# BAB II

# TINJAUAN PUSTAKA

## II.1 Sistem

Sistem adalah kumpulan objek seperti orang, sumber daya, konsep, dan prosedur yang dimaksudkan untuk melakukan suatu fungsi yang dapat diidentifikasi atau untuk melayani suatu tujuan. Sebagai contoh, sebuah universitas adalah suatu sistem mahasiswa, fakultas, staf, administrasi, gedung, perlengkapan, ide-ide, dan aturan dengan tujuan mendidik mahasiswa, menghasilkan riset, dan memberikan layanan kepada komunitas (sistem lain). (Turban; 2005:54-55)

## II.2. Masalah

Masalah merupakan suatu kondisi yang berpotensi menimbulkan kerugian luar biasa atau menghasilkan keuntungan luar biasa. Tindakan memberi respons terhadap masalah untuk menekan akibat burukya atau memanfaatkan peluang keuntungannya disebut pemecahan masalah. Pentingnya pemecahan masalah bukan didasarkan pada jumlah waktu yang dihabiskan, tetapi pada konsekuensinya, yaitu apakah pemecahan masalah tersebut bisa menekan sebanyak mungkin kemungkinan kerugian atau memperoleh sebesar mungkin kemungkinan keuntungan. (Kusrini; 2007:7)

### II.2.1 Keputusan

Keputusan merupakan kegiatan memilih suatu strategi atau tindakan dalam pemecahan masalah tersebut. Tindakan memilih strategi atau aksi yang diyakini manajer akan memberikan solusi terbaik atas sesuatu itu disebut pengambilan keputusan.

Kriteria atau ciri-ciri dari keputusan adalah:

1. Banyak pilihan/*alternative*
2. Ada kendala atau syarat
3. Mengikuti suatu pola/model tingkah laku, baik yang terstruktur maupun tidak terstruktur
4. Banyak input/*variable*
5. Ada faktor resiko
6. Dibutuhkan kecepatan, ketepatan, dan keakuratan. (Kusrini; 2007:7)

### II.2.2 Tahap-Tahap Pembuatan Keputusan

Dalam mengambil keputusan dilakukan langkah-langkah sebagai berikut:

1. Identifikasi Masalah
2. Pemiihan metode pemecahan masalah
3. Pengumpulan data yang dibutuhkan untuk melaksanakan model keputusan tersebut
4. Mengimplementasikan model tersebut
5. Mengevaluasi sisi positif dari setiap *alternative* yang ada
6. Melaksanakan solusi terpilih. (Kusrini; 2007:9)

### II.2.3 Kondisi Pengambilan Keputusan

Ada bebrapa keadaan yang mungkin dialami oleh pngambil keputusan ketika mengambil keputusan, yaitu:

1. Pengambilan keputusan dalam kepastian, semua alternative diketahui secara pasti
2. Pengambilan keputusan dalam berbagai tingkat resiko yang dipilih
3. Pengambilan keputusan dalam kondisi ketidakpastian, ada *alternative* yang tidak diketahuidengan jelas. (Kusrini; 2007:9)

## II.3. Sistem Penunjang Keputusan

### II.3.1. Pengertian Sistem Penunjang Keputusan

Little (1970) mendefinisikan DSS (*Decision Support Systems*) sebagai ”sekumpulan prosedur berbasis model untuk data pemrosesan dan penilaian guna membantu para manajer mengambil keputusan.” Dia menyatakan bahwa untuk sukses, sistem tersebut haruslah sederhana, cepat, mudah dikontrol, adaftif, lengkap dengan isu-isu penting, dan mudah berkomunikasi (Efraim Turban; 2005: 137). Sedangkan menurut Alter;2002 DSS merupakan sistem informasi interaktif yang menyediakan informasi, pemodelan, dan pemanipulasian data. Sistem itu digunakan untuk membantu pengambilan keputusan dalam situasi semiterstruktur dansistuasi yang tidak terstruktur, dimana tak seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat. Dapat diambil kesimpulan bahwa sistem penunjang keputusan adalah sebuah sistem yang dapat memecahkan suatu permasalahan. (Kusrini; 2007:15)

### II.3.2. Tujuan Sistem Penunjang Keputusan

Tujuan sistem penunjang keputusan/DSS (*Decision Support Systems*) (Turban;2005) adalah:

1. Membantu manajer dalam pengambilan keputusan atas masalah semi-terstruktur.
2. Memberikan dukungan atas pertimbangan manajer dan bukannya dimaksudkan untuk menggantikan fungsi manajer.
3. Meningkatkan efektivitas keputusan yang diambil manajer lebih dari pada perbaikan efisiensinya.
4. Kecepatan komputasi. Komputer memungkinkan para pengambil keputusan untuk melakukan banyak komputasi secara cepat dengan biaya yang rendah.
5. Peningkatan produktivitas. Membangun satu kelompok pengambil keputusan, terutama para pakar, bisa sangat mahal. Pendukung terkomputerisasi bisa mengurangi ukuran kelompok dan memungkinkan para anggotanya untuk berada di berbagai lokasi yang berbeda-beda (menghemat biaya perjalanan). Selain itu, produktivitas staf pendukung (misalnya analis keuangan dan hukum) bisa ditingkatkan. Produktivitas juga bisa ditingkatkan menggunakan peralatan optimalisasiyang menentukan cara terbaik untuk menjalankan sebuah bisnis.
6. Dukungan kualitas. Komputer bisa meningkatkan kualitas keputusan yang dibuat. Sebagai contoh, semakin banyak data yang diakses, makin banyak juga alternative yang bisa dievaluasi.

