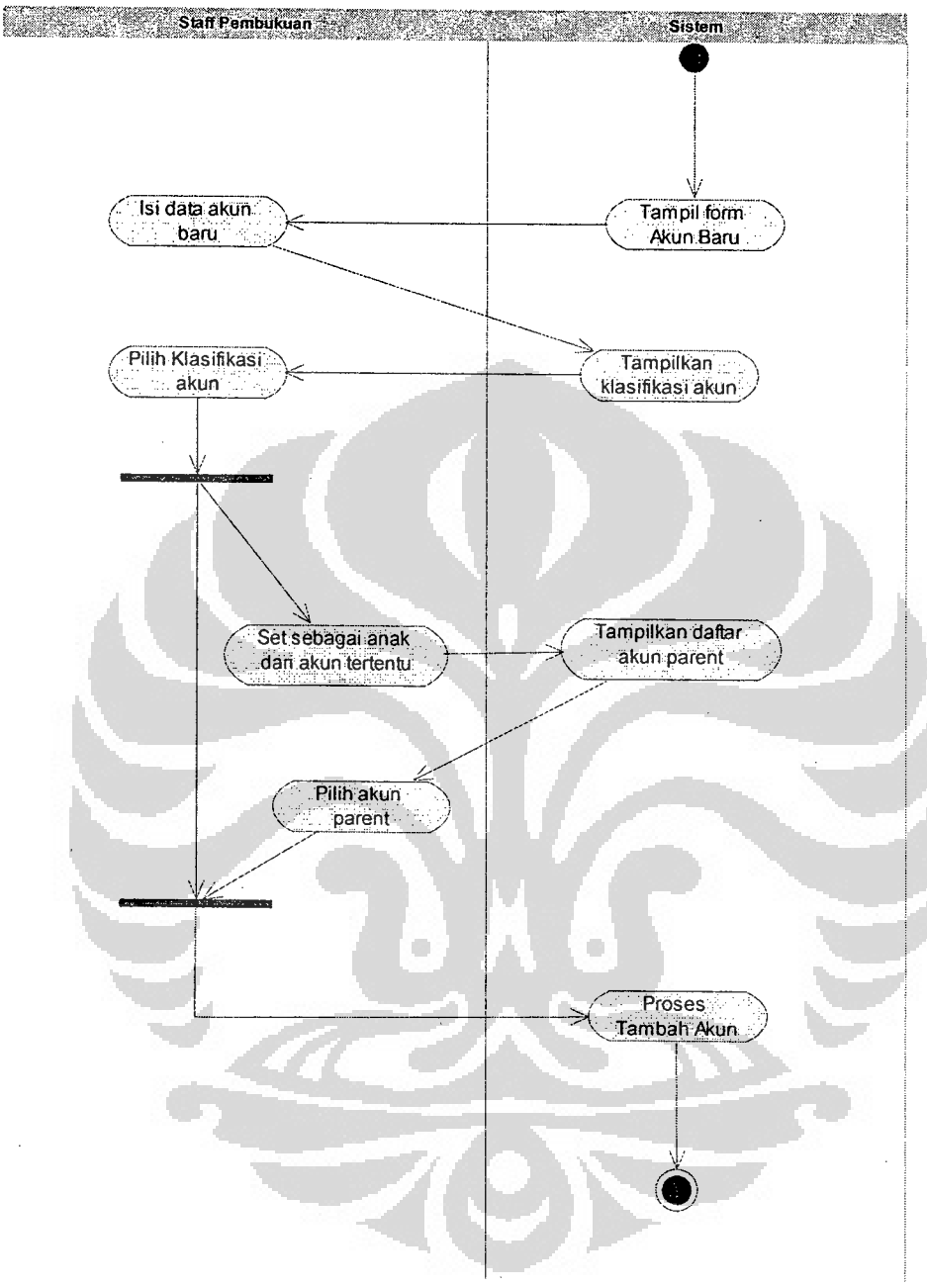


Diagram Aktifitas untuk *use case* Edit Kurs Mata Uang

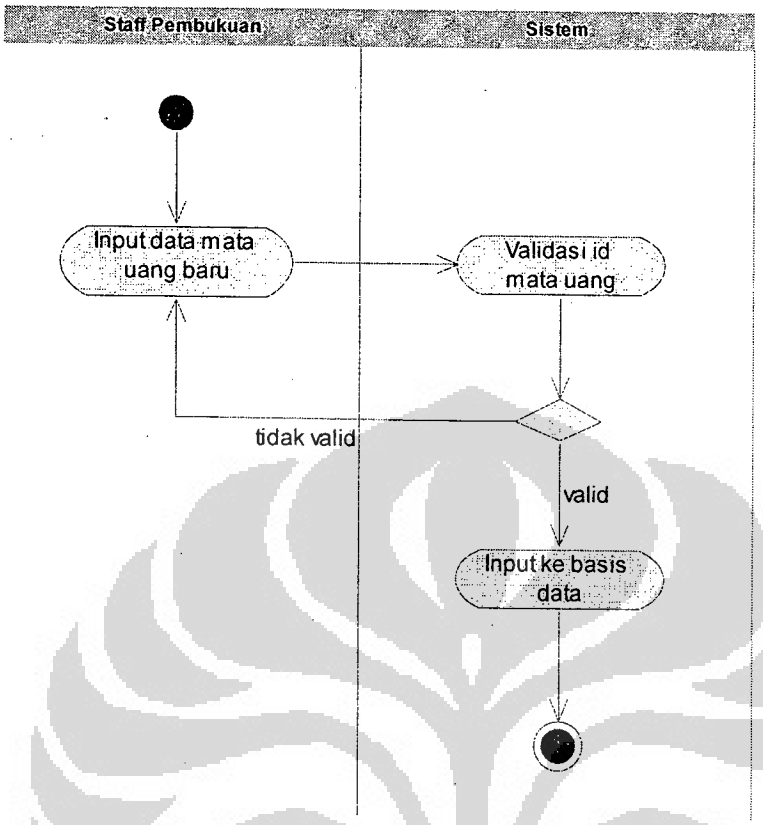


Diagram Aktifitas untuk *use case* Edit Jurnal

