BAB 2
TINJAUAN PUSTAKA

2.1 Profil Instansi

Profil Lembaga Kemahasiswaan merupakan pembahasan mengenai sejarah, visi dan misi, struktur organisasi serta deskripsi jabatan pada Lemabaga Kemahasiswaan yang akan menjadi tempat dibangunnya sistem ini.

2.1.1 Sejarah Instansi

Lembaga Kemahasiswaan adalah sebuah unit kerja dibawah Kantor Wakil Rektor Bidang Akademik dan Kemahasiswaan yang memiliki peran strategis dalam mengembangkan mahasiswa ITB dengan lingkungan kemahasiswaannya agar menghasilkan alumni yang berkarakter dan berkompeten. Lembaga Kemahasiswaan pada dasarnya mencoba membangun kemahasiswaan ITB melalui beberapa pendekatan, yakni; (1) Pengembangan ko-kurikuler, (2) Pemberdayaan Mahasiswa untuk Masyarakat, (3) Peningkatan Kesejahteraan Mahasiswa, (4) Pendidikan Karakter, dam (5) Mempersiapkan Kapasitas Keprofesian Mahasiswa.

2.1.2 Visi dan Misi

Lembaga Kemahasiswaan Institut Teknologi Bandung memiliki Visi dan Misi sebagai berikut :

1. Visi

   Menjadikan Lembaga Kemahasiswaan sebagai mitra mahasiswa dan ujung tombak Institusi dalam membangun lingkungan kemahasiswaan ITB yang berkarakter, sehingga dapat menjadi inspirasi bagi mahasiswa Indonesia.

2. Misi

   a. Sesuai dengan peran dan fungsi Lembaga Kemahasiswaan ITB, Misi Lembaga Kemahasiswaan adalah merumuskan, mendukung, menjaga dan mengarahkan, agar terwujud lingkungan kemahasiswaan yang berkelas dunia dan aktivitas lain yang berhubungan dengan mahasiswa dan

kemahasiswaan di Institut Teknologi Bandung dapat terlaksana dengan sebaik-baiknya.

b. Dalam rensta 2011-2015 misi tersebut secara khusus diorieentasikan pada upaya menjadikan kemahasiswaan ITB sebagai kemahasiswaan yang berkelas dunia melalui pengembangan lingkungan kemahasiswaan yang dapat melahirkan karaakter mahasiswa sesuai dengan tujuan pendidikan di ITB.

2.1.3 Logo Lembaga Kemahasiswaan ITB

Logo Lembaga Kemahasiswaan ITB sama dengan lambang atau logo Institut Teknologi Bandung dikarenakan Lembaga Kemahasiswaan ITB adalah lembaga dibawah naungan Institut Teknologi Bandung. Lembaga Kemhasiswaan ITB tidak memiliki logo sendiri melaikan Lembaga Kemahasiswaan ITB menggunakan logo dari Institut Teknologi Bandung. Logo Lembaga Kemahasiswaan ITB dapat terlihat pada Gambar 2.1