7. Berdaya saing. Manajemen dan pemberdayaan sumber daya perusahaan. Tekanan persaingan menyebabkan tugas pengambilan keputusan menjadi sulit.
8. Mengatasi keterbatasan kognitif daam pemrosesan dan penyimpanan. Menurut Simon (1977), otak manusia memiliki kemampuan yang terbatas untuk memproses dan menyimpan informasi. Orang-orang kadang sulit mengingat dan menggunakan sebuah informasi dengan cara yang bebas dari kesalahan. (Kusrini; 2007:16-17)

**II.3.3 Karakteristik yang diharapkan ada di DSS**

Berikut karakteristik yang diharapkan ada di DSS, yaitu:

1. Dukungan kepada pengambilan keputusan, terutama pada situasi semiterstruktur dari tak terstruktur, dengan menyertakan penilaian manusia dan informasi terkomputerisasi. Masalah-masalah tersebut tidak bisa dipecahkan oleh sistem computer lain atau oleh metode atau alat kuantitatif standar.
2. Dukungan untuk semua level manajerial, dari eksekutif puncak sampai manajer lini.
3. Dukungan untuk individu dan kelompok
4. Dukungan untuk keputusan independen dan/atau sekuensial.
5. Dukungan di semua fase prosespengambilan keputusan.
6. Dukungan di berbagai proses dan gaya pengambilan keputusan.
7. Adaptivitas sepanjang waktu.
8. Pengguna merasa seperti di rumah.

9. Peningkatan efektifitas pengambilan keputusan (akurasi, *timelines*, kualitas) ketimbang pada efisiensinya (biaya pengambilan keputusan).
10. Control penuh oleh pengambil keputusan terhadap semua langkah proses pengambilan keputusan dalam memecahkan suatu masalah.
11. Pengguna akhir bisa mengembangkan dan memodifikasi sendiri sistem sederhana.
12. Biasanya, model-model digunakan untuk menganalisis situasi pengambilan keputusan.
13. Akses disediakan untuk berbagai sumber data, format, dan tipe, mulai dari sistem informasi geografis (GIS) sampai sistem berorientasi objek.
14. Dapat digunakan sebagai alat *standalone* oleh seorang pengambil keputusan pada satu lokasi atau didistribusikan di suatu organisasi secara keseluruhan dan di beberapa organisasi sepanjang rantai persediaan.

Karakteristik dari DSS tersebut memungkinkan para pengambil keputusan untuk membuat keputusan yang lebih baik dan lebih konsisten dalam satu cara yang dibatasi oleh waktu. (Turban ; 2005:141-143)

**II.4. Sertifikasi Guru**

Pelaksanaan sertifikasi guru merupakan komitmen pemerintah, dalam hal ini Departemen Pendidikan Nasional, untuk mengimplementasikan amanat Undang-Undang Nomor 14 tahun 2005 tentang Guru dan Dosen. Keberhasilan pelaksanaan sertifikasi dalam rangka meningkatkan mutupendidikan secara nasional, juga menjadi harapan nyata bagi pembangunan pendidikan, dan

pembangunan guru yang professional pembangunan “Insan Indonesia Cerdas dan Kompetitif”.

Pendidikan yang bermutu memiliki kaitan yang ke depan (*Forward linkage*) dan kaitan ke belakang (*Backward linkage*). *Forward linkage* berupa bahwa pendidikan yang bermutu merupakan syarat utama untuk mewujudkan kehidupan bangsa yang maju, modern, dan sejahtera. Sejarah perkembangan da pembangunan bangsa-bangsa mengajarkan pada kita bahwa bangsa yang maju,modern, makmur, dan sejahtera adalah bangsa-bangsa yang memiliki sistem dan praktik pendidikan yang bermutu. *Backward linkage* berupa bahwa pendidikan yang bermutu sangat tergantung pada keberadaan guru yang bermutu, yakni guru yang professional, sejahtera, dan bermartabat. (Hoyyima Khoiri; 2010:9-10)

### II.4.1. Tujuan Sertifikasi Guru

Terlebih dahulu harus diketahui bahwa tujuan sertifikasi guru adalah:

1. Menentukan kelayakan guru dalam melaksanakan tugas sebagai agen pembelajaran dan mewujudkan tujuan pendidikan nasional.
2. Meningkatkan proses dan mutu hasil pendidikan.
3. Meningkatkan profesionalitas guru.