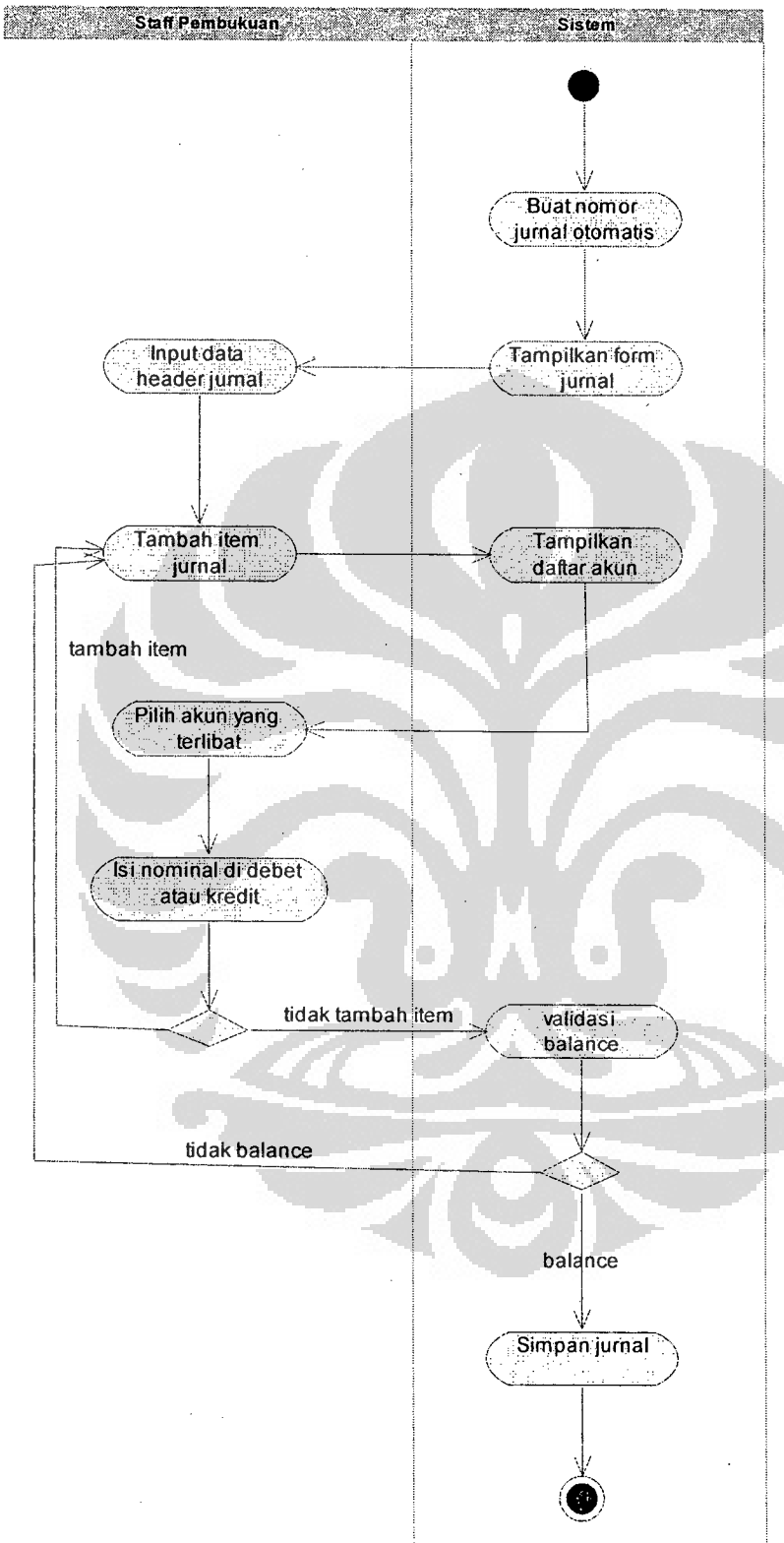


Diagram Aktifitas untuk *use case* Koreksi Jurnal

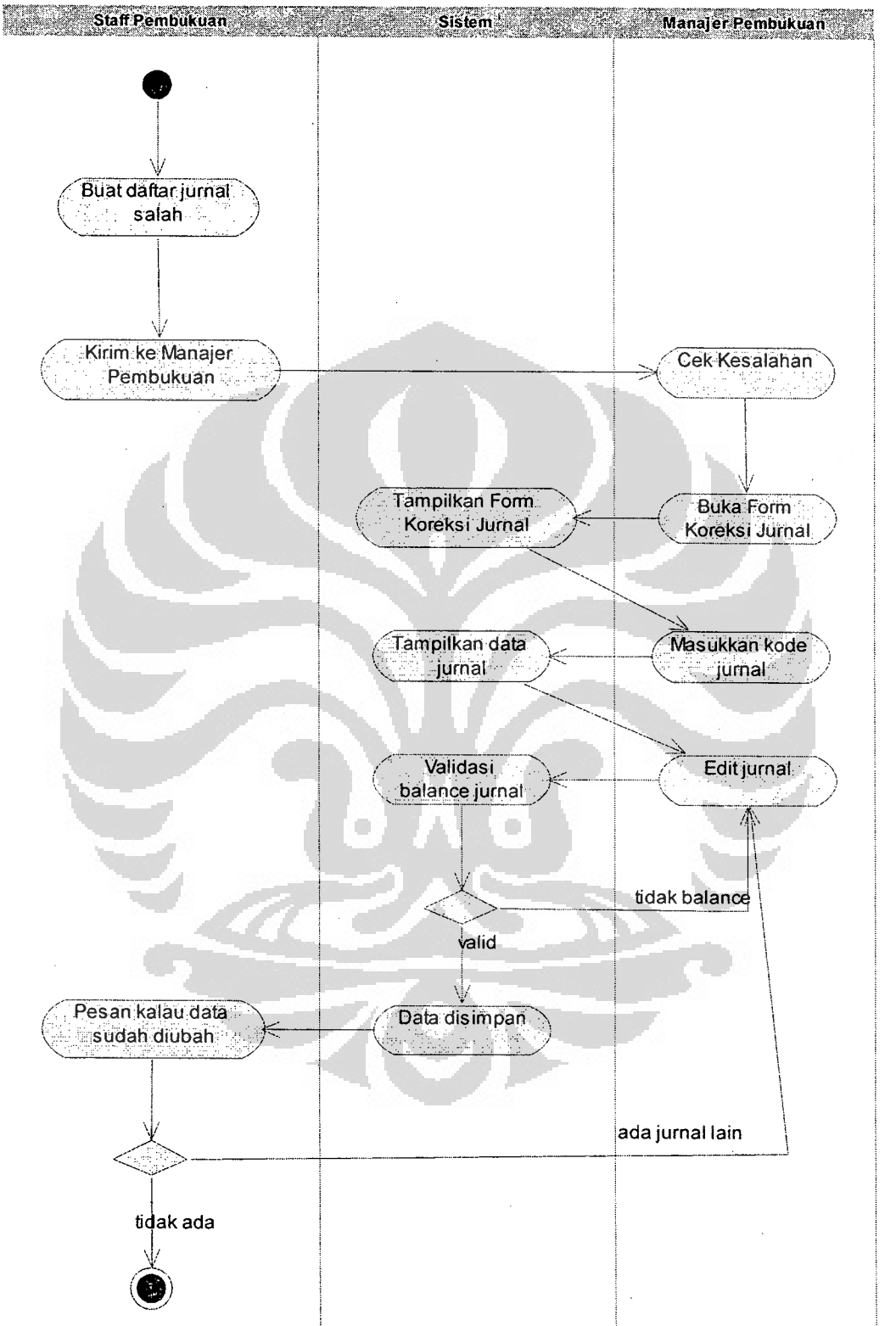


Diagram Aktifitas untuk *use case* Edit Aktiva Tetap