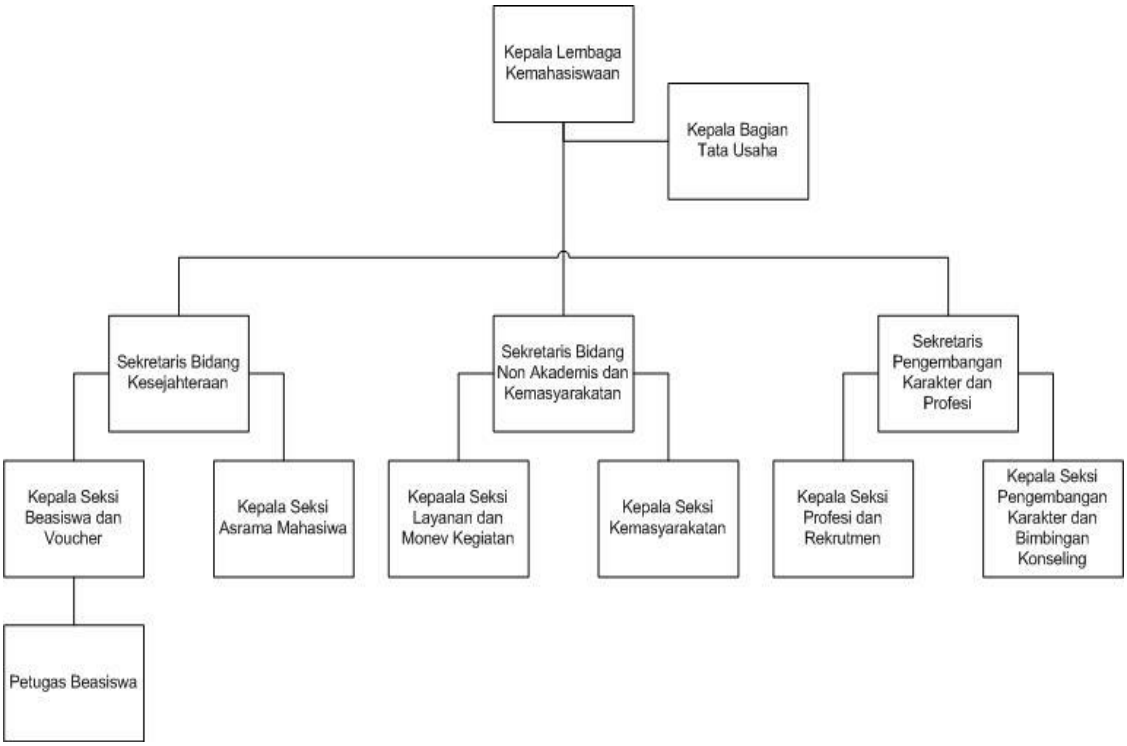
Gambar 2.1 Logo Lembaga Kemahasiswaan ITB

Makna yang terdapat dalam logo ini ialah :

1. Cawan melambangkan ilmu yang tiada habis-habisnya yang dihimpun pada diri Ganesha.
2. Gading yang dipatahkan menunjukan pengorbanan diri untuk memecahkan maslah-masalah yang merintangi kemajuan ilmiah.
3. Tasbih melambangkan kebijaksanaan.
4. Selendang sebagai tanda kesucian.

2.1.4 Struktur Organisasi

Sebuah organisasi yang baik akan terbentuk apabila setiap anggota organisasi mengetahui tugas , wewenang, tanggung jawab serta hubungan komunikasi antar bagian pada struktur organisasi perusahaan. Pada badan usaha ini memiliki struktur organisasi yang bisa dikatakan cukup sehingga dapat menunjang dalam melaksanakan fungsi secara efektif dan efisien. Struktur organisasi yang ada pada Lembaga Kemahasiswaan ITB dapat dilihat pada Gambar 2.2.



Gambar 2.2 Struktur Organisasi Lembaga Kemahasiswaan ITB.

2.1.5 Deskripsi Pekerjaan

Deskripsi pekerjaan digunakan untuk mengetahui tugas , wewenang, tanggung jawab dari masing-masing bagian. Deskripsi pekerjaan dari struktur organisasi Lembaga Kemahasiswaan yang terdapat pada Gambar 2.1 adalah :

1. Kepala Lembaga Kemahasiswaan
   a. Tugas Pokok

      Memimpin, mengkoordinasikan, dan memotivasi serta mengawasi ruang lingkup tugas pokok Lembaga Kemahasiswaan ITB.

   b. Fungsi

      Kepanjangan tugas dari Wakil Rektor Bidang Akademik dan Kemahasiswaan untuk menangani seluruh kegiatan mahasiswa yang berkaitan dengan akademik.
2. Kepala Bagian Tata Usaha.