Dengan demikian, secara umum tujuan program sertifikasi guru adalah meningkatkan kualitas tenaga pendidik dan kesejahteraannya yang berujung pada peningkatan kualitas pendidikan secara berkelanjutan. (Hoyyima Khoiri; 2010:9)

### II.4.2. Tahapan-Tahapan yang Harus Dilalui Peserta Sertifikasi

Khusus untuk sertifikasi guru dalam jabatan, berikut rubrik penilaian portofolio yang harus dipahami, yaitu:

1. Kualifikasi akademik

   Peserta yang belum memenuhi kualifikasi akademik S1/D-IV, hanya ijazah tertinggi yang dinilai. Peserta yang memiliki kualifikasi akademik S1/D-IV yang dicapai melalui program alih jenjang/penyetaraan, ijazah diplomanya tidak dinilai.

2. Pendidikan pelatihan

   R: Relevan; materi diklat secara langsung meningkatkan kompetensi pedagogic dan kompetensi profesional.

   KR: Kurang Relevan ; materi diklat mendukung kinerja professional guru.

   TR: Tidak Relevan; tidak dinilai.

   Pendidikan Prajabatan atau STPP sebagai persyaratan untuk menjadi PNS tidak diperhitungkan. Pelatihan sebelum menjabat sebagai guru tidak dinilai.

3. Pengalaman mengajar

   Dihitung sejak menjadi guru PNS , Guru Tetap Yayasan, atau GTT dengan dilengkapi SK(SK PNS, SK Pemda, SK Yayasan).

4. Perencanaan dan pelaksanaan pembelajaran

   a. Perencanaan pembelajaran

      Lima RP/RPP/SP dinilai oleh asesor dengan menggunakan instrument penilaian RPP dan dihitung skor reratanya.

b. Pelaksanaan pembelajaran

Dinilai oleh kepala sekolah (untuk guru yang ditugasi sebagai kepala sekolah, aspek ini dinilai oleh pengawas/kepala dinas). Jangan sekali-kali minta tanda tangan dalam keadaan belum dinilai.

5. Penilaian dari atasan dan pengawas

Jika baru salah satu, atau kepala sekolah menilai diri sendiri—Catatan Klarifikasi: Jangan sekali-kali minta tanda tangan dalam keadaan belum dinilai.

6. Prestasi akademik

a. Lomba dan karya akademik

Yang dimaksud juara adalah juara I, II, atau III. Kejuaraan dinilai pada setiap kegiatan (*event*).

b. Pembimbingan kepada teman sejawat/siswa

Jenis pembimbingan teman sejawat sebagai instruktur, guru inti, guru pemandu, atau tutor diakui (diberi skor) apabila guru yang bersangkutan telah memiliki hak untuk tugas tersebutdengan bukti pernah mengikuti dan memilki sertifikat *training of trainer* (TOT).

7. Karya pengembangan profesi

Catatan:

a. Buku publikasi nasional adalah buku yang dipakai secara nasional dan br-ISBN dan ditetapkan oleh BSNP sebagai buku standar.

b. Penskoran mempertimbangkan kualitas modul/diktat

c. Penskoran mempertimbangkan kualitas laporan yang meliputi aspek masalah, telaah teoreti, metode, hasil dan tata tulis ilmiah.

d. Penskoran mempertimbangkan kualitas; karya teknologi mempertimbangkan manfaat, dan karya seni mempertimbangkan estetika.

8. Keikutsertaan dalam forum ilmiah

   Forum ilmiah dinilai relevan apabila materinya mendukung kompetensi professional dan pedagogic.

9. Pengalaman organisasi di bidang kependidikan dan sosial

   a. Pengurus organisasi di bidang kependidikan dan sosial.

   b. Tugas tambahan.

10. Penghargaan yang relevan dengan bidang pendidikan

    Daerah khusus adalah daerah yang terpencil atau terbelakang, daerah dengan kondisi masyarakat yang terpencil, daerah perbatasan dengan negara lain, daerah yang mengalami bencana alam, bencana sosial.

    (Hoyyima Khoiri; 2010: 22-30)

## II.5 Metode *Profile Matching*

*Profile Matching* secara garis besar merupakan proses membandingkan antara kompetensi individu kedalam kompetensi jabatan sehingga dapat diketahui perbedaan kompetensinya (disebut juga *gap*), semakin kecil *gap* yang dihasilkan maka bobot nilainya semakin besar yang berarti memiliki peluang lebih besar untuk karyawan menempati posisi tersebut.