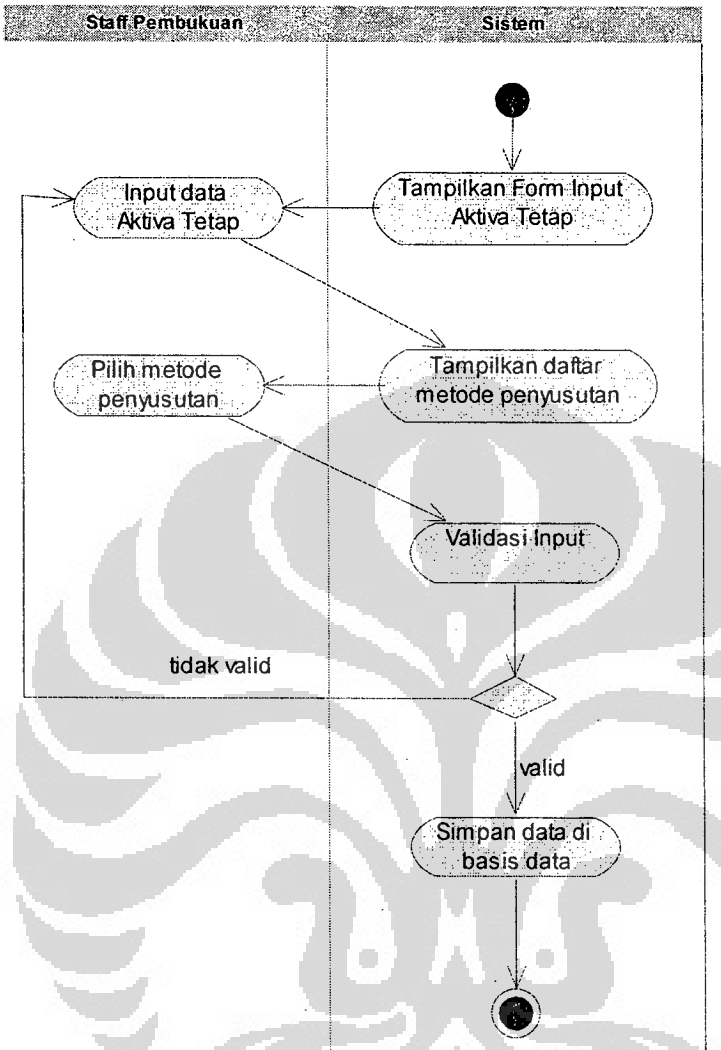


Diagram Aktifitas untuk *use case* Simulasi Penyusutan Aktiva Tetap

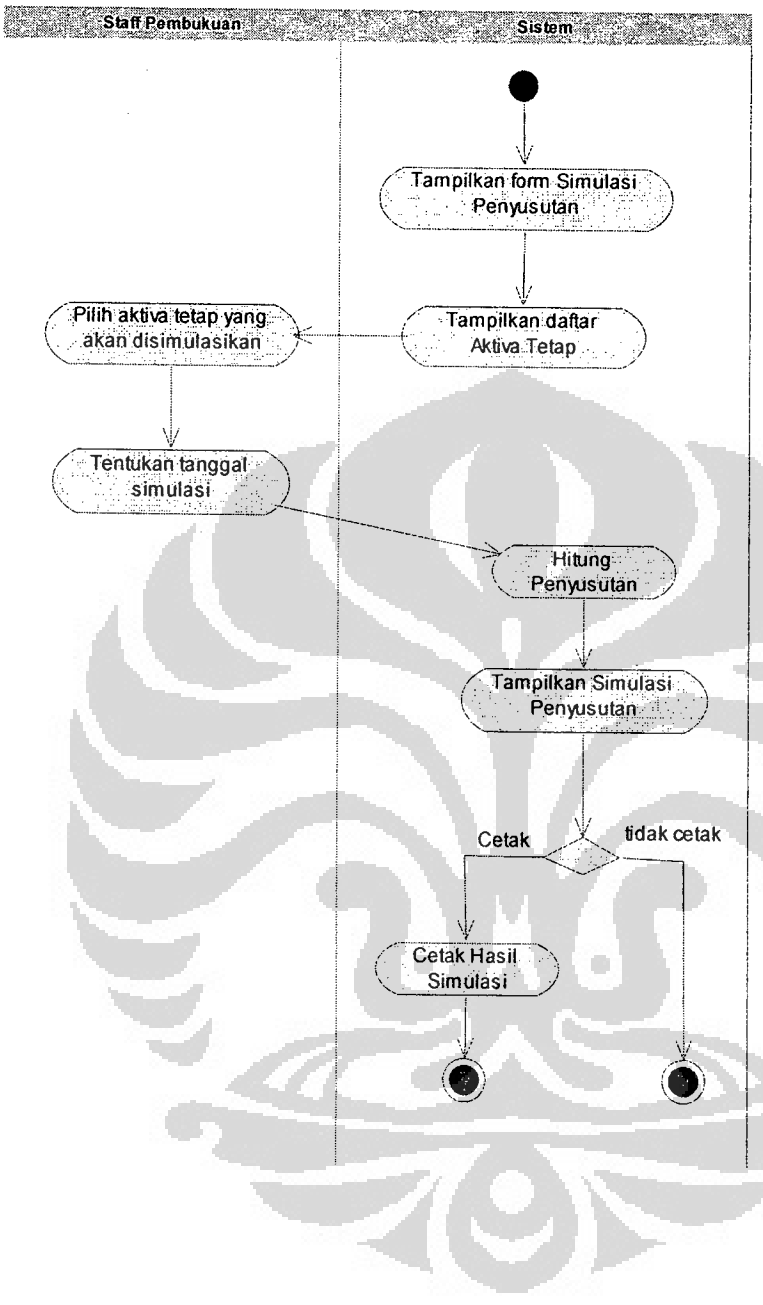


Diagram Aktifitas untuk *use case* Tutup Buku