   Kepala Bagian Tata Usaha mempunyai tugas membantu Kepala Lembaga Kemahasiswaan dibidang ketatausahaan, kepegawaian, keuangan, perlengkapan umum serta perencanaan dan pelaporan dalam administrator manajemen Lembaga Kemahasiswaan.
3. Sekretariat Bidang Kesejahteraan

   Sekretari Bidang Kesejahteraan mempunyai tugas membantu Kepala Lembaga Kemhasiswaan di bidang kesejahteraan mengenai beasiswa dan asrama mahasiswa.
4. Sekretaris Bidang Non Kurikuler dan Kemasyarakatan

   Sekretaris Bidang Non Kurikuler dan Kemasyarakatan mempunyai tugas membantu Kepala Lembaga Kemahasiswaan di bidang layanan dan monev kegiatan serta kemasyarakatan.
5. Sekretaris Bidang Pengembangan Karakter dan Profesi

   Sekretaris Bidang Pengembangan Karakter dan Profesi mempunyai tugas membantu Kepala Lembaga Kemahasiswaan di bidang profesi dan karakte serta pengembangan karakter dan bimbingan konseling.
6. Kepala Seksi Beasiswa dan Voucher

   Kepala Seksi Beasiswa dan Voucher mempunyai tugas menangani secara teknis penyaluran beasiswa yang diberikan kepada mahasiswa.
7. Kepala Seksi Asrama Mahasiswa

   Kepala Seksi Asrama Mahasiswa mempunyai tugas menangani secara teknis bagian asrama yang digunakan mahasiswa

8. Kepala Seksi Layanan dan Monev Kegiatan

   Kepala Seksi Layanan dan Monev Kegiatan mempunyai tugas menangani segala kegiatan yang diselenggarakan mahasiswa.

9. Kepala Seksi Kemasyarakatan

   Kepala Seksi Kemasyarakatan mempunyai tugas secara teknis mengangi kemasyarakatan untuk mahasiswa.

10. Kepala Profesi dan Rekrutmen

    Kepala Profesi dan Rekrutmen mempunyai tugas menangani secara teknis untuk profesi dan rekrutmen mahasiswa bertujuan untuk menjadikan mahasiswa menjadi SDM yang lebih berkualitas.

11. Kepala Seksi Pengembangan Karakter dan Bimbingan Konseling

    Kepala Seksi Pengembangan Karakter dan Bimbingan Konseling mempunyai tugas menangani pengambangan karakter mahasiswa dan sarana bimbingan konseling bagi mahasiswa.

12. Petugas Beasiswa

    Petugas Beasiswa mempunyai tugas untuk mengolah data mahasiswa pendaftar beasiswa dan data perusahaan sebagai pemberi beasiswa, beserta menyeleksi penerima beasiswa sesuai dengan kriteria perusahaan.

2.2 Landasan Teori

Landasan teori dari penulisan skripsi ini menjelaskan tentang sisitem informasi, sisitem pendukung keputusan, Model *Fuzzy Multiple Attribute Decision Making (FMADM),* Metode *Simple Additive Weighting(SAW) , Unfield Modelling Languange (UML),*Teknologi *Java,* yang menguraikan proses analisis sistem serta mendukung proses perancangan Sstem Pendukung Keputusan Peneriamaan Beasiswa di Institut Teknologi Bandung.

2.2.1 Sistem Informasi

Sistem informasi dapat didefinisikan sebagai suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi, mendukung operasi, bersifat manajerial, dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan [3].

Sistem informasi digunakan untuk mengolah transaksi-transaksi, mengurangi biaya dan menghasilkan pendapatan sebagai salah satu produk dan pelayanan. Komponen dalam sistem informasi ada 3, yaitu :

a. Perangkat Keras (*Hardware*) dan Perangkat Lunak (*Software*) yang berfungsi sebagai mesin (sistem).

b. Manusia (*People*) dan Prosedur (*Procedures*) yang merupakan manusia dan tata cara menggunakan mesin (sistem).

c. Data merupakan jembatan penghubung antara manusia dan mesin (sistem) agar terjadi suatu proses pengolahan data [3].