**Langkah Penyelesaian :**

**1) *Gap = Profil Karyawan – Profile Jabatan***

Setelah menentukan bobot nilai *gap* untuk ketiga aspek yaitu aspek kapasitas intelektual, sikap kerja dan perilaku dengan cara yang sama. Kemudian tiap aspek dikelompokkan menjadi 2 (dua) kelompok yaitu kelompok *Core Factor* dan *Secondary Factor*. Untuk perhitungan core factor dapat ditunjukkan pada rumus di bawah ini:

**2)** $$\mathbf{NCF = \frac{\Sigma\ NC\ (I, s, p)}{\Sigma\ IC}}$$

Keterangan:

NCF : Nilai rata-rata *core factor*

NC(*i, s, p*) : Jumlah total nilai *core factor (Intelektual, Sikap kerja, Perilaku)*

IC : Jumlah *item core factor*

Sedangkan untuk perhitungan *secondary factor* dapat ditunjukkan pada rumus di bawah ini:

**3)** $$\mathbf{NCS = \frac{\Sigma\ NS\ (I, s, p)}{\Sigma\ IS}}$$

Keterangan:

NSF : Nilai rata-rata *secondary factor*

NS(*i, s, p*) : Jumlah total nilai *secondary factor (Intelektual, Sikap kerja, Perilaku)*

IS : Jumlah *item secondary factor*

Dari hasil perhitungan dari tiap aspek di ataskemudian dihitung nilai total berdasar presentasi dari *core* dan *secondary* yang diperkirakan berpengaruh

terhadap kinerja tiap-tiap profil. Contoh perhitungan dapat dilihat pada rumus di bawah ini:

**4)** $\boxed{N\,(i, s, p) = (x)\%NCF(i, s, p) + (x)\%NSF(i, s, p)}$

Keterangan:

*(i,s,p) : (Intelektual, Sikap Kerja, Perilaku)*

*N(i,s,p)* : Nilai total dari aspek

*NCF(i,s,p)* : Nilai rata-rata *core factor*

*NSF(i,s,p)* : Nilai rata-rata *secondary factor*

*(x)%* : Nilai persen yang di*input*kan

Hasil akhir dari proses ini adalah ranking dari kandidat yang diajukan untuk mengisi suatu jabatan tertentu. Penentuan ranking mengacu pada hasil perhitungan tertentu. Perhitungan tersebut dapat ditunjukkan pada rumus di bawah ini:

**5)** $\boxed{Ha = (x)\%Ni + (x)\%Ns + (x)\%Np}$

Keterangan:

*Ha* : Hasil Akhir

*Ni* : Nilai Kapasitas Intelektual

*Ns* : Nilai Sikap Kerja

*Np* : Nilai Perilaku

*(x)%* : Nilai Persen yang di*input*kan

Contoh Kasus :

Np SALES-0001 = (60% x 4,5) . (40% x 4,5) = 4,5

Np SALES-0001 = (60% x 4,75) . (40% x 4,5) = 4,65

Np SALES-0001 = (60% x 4) . (40% x 3) = 3,6

**Tabel II.1 Nilai Total Gap Aspek Perilaku**

| No | ID_Karyawan | NCF | NSF | Np |
|---|---|---|---|---|
| 1 | SALES-001 | 4,5 | 4,5 | 4,5 |
| 2 | SALES-002 | 4,75 | 4,5 | 4,65 |
| 3 | SALES-003 | 4 | 3 | 3,6 |

( Ilman Fahma Dwijaya;2009:5)

**1. Perhitungan Penentuan Hasil Akhir/Ranking**

Hasil akhir dari proses ini adalah ranking dari kandidat yang diajukan untuk mengisi suatu jabatan tertentu. Penentuan ranking mengacu pada hasil perhitungan tertentu. Perhitungan tersebut dapat ditunjukkan pada rumus di bawah ini:

$Ha = (x)\%Ni + (x)\%Ns + (x)\%Np$

Keterangan:

*Ha* : Hasil Akhir

*Ni* : Nilai Kapasitas Intelektual

*Ns* : Nilai Sikap Kerja

*Np* : Nilai Perilaku

*(x)%* : Nilai Persen yang di*input*kan

Sebagai contoh dari rumus untuk perhitungan hasil akhir di atas maka hasil akhir dari karyawan dengan asub aspek PH001 dengan nilai persen = 20%, 30% dan 50%. Dapat dilihat pada proses ini :

Hasil akhir SALES-0001

= (20% x 3,94) + (30% x 4,13) + (50% x 4,5)

= 0,78 + 1,24 + 2,25

= 4,278

Hasil akhir SALES-0002

= (20% x 4,2) + (30% x 3,56) + (50% x 4,65)

= 0,84 + 1,06 + 2,32

= 4,235

Hasil akhir SALES-0003

= (20% x 4,16) + (30% x 3,73) + (50% x 3,6)

= 0,83 + 1,11+ 1,8

= 3,752

Proses di atas dapat dilihat pada tabel II.2 :

**Tabel II.2 Hasil Akhir**

| No | ID_Karyawan | Ni | Ns | Np | Ha |
|---|---|---|---|---|---|
| 1 | SALES-001 | 3,94 | 4,13 | 4,5 | 4,278 |
| 2 | SALES-002 | 4,2 | 3,56 | 4,65 | 4,235 |
| 3 | SALES-003 | 4,16 | 3,73 | 3,6 | 3,752 |

( Ilman Fahma Dwijaya;2009:5)

Nilai terbesar ada pada Ha adalah alternatif yang terpilih sebagai alternatif terbaik pada proses kenaikan jabatan. (Ilman Fahma Dwijaya; 2009: 2-3,5-6)