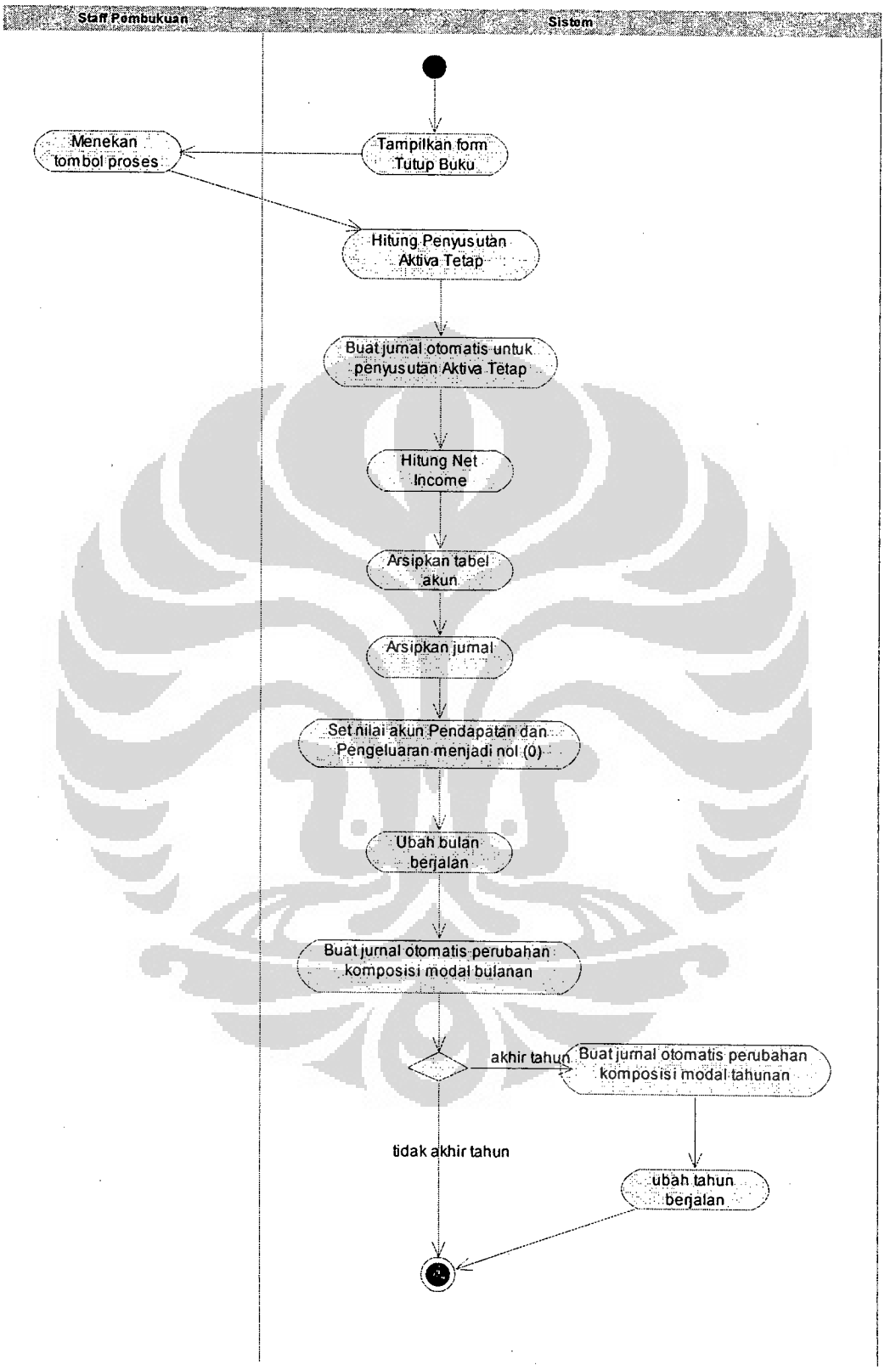


Diagram Aktifitas untuk *use case* Buat Laporan Keuangan

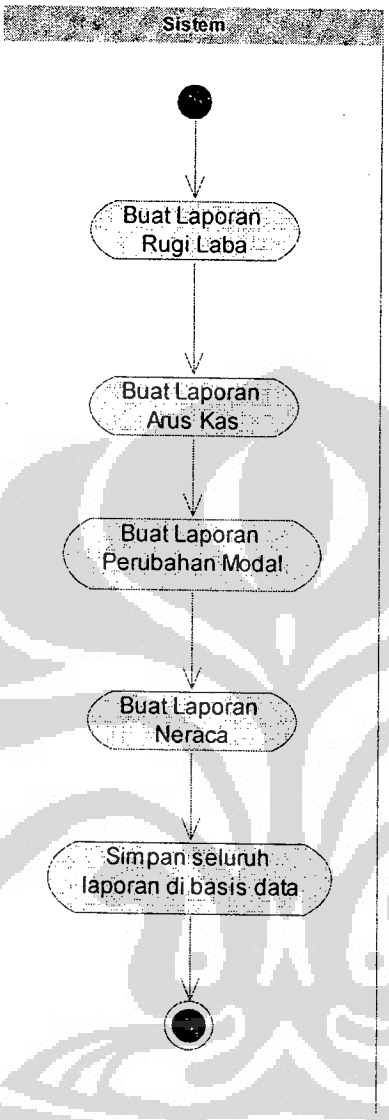


Diagram Aktifitas untuk *use case* Cetak Laporan Keuangan