2.2.2 Sistem Pendukung Keputusan

Sistem Pendukung Keputusan merupakan suatu sistem interaktif yang mendukung keputusan dalam proses pengambilan keputusan melalui alternatif – alternatif yang diperoleh dari hasil pengolahan data, informasi dan rancangan model.

2.2.2.1 Karakteristik Sistem Pendukung Keputusan

Dari pengertian Sistem Pendukung Keputusan maka dapat ditentukan karakteristik antara lain **[5]** :

1. Mendukung proses pengambilan keputusan, menitik beratkan pada *management by perception*
2. Adanya *interface* manusia / mesin dimana manusia (*user*) tetap memegang *control* proses pengambilan keputusan
3. Mendukung pengambilan keputusan untuk membahas masalah terstruktur, semi terstruktur dan tak struktur
4. Memiliki kapasitas dialog untuk memperoleh informasi sesuai dengan kebutuhan
5. Memiliki subsistem – subsistem yang terintegrasi sedemikian rupa sehingga dapat berfungsi sebagai kesatuan item.
6. Membutuhkan struktur data komprehensif yang dapat melayani kebutuhan informasi seluruh tingkatan manajemen

2.2.2.2 Komponen Penyusun Sistem Pendukung Keputusa

Suatu Sistem Pendukung Keputusan (SPK) memiliki tiga subsistem utama yang menentukan kapabilitas teknis sistem pendukung keputusan, antara lain **[5]** :

1. Subsistem Manajemen Basis Data

Subsistem data merupakan bagian yang menyediakan data – data yang dibutuhkan oleh *Base management Subsystem (DBMS).* DBMS sendiri merupakan susbsistem data yang terorganisasi dalam suatu basis data. Data – data yang merupakan suatu Sistem Pendukung Keputusan dapat berasal dari luar lingkungan. Keputusan pada manajemen level atas seringkali harus memanfaatkan data dan informasi yang bersumber dari luar perusahaan.

Kemampuan subsistem data yang diperlukan dalam suatu Sistem Pendukung Keputusan, antara lain :

a. Mampu mengkombinasikan sumber – sumber data yang relevan melalui proses ekstraksi data
b. Mampu menambah dan menghapus secara cepat dan mudah
c. Mampu menangani data *personal* dan *non ofisial*, sehingga *user* dapat bereksperimen dengan berbagai alternatife keputusan
d. Mampu mengolah data yang bervariasi dengan fungsi manajemen data yang luas

2. Subsistem Manajemen Model

Subsistem model dalam Sistem Pendukung Keputusan memungkinkan pengambil keputusan menganalisa secara utuh dengan mengembangkan dan membandingkan alternatife solusi. *Integrasi* model – model dalam Sistem Informasi Manajemen yang berdasarkan *integrasi* data – data dari lapangan menjadi suatu Sistem Pendukung Keputusan.

Kemampuan subsistem model dalam Sistem Pendukung Keputusan antara lain:

1. Mampu menciptakan model – model baru dengan cepat dan mudah.
2. Mampu mengkatalogkan dan mengelola model untuk mendukung semua tingkat pemakai.
3. Mampu menghubungkan model – model dengan basis data melalui

hubungan yang sesuai

4. Mampu mengelola basis model dengan fungsi manajemen yang analog dengan database manajemen

3. Subsistem Dialog

Subsistem dialog merupakan bagian dari Sistem Pendukung Keputusan yang dibangun untuk memenuhi kebutuhan representasi dan mekanisme *control* selama proses analisa dalam Sistem Pendukung Keputusan ditentukan dari kemampuan berinteraksi antara sistem yang terpasang dengan *user.*

Pemakai terminal dan sistem perangkat lunak merupakan komponen – komponen yang terlibat dalam subsistem dialog yang mewujudkan komunikasi antara *user* dengan sistem tersebut. Komponen dialog menampilkan keluaran sistem bagi pemakai dan menerima masukkan dari pemakai ke dalam Sistem Pendukung Keputusan.