## II.6 Pengenalan *Personal Home Page (PHP)*

PHP adalah bahasa *server-side scripting* yang menyatu dengan HTML untuk membuat halaman *web* yang dinamis. Maksud dari *server-side scripting* adalah sintaks dan perintah-perintah yang diberikan akan sepenuhnya dijalankan di *server* tetapi disertakan pada dokumen HTML. Pembuatan *web* ini merupakan kombinasi antara PHP sendiri sebagai bahasa pemrograman dan HTML sebagai pembangunan halaman *web*. (Bimo Sunarfrihantono; 2003:23)

PHP merupakan *software* yang *open source* (gratis) dan mampu lintas *platform*, yaitu dapat digunakan dengan sistem operasi dan *web server* apapun. Kode Program PHP menyatu dengan tag-tag HTML dalam satu file. Kode PHP diawali dengan tag <?php dan ditutup dengan tag ?> . File yang berisi tag HTML dan kode PHP ini diberi ekstensi php. Berdasarkan ekstensi ini, pada saat file diakses, *server* akan tahu bahwa file ini mengandung kode PHP. Server akan menerjemahkan kode ini dan menghasilkan *output* dalam bentuk tag HTML yang akan dikirim ke *browser client* yang mengakses file tersebut. PHP menawarkan koneksi yang baik dengan beberapa *database* antara lain, *PostgreSQL, mSQL, MySQL, FilePro, Velocis, Oracle, Unix dbm* dan tak terkecuali semua *databse* ber-*interface* ODBC. (Bimo Sunarfrihantono; 2003:24)

## II.7 *MySQL*

*MySQL* adalah *multiuser database* yang menggunakan bahasa *Structured Query Language* (SQL). MySQL dalam operasi *client-server* melibatkan *server daemon* MySQL di sisi *server* dan berbagai macam program serta *library* yang

berjalan di sisi *client.* MySQL mampu menangani data yang cukup besar. (Bimo Sunarfrihantono; 2003:65)

MySQL merupakan database yang paling digemari dan paling digemari dikalangan pemrograman web, dengan alasan bahwa program ini merupakan *database* yang sangat kuat dan cukup stabil untuk digunakan sebagai media penyimpanan data.

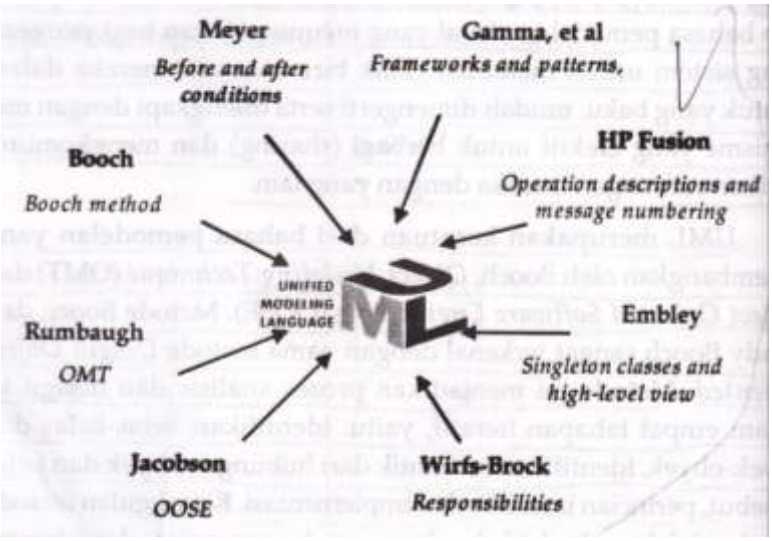
## II.8. UML (*Unified Modelling Language*)

UML( *Unified Modelling Language* ) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (Sharing) dan mengkomunikasikan rancangan dengan baik. (Munawar; 2005:17)

UML merupakan kesatuan bahasa pemodelan yang dikembangkan oleh Booch, *Object Modeling Technique* (OMT) dan *object Oriented Engineering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented.* Metode ini menjadikan proses analisis dan design ke dalam empat tahapan iteratif, yaitu: identifikasi kelas-kelas dan objek-objek, identifikasi semantik dari hubungan objek dan kelas tersebut, perincian interface dan implementasi. Keunggulan metode Booch adalah pada detil dan kayanya dengan notasi dan elemen. Pemodelan OMT yang dikembangkan oleh Rumbaugh

didasarkan pada analisis terstruktur dan pemodelan *entity-relationship*. Tahapan utama dalam metodologi in adalah analisis, desain sistem, desain objek dan impelmentasi. Keunggulan motode ini adalah dalam penotasian yang mendukung semua konsep OO. Metode OOSE dari Jacobson lebih memberi penekanan dan *use case.* OOSE memiliki tiga tahapan yaitu membuat model *requirement* dan analisis, desain dan implementasi dan model pengujian (test Model). Keunggulan metode ini adalah mudah dipelajari karena memiliki notasi yang sederhananamun mencangkup seluruh tahapan dalam rekayasa perangkat lunak.