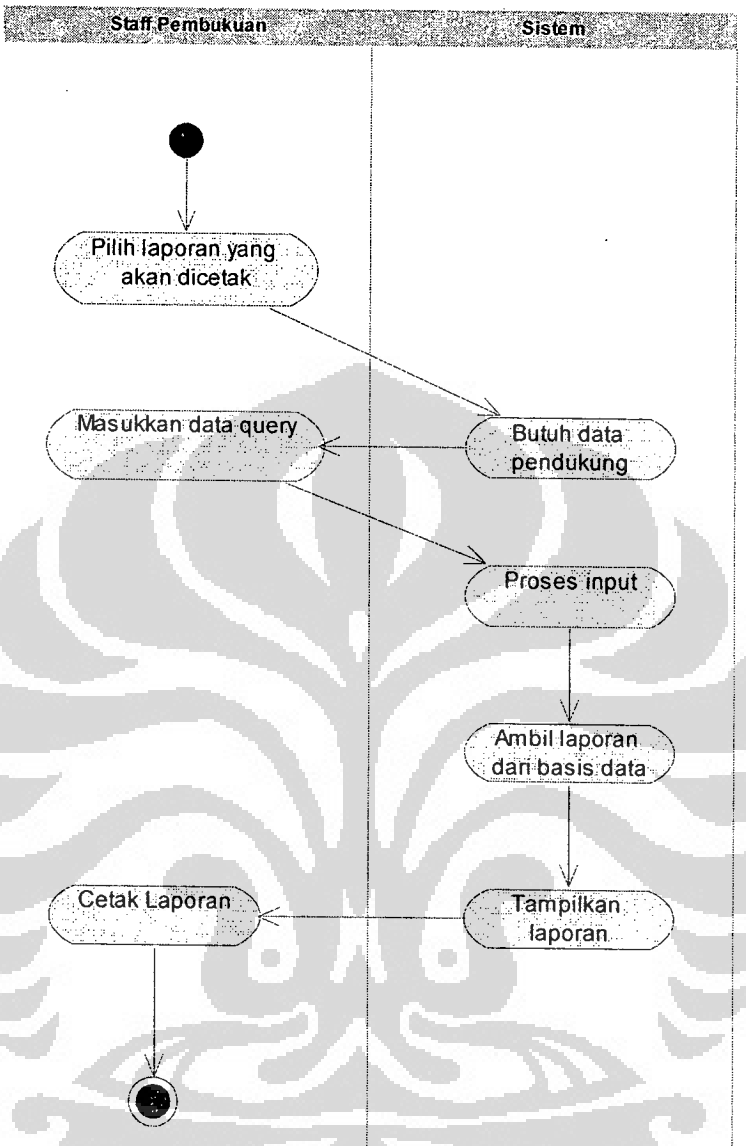


Diagram Aktifitas untuk *use case* Edit Departemen

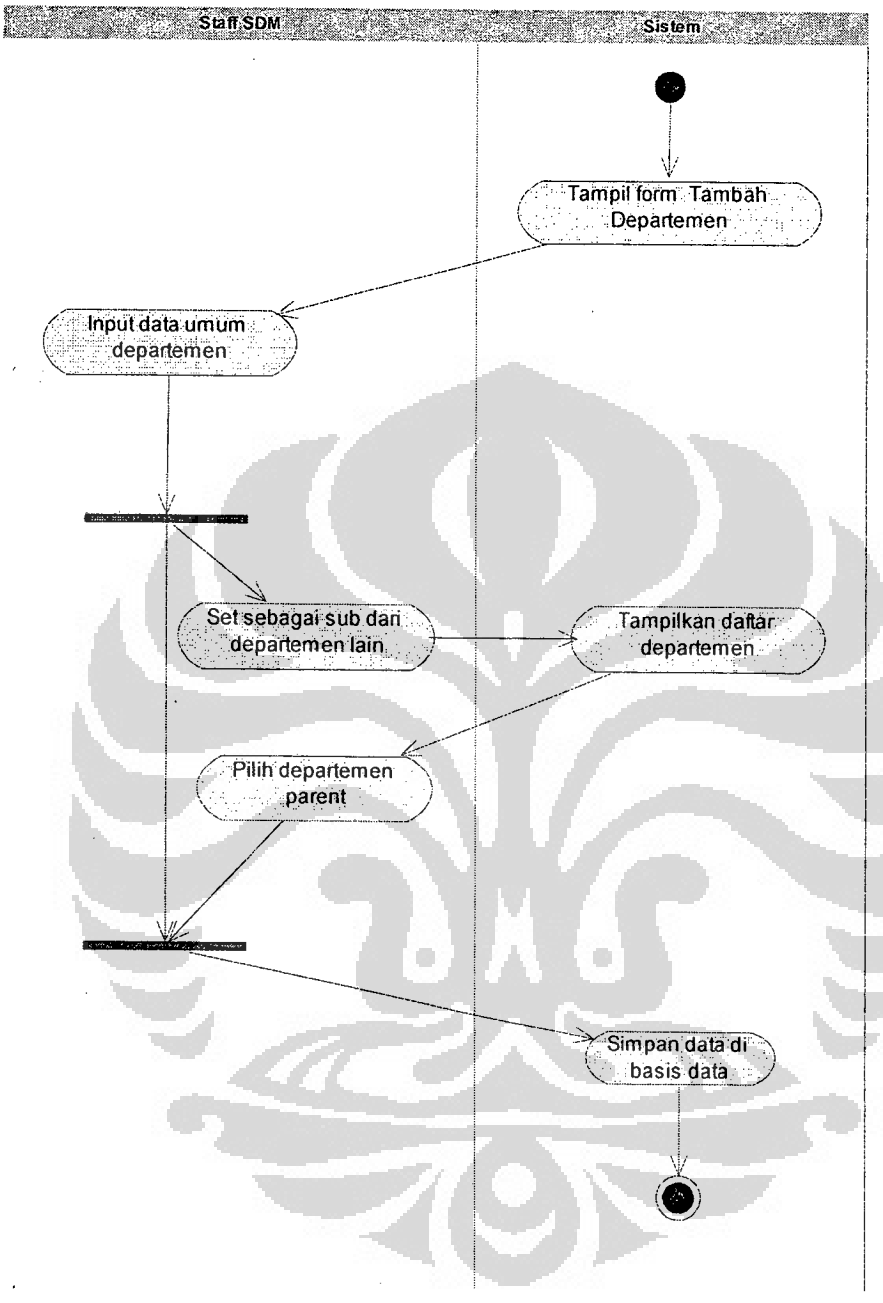


Diagram Aktifitas untuk *use case* Edit Pegawai