Adapun subsistem dialog dibagi menjadi tiga, antara lain :

1. Bahasa Aksi *(The Action Language)*

   Merupakan tindakan – tindakan yang dilakukan *user* dalam usaha untuk membangun komunikasi dengan sistem. Tindakan yang dilakukan oleh *user* untuk menjalankan dan mengontrol sistem tersebut tergantung rancangan sistem yang ada.

2. Bahasa Tampilan *(The Display or Presentation Langauage)*

   Merupakan keluaran yang dihasilkan oleh suatu Sistem Pendukung Keputusan dalam bentuk tampilan – tampilan akan memudahkan *user* untuk mengetahui keluaran sistem terhadap masukan – masukan yang telah dilakukan.

3. Bahasa Pengetahuan *(Knowledge Base Language)*

   Meliputi pengetahuan yang harus dimiliki *user* tentang keputusan dan tentang prosedur pemakaian Sistem Pendukung Keputusan agar sistem dapat digunakan secara efektif. Pemahaman *user* terhadap permasalahan yang dihadapi dilakukan diluar sistem, sebelum *user* menggunakan sistem untuk mengambil keputusan.

2.2.2.3 Tingkat Teknologi Dalam Sistem Pendukung Keputusan

Dalam Sistem Pendukung Keputusan terdapat tiga keputusan tingkatan perangkat keras maupun lunak. Masing – masing tingkatan berdasarkan tingkatan kemampuan berdasarkan perbedaan tingkat teknik, lingkungan dan tugas yang akan dikerjakan.

Ketiga tingkatan tersebut adalah :

a. Sistem Pendukung Keputusan (*Specific DSS*)
b. Pembangkit Sistem Pendukung Keputusan (*DSS Generator*)
c. Peralatan Sistem Pendukung Keputusan (*DSS Tools*)

2.2.3 Model *Fuzzy Multiple Attribute Decision Making* (FMADM)

Fuzzy Multiple Attribute Decision Making(FMADM) **[6]** adalah *" suatu metode yang digunakan untuk mencari alternatif optimal dari sejumlah alternatif dengan kriteria tertentu"*. Inti dari FMADM adalah menentukan nilai bobot untuk setiap atribut,kemudian dilanjutkan dengan proses perankingan yang akan menyeleksi alternatif yang sudah diberikan.

Pada dasarnya, ada tiga pendekatan untuk mencari nilai bobot atribut, yaitu pendekatan subyektif, pendekatan obyektif dan pendekatan integrasi antara subyektif & obyektif. Masing-masing pendekatan memiliki kelebihan dan kelemahan.

Pada pendekatan subyektif, nilai bobot ditentukan berdasarkan subyektifitas dari para pengambil keputusan, sehingga beberapa faktor dalam proses perangkingan alternatif bisa ditentukan secara bebas. Sedangkan pada pendekatan obyektif, nilai bobot dihitung secara matematis sehingga mengabaikan subyektifitas dari pengambil keputusan. **[6]**.

Ada beberapa metode yang dapat digunakan untuk menyelesaikan masalah FMADM. antara lain**[6]:**

a. *Simple Additive Weighting Method (SAW)*
b. *Weighted Product (WP)*
c. *ELECTRE*
d. *Technique for Order Preference by Similarity toIdeal Solution (TOPSIS)*

e. Analytic Hierarchy Process (AHP)

Algoritma FMADM adalah:

1. Memberikan nilai setiap alternatif (Ai) pada setiap kriteria (Cj) yang sudah ditentukan, dimana nilai tersebut di peroleh berdasarkan nilai *crisp*; i=1,2,…m dan j=1,2,…n.
2. Memberikan nilai bobot (W) yang juga didapatkan berdasarkan nilai crisp.
3. Melakukan normalisasi matriks dengan cara menghitung nilai rating kinerja ternormalisasi (rij) dari alternative Ai pada atribut Cj berdasarkan persamaan yang disesuaikan dengan jenis atribut (atribut keuntungan/*benefit* = maksimum atau atribut biaya/*cost*=minimum.)

