BAB II

LANDASAN TEORI

II.1. Sistem

Sistem merupakan kumpulan dari unsur atau elemen-elemen yang saling berkaitan / berinteraksi dan saling mempengaruhi dalam melakukan kegiatan bersama untuk mencapai suatu tujuan tertentu. Menurut Jerry FithGerald, sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan dan berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu. Contoh : Sistem komputer, terdiri dari software, hardware, dan brainware (Hendra, 2012 : 157).

Berdasarkan prinsip dasar secara umum, sistem terbagi dalam:

1. *Sistem Terspesialisasi* adalah sistem yang sulit diterapkan pada lingkungan yang berbeda, misalnya sistem biologi: ikan yang dipindahkan ke darat.
2. *Sistem Besar* adalah sistem yang sebagian besar sumber dayanya berfungsi melakukan perawatan harian. Misalnya dinosaurus sebagai sistem biologi menghabiskan sebagian besar masa hidupnya dengan makan.
3. *Sistem Sebagai Bagian dari Sistem Lain*. Sistem selalu merupakan bagian dari sistem yang lebih besar, dan dapat terbagi menjadi sistem yang lebih kecil.

4. *Sistem Berkembang*. Walaupun tidak berlaku bagi semua sistem, hamper semua sistem selalu berkembang. (Hendra, 2012 : 163)

II.1.1. Karakteristik Sistem

Model umum sebuah sistem terdiri dari input, proses, dan output. Selain itu sebuah sistem juga memiliki karakteristik atau sifat-sifat tertentu, yang mencirikan bahwa hal tersebut bisa dikatakan sebagai suatu sistem. Adapun karakteristik yang dimaksud adalah sebagai berikut (Hendra, 2012 : 158-160) :

1. Komponen Sistem *(Components)*

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang bekerja sama membentuk satu kesatuan. Komponen-komponen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem.

2. Lingkungan Luar Sistem *(Environtment)*

   Environment merupakan segala sesuatu diluar batas sistem yang memengaruhi operasi dari suatu sistem. Lingkungan luar sistem ini dapat bersifat menguntungkan atau merugikan. Lingkungan luar yang menguntungkan harus dipelihara dan dijaga agar tidak hilang pengaruhnya, sedangkan lingkungan luar yang merugikan harus dimusnahkan atau dikendalikan agar tidak mengganggu operasi sistem.

3. Batasan Sistem *(Boundary)*

Batasan sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

4. Penghubung Sistem *(interface)*

   Media penghubung antara suatu subsistem dengan subsistem yang lainnya untuk membentuk satu kesatuan sehingga sumber-sumber daya mengalir dari subsistem yang satu ke subsistem yang lainnya. Dengan kata lain, *output* dari suatu subsistem akan menjadi *input* dari subsistem yang lainnya.

5. Masukan Sistem *(Input)*

   Energi yang dimasukan ke dalam sistem disebut masukan sistem. Masukan dapat berupa Masukan Perawatan (*Maintenance Input*) adalah energi yang dimasukkan supaya sistem tersebt dapat beroperasi.

6. Keluaran Sistem *(Output)*

   Merupakan hasil dari energi yang diolah oleh sistem, meliputi *output* yang berguna, contohnya informasi yang dikeluarkan oleh komputer. Dan *output* yang tidak berguna dikenal sebagai sisa pembuangan, contohnya panas yang dikeluarkan oleh komputer.

7. Pengolahan Sistem (*Process*)

   Bagian yang memproses masukan untuk menjadi keluaran yang diinginkan.

8. Tujuan Sistem *(Goal)*

   Setiap sistem pasti mempunyai tujuan ataupun sasaran yang memengaruhi *input* yang dibutuhkan dan *output* yang dihasilkan. Dengan kata lain, suatu sistem akan dikatakan berhasil kalau pengoperasian sistem itu mengenai sasaran atau tujuannya.

II.2. Sistem Pendukung Keputusan

DSS merupakan sistem informasi interaktif yang menyediakan informasi, pemodelan dan pemanipulasian data. Sistem itu digunakan untuk membantu pengambilan keputusan dalam situasi yang semiterstruktur dan situasi yang tidak terstruktur, di mana tak seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat. (Puspitasari, 2013 : 13).

Persoalan pengambilan keputusan, pada dasarnya adalah bentuk pemilihan dari berbagai alternatif tindakan yang mungkin dipilih yang prosesnya melalui mekanisme yang terbaik. Penyusunan model keputusan merupakan suatu cara untuk mengembangkan hubungan-hubungan logis yang mendasari persoalan keputusan ke dalam suatu model matematis yang mencerminkan hubungan yang terjadi diantara faktor-faktor yang terlibat. (Puspitasari, 2013 : 13).