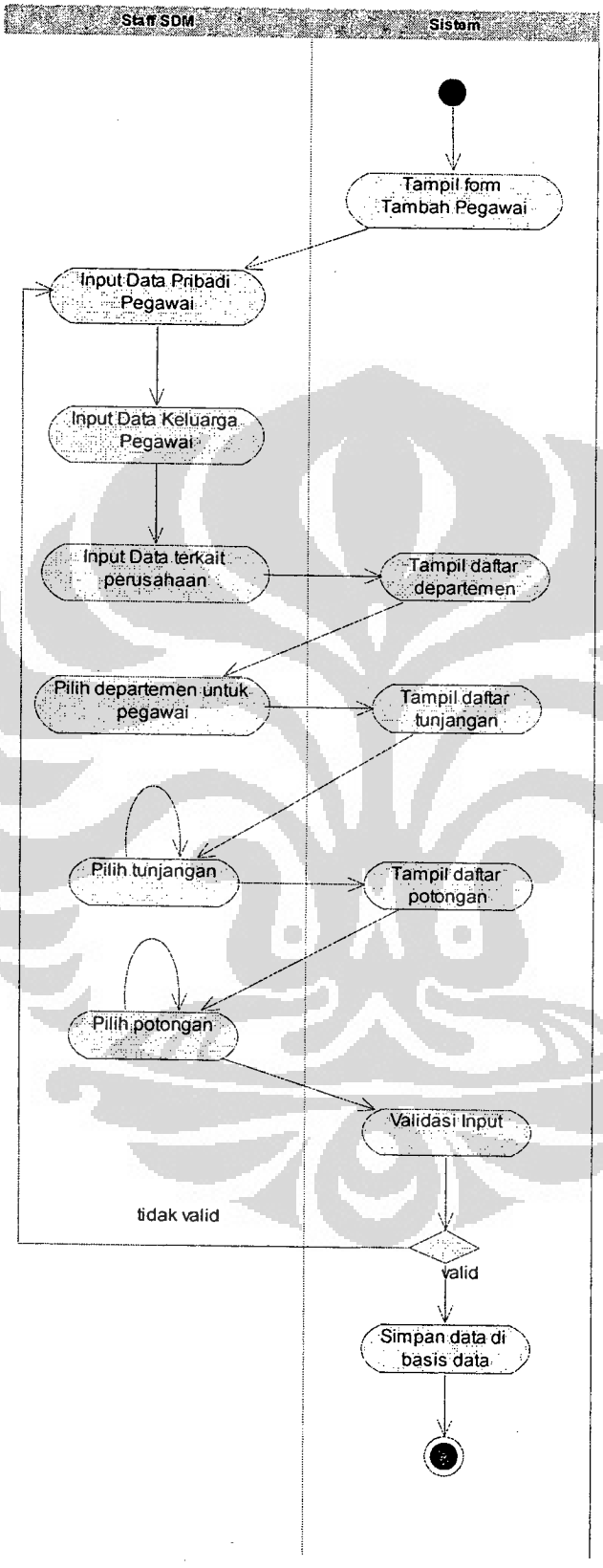


Diagram Aktifitas untuk *use case* Edit Tunjangan

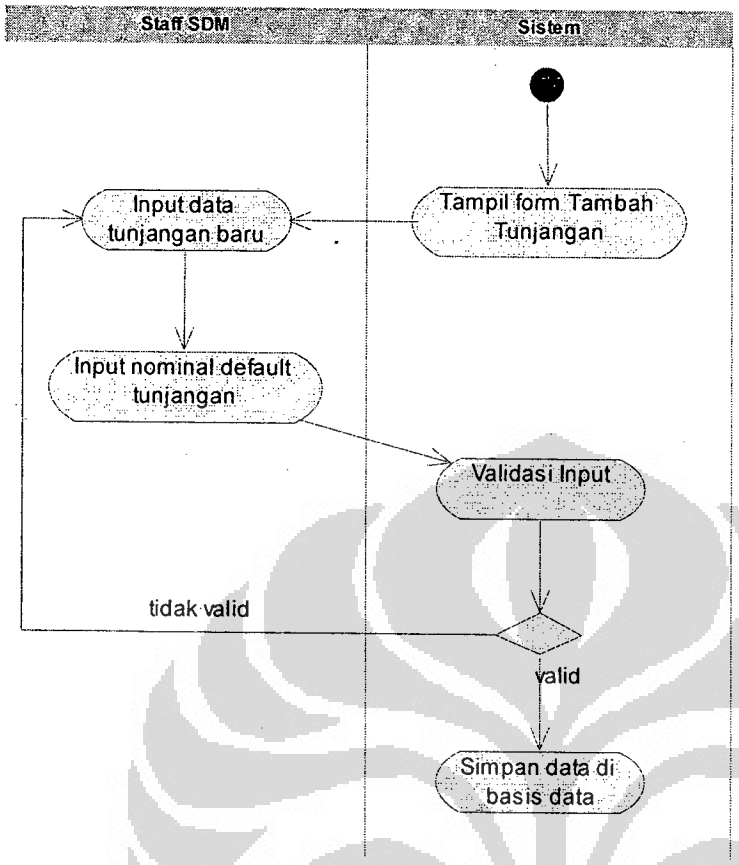


Diagram Aktifitas untuk *use case* Edit Potongan