Dengan UML, metode Booch, OMT dan OOSE digabungkan dengan membuang elemen-elemen yang tidak praktis ditambah dengan elemen-elemen dari metode lain yang lebih efektif dan elemen-elemen baruyang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam daripada metode lainnya. Unsur-unsur yang membentuk UML ditunjukkan dalam Gambar II.1

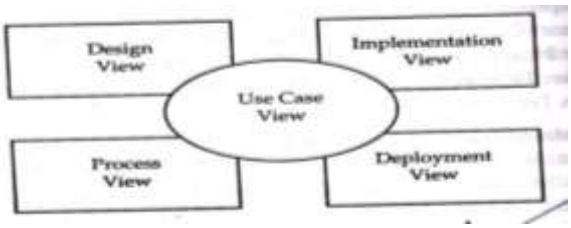



**Gambar II.1 Unsur-unsur yang membentuk UML**

( Munawar;2005:18)

UML adalah hasil kerja dari konsorsium berbagai organisasi yang berhasil dijadikan sebagai standar baku dalam OOAD (*Object Oriented Analysis dan Design* ). UML tidak hanya dominan dalam penotasian di lingkungan OO tetapi juga populer di luar lingkungan OO. Ada tiga karakter penting yang melekat di UML yaitu sketsa, cetak biru dan bahasa pemrograman. Sebagai sebuah sketsa UML bisa berfungsi sebagai sebuah cetak biru kerena sangat lengkap dan detil. Dengan cetak biru ini maka akan bisa diketahui informasi detil tentang coding program (*Forward engineering* ) atau bahkan membaca program dan mengiterpretasikannya kembali ke dalam diagram (*reverse engineering* ). *Reverse engineering* sangat berguna pada situasi dimana kode program yang tidak terdokumentasi asli hilang atau bahkan belum dibuat sama sekali. Sebagai bahasa pemrogaraman, UML dapat menterjemahkan diagaram yang ada di UML menjadi kode program siap untuk dijalankan.

UML dibangun atas model 4+1 *view*. Model ini didasarkan pada fakta bahwa struktur sebuah sistem dideskripsikan dalam *view* dimana salah satu diantaranya *use case view*. *Use case view* ini memegang peran khusus untuk mengintegrasikan *content* ke *view* yang lain. Model 4+1 *view* ditunjukkan pada gambar II.2.




**Gambar II.2 Model 4+1 *View***

(Munawar;2005:20)

Kelima *view* tersebut tidak berhubungan dengan diagram yang dideskripsikan di UML. Setiap *view* berhubungan dengan perspektif tertentu dimana sistem akan diuji. *View* yang berbeda akan menekankan pada aspek yang berbeda dari sistem yang mewakili tentang sistem bisa dibentuk dengan menggabungkan informasi-informasi yang ada pada kelima *view* tersebut.

*Use case view* mendefinisikan perilaku eksternal sistem. Hal ini menjadi daya tari bagi *end user,* analis dan tester. Pandangan ini mendefinisikan kebutuhan sistem karena mengandung semua *view* yang lain yang mendeskripsikan aspek-aspek tertentu dari peran dan sering dikatakan yang mendrive proses pengembangan perangkat lunak.

*Design view* mendeskripsikan struktur logika yang mendukung fungsi-fungsi yang dibutuhkan di *use case*. *Design view* ini berisi definisi komponen program, class-class utama bersama-sama dengan spesifikasi data, perilkau dan interaksinya. Informasi yang terkandung di *view* ini menjadi perjatian para programer karena menjelaskan secara detil bagaimana fungsionalitas sistem akan diimplementasikan.

Implemantasi *view* menjelaskan komponen-komponen fisi dari sistem yang akan dibangun. Hal ini berbeda dengan komponen logic yang dideskripsikan pada *design view.* Termasuk disini diantaranya *file exe*, *library* dan *database.* Informasi yang ada di *view* dan integrasi sistem.

Proses *view* berhubungan dengan hal-hal yang berkaitan dengan *concurrency* do dalam sistem. Sedangkan *deployment view* menjelaskan bagaimana komponen-komponen fisik didistribusikan ke lingkungna fisik seperti

jaringan komputer dimana sistem akan dijalankan. Kedua *view* ini menunjukkan kebutuhan non fungsional dari sistem seperti toleransi kesalahan dan hal-hal yang berhubungan dengan kinerja (Munawar;2005:17-21).

**II.8.1. Use Case Diagram**

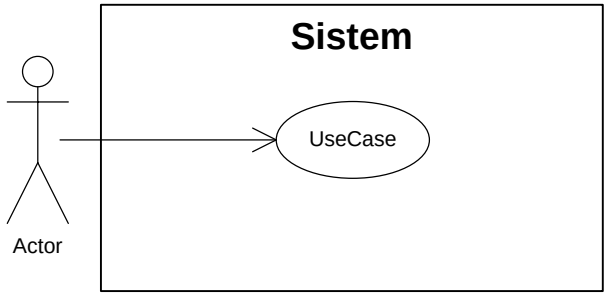
*Use case* adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara deskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut *scenario.* Setiap *scenario* mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, sistem yang lain, perangkat keras atau urutan waktu. Dengan demikian secara singkat bisa dikatakan *use case* adalah serangkaian *scenario* yang digabungkan bersama-sama oleh tujuan umum pengguna.