II.3. Metode *Profile Matching*

Profile Matching merupakan suatu proses yang sangat penting dalam manajemen SDM di mana terlebih dahulu ditentukan kompetensi (kemampuan) yang diperlukan oleh suatu jabatan. Kompetensi kemampuan tersebut haruslah dapat dipenuhi oleh pemegang atau calon yang akan dinilai kinerjanya. Dalam proses *Profile Matching* secara garis besar merupakan proses membandingkan

antara kompetensi individu ke dalam kompetensi jabatan sehingga dapat diketahui perbedaan kompetensinya (disebut juga *gap*), Semakin kecil *gap* yang dihasilkan maka bobot nilainya semakin besar berarti memiliki peluang lebih besar untuk karyawan menempati posisi tersebut. . (Puspitasari, 2013 : 13).

II.3.1 Perhitungan *Profile Matching*

Dalam proses *profile matching* secara garis besar merupakan proses membandingkan antara setiap kriteria setiap penilaian dalam sebuah proposal usulan penelitian yang diajukan sehingga diketahui perbedaan skornya (disebut juga *gap*), semakin kecil *gap* yang dihasilkan maka bobot nilainya semakin besar yang berarti memiliki peluang lebih besar untuk prioritas kelayakan/kelulusan. Nilai gap dapat dihitung menggunakan persamaan (1). Sedangkan pembobotan nilai gap ditentukan berdasarkan Tabel 1.

$$GAP = Profile\ proposal - Profile\ ideal \dots\dots\dots\dots\dots\dots\dots\dots (1)$$

Langkah selanjutnya adalah menghitung nilai *core factor* dan *secondary factor*. *Core factor* merupakan kriteria penilaian yang paling utama harus terkandung dalam sebuah proposal penelitian. Perhitungan *core factor* menggunakan persamaan.

Tabel II.1. Pembobotan Nilai GAP

No	Selisih	Bobot	Keterangan
1	0	5	Tidak ada selisih skor kriteria
2	1	4,5	Kriteria kelebihan 1 level
3	-1	4	Kriteria kekurangan 1 level
4	2	3,5	Kriteria kelebihan 2 level
5	-2	3	Kriteria kekurangan 2 level
6	3	2,5	Kriteria kelebihan 3 level
7	-3	2	Kriteria kekurangan 3 level

Sumber : Edi Faizal (2014 : 61)

$$NCF = \frac{\sum NC(kriteria)}{\sum IC} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (2)$$

Keterangan :

NCT : Nilai rata-rata *core factor*

NC : Jumlah Total nilai *core factor*

IC : Jumlah item *core factor*

Sedangkan secondary factor merupakan itemitem selain yang ada pada faktor utama (core factor). Secondary factor dihitung menggunakan persamaan.

$$NSF = \frac{\sum NS(kriteria)}{\sum IS} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3)$$

Keterangan :

NST : Nilai rata-rata *secondary factor*

NS : Jumlah Total nilai *secondary factor*

IS : Jumlah item *secondary factor*

Selanjutnya perhitungan nilai total berdasar nilai dari *core* dan *secondary factor* yang digunakan sebagai kriteria penilaian yang berpengaruh terhadap kelulusan proposal penelitian. Perhitungan dapat dilakukan menggunakan persamaan.

$$N\ (Tot\_kriteria) = (x)\%NCF + (x)\%NSF \dots\dots\dots\dots\dots\dots\dots\dots (4)$$

Keterangan :

NCT : Nilai rata-rata *core factor*

NST : Nilai rata-rata *secondary factor*

NT : Nilai total kriteria penilaian

Langkah terakhir adalah perhitungan ranking, yang dilakukan dengan menggunakan persamaan

$$Ranking\ = (x)\%N1 + (x)\%N2 + (x)\%Nn \dots\dots\dots\dots\dots\dots\dots\dots (5)$$

Keterangan :

N1, N2, Nn : Nilai total per kriteria

(x)% : Persentase nilai kriteria

Studi kasus dalam implementasi metode *Profile Matching* untuk penentuan penerimaan usulan penelitian internal Dosen STMIK El Rahma ditentukan dalam beberapa tahap.