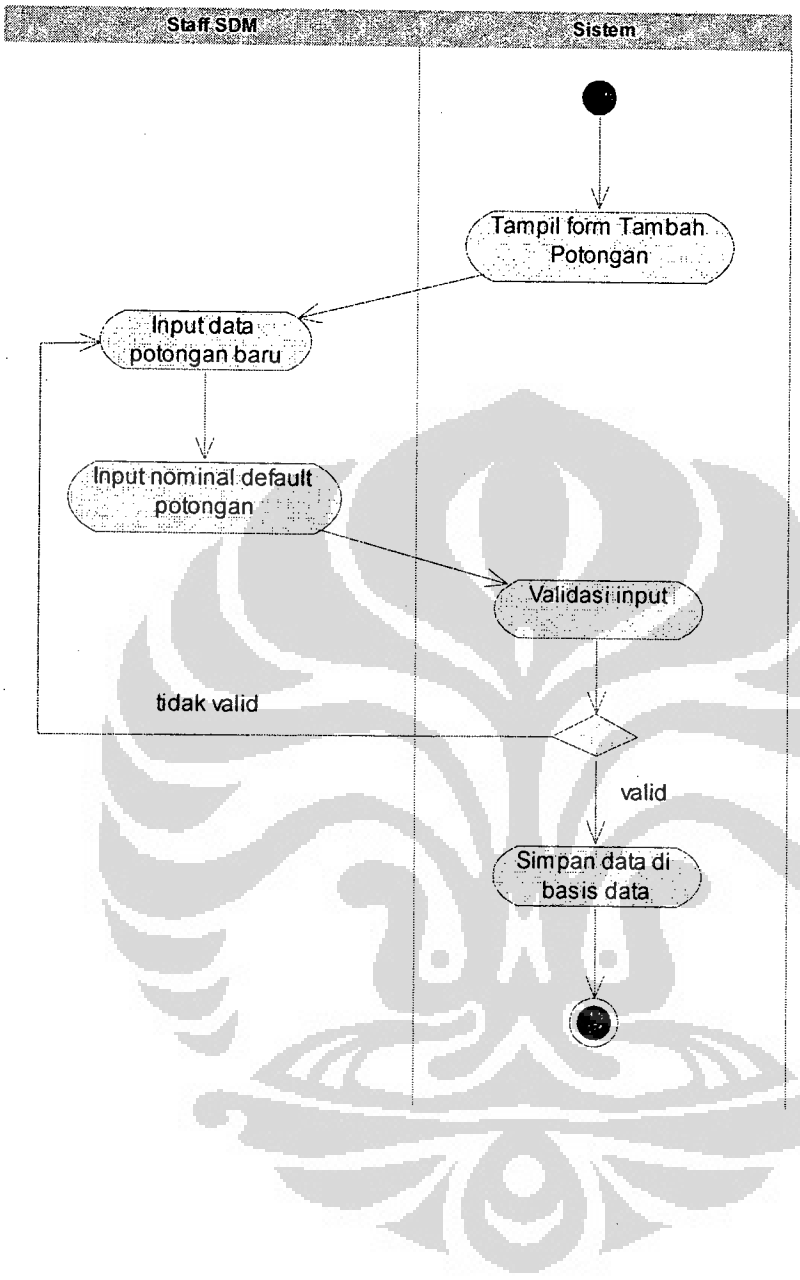


Diagram Aktifitas untuk *use case* Buat Daftar Payroll

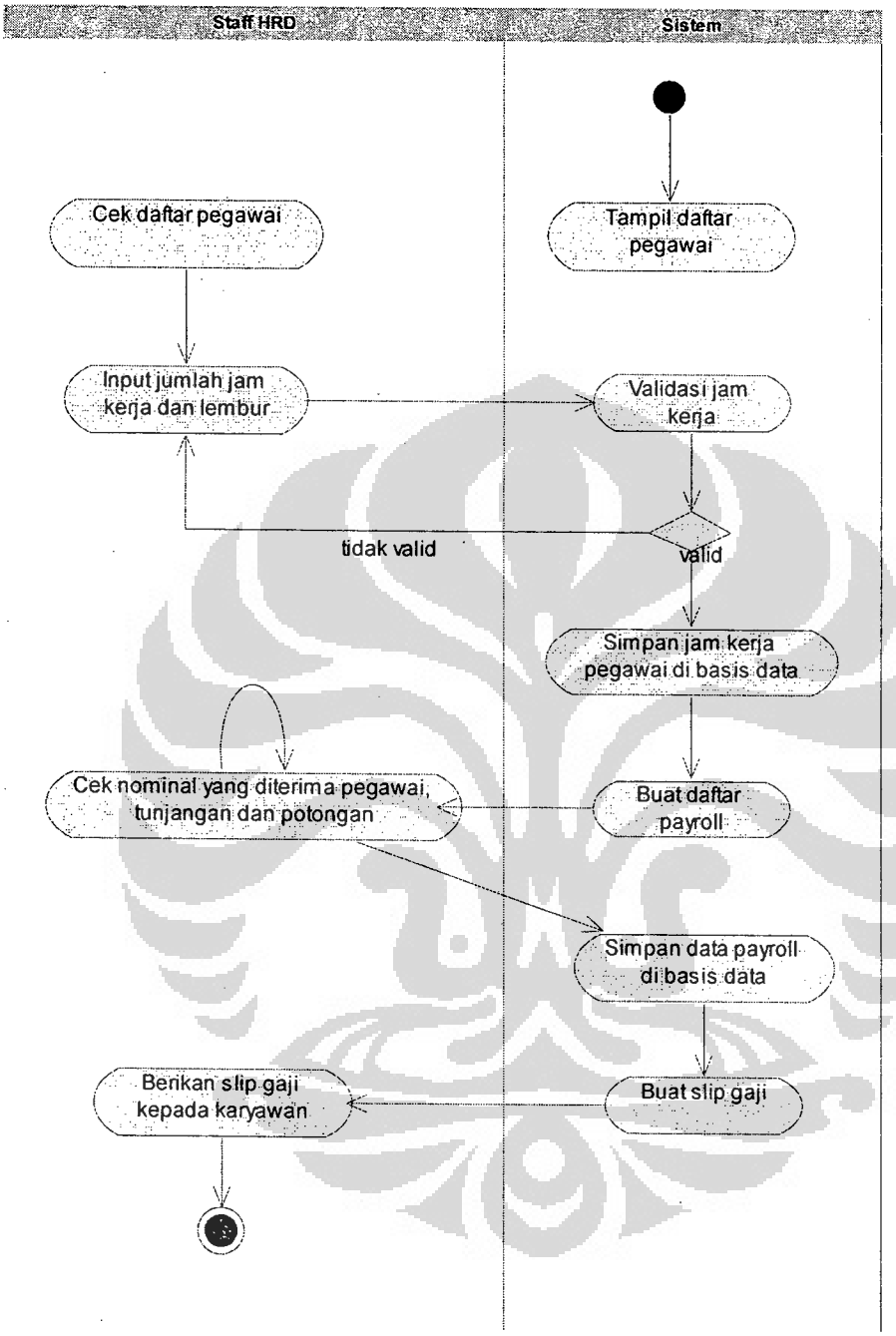


Diagram Aktifitas untuk *use case* Edit Pengguna