Apabila berupa artibut keuntungan maka nilai *crisp* (Xij) dari setiap kolom atribut dibagi dengan nilai *crisp* MAX (MAX Xij) dari tiap kolom, sedangkan untuk atribut biaya, nilai *crisp* MIN (MIN Xij) dari tiap kolom atribut dibagi dengan nilai *crisp* (Xij) setiap kolom.

4. Melakukan proses perankingan dengan cara mengalikan matriks ternormalisasi (R) dengan nilai bobot (W).
5. Menentukan nilai preferensi untuk setiap alternatif (Vi) dengan cara menjumlahkan hasil kali antara matriks ternormalisasi (R) dengan nilai bobot (W). Nilai Vi yang lebih besar mengindikasikan bahwa alternative Ai lebih terpilih.

Dalam penelitian ini menggunakan FMADM desngan metode SAW. Adapun langkah-langkahnya adalah:

1. Menentukan kriteria-kriteria yang akan dijadikan acuan dalam pengambilan keputusan, yaitu Ci.
2. Menentukan rating kecocokan setiap alternative pada setiap kriteria.
3. Membuat matriks keputusan berdasarkan kriteria (Ci), kemudian melakukannormalisasi matriks berdasarkan persamaan yang disesuaikan dengan jenis atribut (atribut keuntungan ataupun atribut biaya) sehingga diperoleh matriks ternormalisasi R.
4. Hasil akhir diperoleh dari proses perankingan yaitu penjumlahan dari perkalian matriks ternormalisasi R dengan vektor bobot sehingga diperoleh

nilai terbesar yang dipilih sebagai alternatif terbaik (Ai) sebagai solusi. **[6].**

2.2.4 Metode *Simple Additive Weighting* (SAW)

Metode *Simple Additive Weighting* (SAW) sering juga dikenal istilah metode penjumlahan terbobot. Konsep dasar metode SAW adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut **[6].**

Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternatif yang ada.

Formula untuk melakukan normalisasi tersebut adalah sebagai berikut:
Nilai preferensi untuk setiap alternatif (V_i) seperti pada rumsu 2.1:

$$rij = \frac{Xij}{\text{Max xij}} \quad (2.1)$$

Keterangan

rij = nilai rating kinerja ternormalisasi

xij =nilai atribut yang dimiliki dari setiap kriteria

Max xij =nilai terbesar dari setiap kriteria

Langkah terakhir adalah menjumlahkan setiap altenatif, misal baris pertama dari matriks W * R =A1,baris kedua=A2,dan baris ketiga =A3. Setelah dilakukan penjumlahan maka didapatkan sebuah nilai seperti pada rumus 2.2

$$V_i = \sum_{j=1}^{n} Wj \; rij \quad (2.2)$$

Keterangan :

V_i = ranking untuk setiap alternatif

Wj = nilai bobot dari setiap kriteria

rij = nilai rating kinerja ternomalisasi

Nilai V_i yang lebih besar mengindikasikan bahwa alternatif A_i lebih terpilih.

2.2.5 Unified Modelling Language (UML)

Unified Modelling Language (UML) adalah sebuah bahasa pemodelan standar yang memiliki sintaks dan semantic. Pemodelan ini sangat cocok digunakan untuk merancang dan memodelkan sistem berorientasi objek[7].