Dalam pembicaraan tentang *use case*, pengguna biasanya disebut dengan *actor*. *Actor* adalah sebuah peran yang bisa dimainkan oleh pengguna dalam interaksinya dengan sistem.

Model *use case* adalah bagai dari model *requirement.* Termasuk disini adalah problem domain object dan penjelasan tentang *user interface*. *Use case* memberikan spesifikasi fungsi-fungsi yang ditawarkan oleh sistem dari *perspectif user.*

Notasi *use case* menunjukkan 3 aspek dari sistem yaitu *actor use case* dan *system/sub system boundary. Actor* mewakili peran orang, *system* yang lain atau

alat ketika berkomunikasi dengan *use case.* Ilustrasi *actor, usecase* dan *system* ditunjukkan pada gambar II.3




**Gambar II.3 *Usecase* Diagram**

(Munawar;2005:64)

Untuk mengidentifikasi *actor,* harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks targer sistem. *Actor* adalah *abstraction* dari orang dan sistem yang lain yang mengaktifkan fungsi dari target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa *actor* berinteraksi dengan *use case*, tetapi tidak memiliki kontrol atas *use case.*

*Use case* adalah abstraksi dari interaksi antara sistem dan *actor.* Oleh karena itu sangat penting untuk memilih abstraksi yang cocok. *Use case* dibuat berdasarkan keperluan *actor. Use case* harus merupakan ‘apa’ yang dikerjakan *software* aplikasi, bukan ‘bagaimana’ *software* aplikasi mengerjakannya. Setiap *use case* harus diberi nama yang menyatakan apa hal yang dicapai dari hasil interaaksinya dengan *actor.* Namun *use case* boleh terdiri dari beberapa kata dan tidak boleh ada dua *use case* yang memiliki nama yang sama (Munawar;2005:63-66).

### II.8.2. Class Diagram

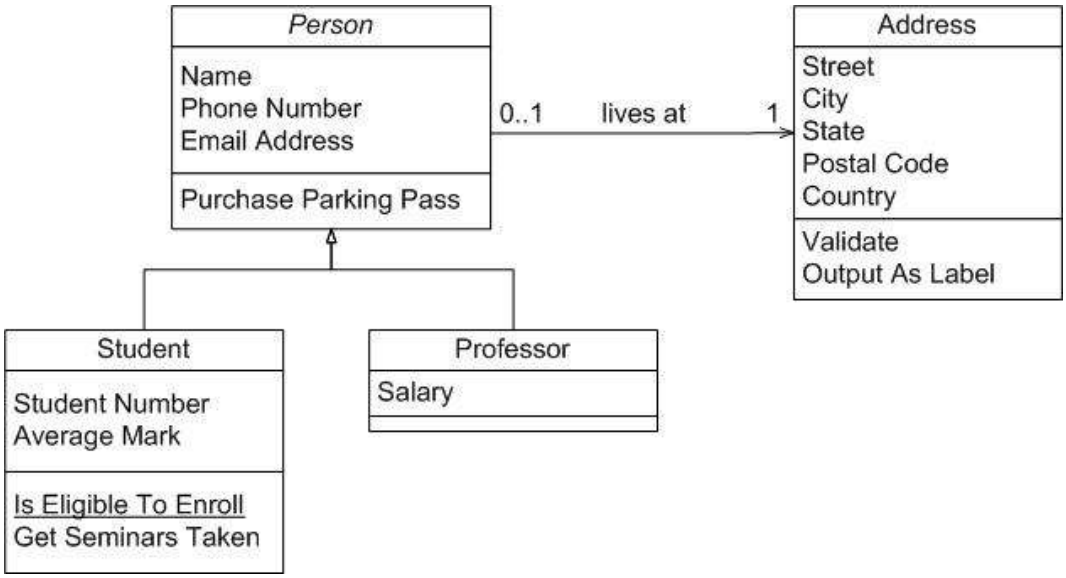
*Class* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (*atribut*/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. *Class* memiliki tiga area pokok :

1. Nama kelas
2. Atribut
3. Metode

*Atribut* dan metode dapat memiliki salah satu sifat berikut :

1. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.
2. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan.
3. *Public*, dapat dipanggil oleh siapa saja.