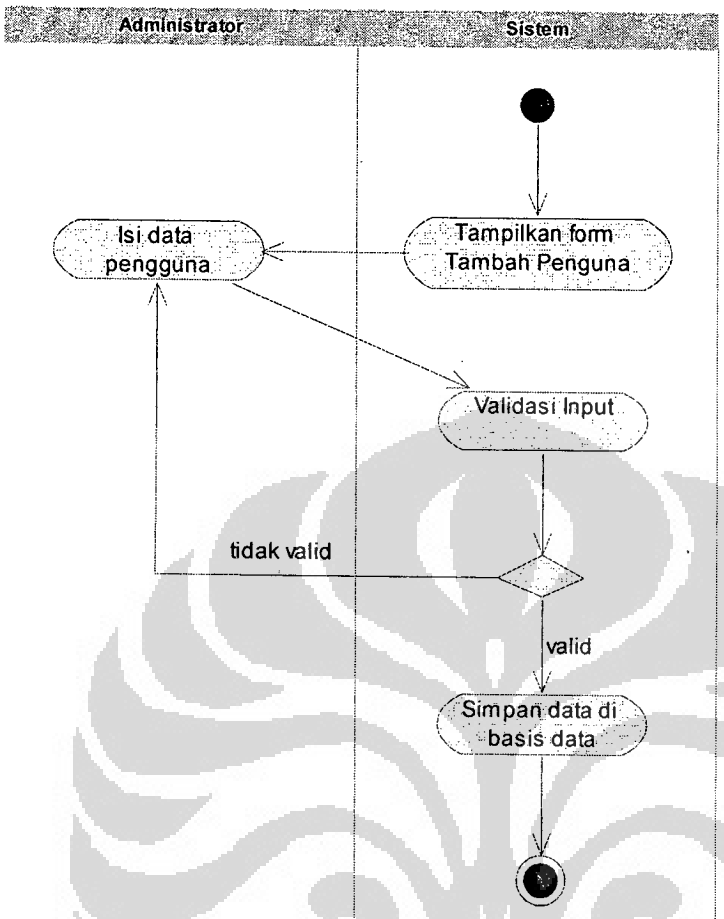


Diagram Aktifitas untuk *use case* Tambah Hak Akses

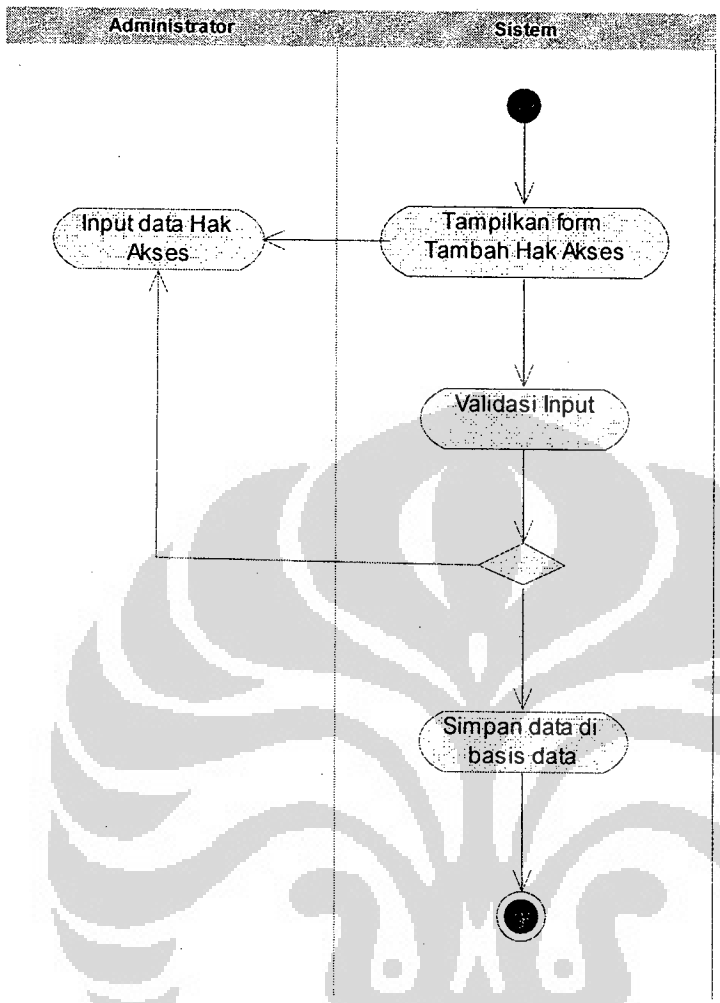


Diagram Aktifitas untuk *use case* Assign Hak Akses ke Pengguna