Diagram pada UML dibagi menjadi dua bagian yaitu *structural* diagram dan *behavior* diagram. *Structural* diagram digunakan untuk mendeskripsikan relasi antar kelas. *Tools* yang digunakan pada bagian ini yaitu *class diagram*. Sedangkan *behavior* diagram digunakan untuk mendeskripsikan interaksi antara aktor dan sebuah *use case* (bagaimana seorang aktor menggunakan sistem). *Tools* yang digunakan pada bagian ini yaitu *Use case* diagram, *Sequence* diagram, *Collaboration* diagram, *Statechart* diagram, dan *Activity* diagram.

2.2.5.1 Diagram *Class*

Diagram *class* digunakan untuk menggambarkan keadaan suatu sistem dengan menjelaskan keterhubungan antara suatu *class* dengan *class* yang lain yang terdapat pada sistem[9]. Sebuah *class* terdiri dari nama, atribut dan *method.*

Atribut dan *method* dari sebuah kelas mempunyai visibility, Ada tiga jenis visibility yang digunakan yaitu *private*, *public*, dan *protected*. Setiap *class* pada *class* diagram mempunyai hubungan dengan *class* lainnya, ada beberapa jenis hubungan *class*, yaitu *dependency*, asosiasi, agregasi, komposisi dan generalisasi.

a. *Dependency*

Depedency merupakan hubungan terlemah antara *class. Depedency* bermakna satu *class* menggunakan atau memiliki pengetahuan terhadap *class* lain, namun hubungannya hanya sementara dan tidak ada batas waktu yang jelas.

b. Asosiasi

Asosiasi mempunyai hubungan yang lebih kuat dari hubungan *dependency*, dimana suatu *class* tetap berhubungan dengan *class* lain seterusnya. Hubunganasosiasi dibagi menjadi dua jenis yaitu *directional* dan *bidirectional*.

c. Agregasi

Agregasi merupakan bentuk hubungan yang mengimplikasikan kepemilikan suatu *class*. Agregasi juga merupakan bentuk yang lebih kuat dari

asosiasi. Hubungan agregasi dinyatakan dengan simbol *diamond* pada pemilik kelas dan garis utuh berpanah kekelas yang dimiliki.

d. Komposisi

Komposisi merupakan bentuk hubungan antar *class* yang paling kuat. Komposisi digunakan untuk mengambil seluruh bagian dari *class* yang berhubungan. Aturan dari komposisi yaitu hanya boleh ada satu komposisi dalam satu waktu.

e. Generalisasi

Generalisasi merupakan bentuk hubungan antar *class*, dari *class* yang umum dengan *class* yang lebih khusus. Contoh *class* hewan memiliki hubungan generalisasi dengan *class* kucing, karena kucing merupakan *class* khusus dari *class* hewan yang lebih umum.

2.2.5.2 Diagram *Use Case*

Diagram *use case* digunakan untuk mendeskripsikan kejadian-kejadian apa saja yang dapat dilakukan oleh *user*/aktor dan fungsionalitas-fungsionalitas apa saja yang diharapkan dari sistem yang akan dibangun, tanpa mendeskripsikan bagaimana sistem menyelesaikannya[7]. Sebuah *use case* menggambarkan suatu urutan interaksi antara satu atau lebih aktor dan sistem. Dalam fase requirements, model *use case* mengambarkan sistem sebagai sebuah kotak hitam dan interaksi antara aktor dan sistem dalam suatu bentuk naratif, yang terdiri dari masukan dari pengguna dan respon-respon sistem.

Setiap *use case* menggambarkan perilaku sejumlah aspek sistem, tanpa mengurangi struktur internalnya. Selama pembuatan model *use case* secara pararel juga harus ditetapkan obyek-obyek yang terlibat dalam setiap *use case*.