*Class* dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metode. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class.* Contoh diagram *class* dapat dilihat pada gambar II.4 dibawah ini:



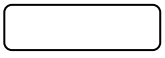
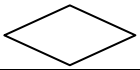

**Gambar II.4 *Class* Diagram**

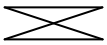
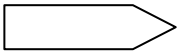
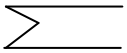
(Munawar;2005:220)

### II.8.3. Activity Diagram

*Activity Diagram* adalah teknik untuk mendiskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity Diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity* diagram bisa mendukung perilaku paralel sedangkan flowchart tidak bisa. Adapun simbol *activity diagram* dapat dilihat pada table II.7 :

**Tabel II.7. Simbol *Activity Diagram***

| Notasi | Keterangan |
|---|---|
|  | Titik Awal |
|  | Titik Akhir |
|  | *Activity* |
|  | Pilihan untuk pengambilan keputusan |

| | |
|---|---|
|  | *Fork* digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu |
|  | *Rake* menunjukkan adanya dekomposisi |
|  | Tanda waktu |
|  | Tanda pengiriman |
|  | Tanda penerimaan |
|  | Aliran Akhir (*Flow Final*) |

(Munawar;2005:110)

**II.8.4. Squence Diagram**

*Sequence diagram* digunakan untuk menggambarkan perilaku pada sebuah sekenario. Diagram ini menunjukkan sejumlah contoh objek dan pesan yang diletakkan diantaran objek-objek ini di dalam *use case.*

Komponen utama *sequence diagram* terdiri atas objek yang ditulisakan dengan kotak segiempat bernama. *Messege* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress vertical.*

1. Objek /*participant*

Objek diletakkan di dekat bagian atas diagram dengan urutan dari kiri ke kanan. Mereka diatur dalam urutan guna menyederhanakan diagram. Setiap *participant* dihubungkan dengan garis titik-titik yang disebut *lifeline.* Sepanjang *lifeline* ada kotak yang disebut *activation. Activation* mewakili sebuah eksekusi operasi dari *participant.* Panjang kotak ini berbanding lurus dengan durasi *activation.* Bentuk *participant* dapat dilihat pada gambar II.5

**Gambar II.5 Bentuk *Participant***

(Munawar;2005:88)

2. *Messege*

Sebuah *messsage* bergerak dari satu *participant* ke *participant* yang lain dan dari satu *lifeline* ke *lifeline* yang lain. Sebuah *participant* bisa mengirim sebuah *message* kepada dirinya sendiri.

Sebuah *message* bisa jadi *simple, synchronous* atau *asynchoronous. Message* yang *simple* adalah sebuah perpindahan (transfer), contoh dari satu *participant* ke *participant* yang lainnya. Jika sebuah *participant* mengirimkan sebuah *messagae* tersebut akan ditunggu sebelum diproses dengan urusannya. Namun jika *message asynchoronous* yang dikirimkan, maka jawabannya atas *message* tersebut tidak perlu ditunggu. Simbol *message* pada *squnence diagram* dapat dilihat pada gambar II.6

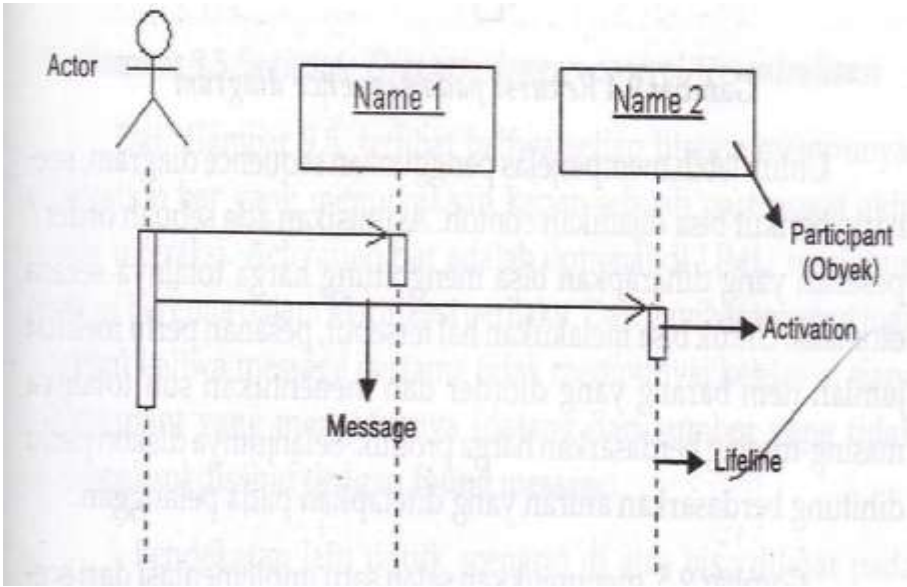

**Gambar II.6 Bentuk *Messege***

(Munawar;2005:88)

*3. Time*

*Time* adalah diagram yang mewakili waktu pada arah vertikal. Waktu dimulai dari ata ke bawah. *Message* yang lebih dekat dari atas akan dijalankan terlebih dahulu dibanding *message* yang lebih dekat ke bawah.

Terdapat dua dimensi pada *squence diagram* yaitu dimensi dari kiri ke kanan menunjukkan tata letak *participant* dan dimensi dari atas ke bawah menunjukkan lintasan waktu. Simbol-simbol yang ada pada *squence diagram* ditunjukkan pada gambar II.7




**Gambar II.7 Bentuk *Time***

(Munawar;2005:89)