2.2.5.3 Diagram *Sequence*

Diagram *sequence* menggambarkan interaksi antar objek di dalam dan di sekitar sistem yang menekankan pada pengiriman pesan dalam suatu waktu tertentu[7]. *Sequence* diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah

event. Untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan

2.2.5.4 Diagram *Collaboration*

Diagram *collaboration* menunjukan informasi yang sama seperti dalam *sequence* diagram. Perbedaannya adalah dari cara mengelompokan objek *sequence* berdasarkan urutan nomor dari pesan[7].

2.2.5.5 Diagram *Statechart*

Diagram *statechart* menggambarkan keadaan-keadaan pada sistem, memuat status (*state*), transisi, kejadian serta aktifitas[7].

2.2.5.6 Diagram *Activity*

Diagram *activity* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir[7]. *Activity* diagram merupakan *state* diagram khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity* diagram tidak menggambarkan *behaviour* internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

2.2.6 Teknologi *Java*

Java adalah sebuah teknologi yang diperkenalkan oleh *Sun Microsystems* pada pertengahan tahun 1990. Menurut definisi dari *Sun*, *Java* adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *stand alone* ataupun pada lingkungan jaringan. *Java* berdiri di atas sebuah mesin *interpreter* yang diberi nama *Java Virtual Machine* (JVM). JVM inilah yang akan membaca *bytecode* dalam *file .class* dari suatu program sebagai representasi langsung program yang berisi bahasa mesin.

Oleh karena itu, bahasa *Java* disebut sebagai bahasa pemrograman yang *portable* karena dapat dijalankan pada berbagai sistem operasi, asalkan pada

sistem operasi tersebut terdapat JVM. *Platform Java* terdiri dari kumpulan *library*, JVM, kelas-kelas *loader* yang dipaket dalam sebuah lingkungan rutin *Java*, dan sebuah *compiler*, *debuger*, dan perangkat lain yang dipaket dalam *Java Development Kit* (JDK)[8].

Java merupakan salah satu bahasa pemrograman yang menggunakan paradigma pemrograman berbasis objek. Paradigma ini yaitu menggunakan objek untuk membungkus atribut dan operasi yang mungkin pada objek tersebut. Java mempunyai kelebihan dan kekurangan yaitu:

2.6.6.1 Kelebihan Java

Beberapa kelebihan dari java yaitu sebagai berikut:

1. Multiplatform

   Kelebihan utama dari *java* ialah dapat dijalankan di beberapa *platform*/sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan dimana saja. Kelebihan ini memungkinkan sebuah program berbasis java dikerjakan diatas sistem operasi *linux* tetapi dijalankan dengan baik di atas *Microsoft Windows*.

2. OOP (*Object Oriented Programming*)

   Java merupakan salah satu bahasan pemrograman berbasis objek secara murni. Semua tipe data diturunkan dari kelas dasar yang disebut objek. Hal ini sangat memudahkan pemrogram untuk mendesain, membuat, mengembangkan, dan mengalokasi kesalahan sebuah program dengan basis *java* secara cepat, tepat, mudah dan terorganisir.

3. *Library* yang lengkap

   Java terkenal dengan kelengkapan *library*/perpustakaan (kumpulan program-program yang disertakan dalam pemrogram *java*) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.

4. Bergaya C++

   Java memiliki sintaks seperti bahasa pemrograman C++ sehingga menarik

banyak pemrogram C++ untuk pindah ke *java*. Saat ini pengguna *java* sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke *java*.

5. Pengumpulan sampah otomatis

   Java memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas) [8].

2.6.6.2 Kekurangan Java

Beberapa kekurangan Java antara lain :

1. Mudah didekompilasi

   Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini memungkinkan karena kode jadi *java* merupakan *bytecode* yang menyimpan banyak atribut bahasa tingkat tinggi seperti nama-nama kelas, metode dan tipe data.

2. Penggunaan memory yang banyak

   Penggunaan memori untuk program berbasis *java* jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan *Pascal* (lebih spesifik lagi *Delphi* dan Objek *Pascal*)[8].