1. Menentukan nilai bobot kriteria

Tabel II.2. Bobot Kriteria

No	Kriteria	Bobot (%)
1	Abstrak (NA)	12.5
2	Pendahuluan (NP)	25
3	Tinjauan pustaka (NT)	25
4	Metode penelitian (NM)	25
5	Anggaran dan jadwal penelitian (NJ)	12.5
Total		100

Sumber : Edi Faizal (2014 : 63)

2. Penilaian kriteria dan sub kriteria, setiap kriteria dan sub kriteria akan digunakan untuk melakukan perhitungan gap. Berdasarkan lima kriteria yang digunakan, terdapat satu penilaian yang memiliki sub kriteria yaitu kriteria pendahuluan.
3. Perhitungan gap dan pembobotan kriteria, perhitungan nilai gap dilakukan menggunakan persamaan (1) sehingga diperoleh bobot masingmasing kriteria berdasarkan Tabel II.2. Untuk perhitngan *core factor* dan *secondary factor* menggunakan persamaan (2) dan persamaan (3).
4. Penggabungan nilai sub kriteria, proses ini bertujuan untuk memperoleh perhitungan nilai total kriteria. Perhitungan nilai *core* dan *secondary factor* hanya dilakukan pada kriteria pendahuluan. Berdasarkan Tabel 3 dapat dilakukan perhitungan nilai total

menggunakan persamaan (4). Penggunaan persentase pada *core factor* (CF) dan *secondary factor* (SF) masing-masing adalah 60% dan 40% maka diperoleh nilai kriteria pendahuluan (NP).

Tabel II.3. Nilai total kriteria NP

No	ID_Prop	CF	SF	NP
1	PR001	4,00	5,00	4,4
2	PR002	4,33	4,25	4,3
3	PR003	4,33	4,00	4,2
4	PR004	4,67	4,00	4,4
5	PR005	4,00	4,50	4,2

Sumber : Edi Faizal (2014 : 63)

5. Perhitungan nilai total dan ranking, Perhitungan dilakukan pada semua nilai total kriteria dan bobot kriteria, untuk menghasilkan perangkingan nilai kelayakan. Berdasarkan persentase setiap kriteria (dari Tabel 3), maka dapat dihitung nilai akhir (nilai total) dan perangkingan nilai sebagaimana disajikan pada Tabel 1I.4. Semakin besar nilai akhir maka semakin tinggi prioritas kelulusan proposal tersebut dan sebaliknya. Mengingkat kuota yang terbatas pada setiap periode penyelenggaraan penelitian internal, maka pemilihan proposal yang dinyatakan lulus dan di biayai dipilih dengan pengurutan rangking nilai.

Tabel II.3. Nilai total dan rangking

No	ID_ Prop	NA	NP	NT	NM	NJ	Nilai Akhir	Rang king
1	PR001	4	4,4	5	5	5	4,73	1
2	PR002	3	4,3	4	3	4	3,70	4
3	PR003	5	4,2	4	4	4	4,18	2
4	PR004	3	4,4	4	3	5	3,85	3
5	PR005	2	4,2	3	5	3	3,68	5

Sumber : Edi Faizal (2014 : 63)

II.5. *Microsoft Visual Basic 2010*

Microsoft Visual Basic.NET (VB.NET) adalah suatu pengembangan aplikasi bahasa pemrograman berbasis Visual Basic dan merupakan bahasa pemrograman terbaru buatan Microsoft setelah Microsoft Visual Basic 6.0. Pengembangan yang signifikan dari VB.NET ialah kemampuannya memanfaatkan platform NET, sehingga pengguna dapat membuat aplikasi Windows, aplikasi konsol, pustaka kelas, layanan NT, aplikasi web form, dan XML Web Service, yang secara keseluruhan memungkinkan integrasi tanpa batas dengan bahasa pemrograman lain sehingga berpeluang untuk berintegrasi dengan web. (Mulyani, Purnama, 2015 : 16).

Beberapa keunggulan lainnya yang dimiliki VB.NET, seperti memiliki penanganan debug yang baik sehingga pembangun aplikasi dapat mengetahui kesalahan kode yang terjadi secara cepat dan memiliki Windows form design yang memungkinkan pembangun/developer memperoleh aplikasi desktop dalam waktu singkat. VB.NET memiliki Interface Development Environment (IDE) yang lebih lengkap dan mudah bagi user pemula untuk mencari komponen atau objek yang kita inginkan, seperti menempelkan kontrol-kontrol yang terdapat pada toolbox, mampu memformat secara otomatis ukuran textbox, serta mengatur

property dari masing-masing kontrol. VB.NET juga memiliki .NET Framework. Microsoft .NET ialah sebuahplatform untuk membangun, menjalankan, dan meningkatkan generasi lanjut dari aplikasi terdistribusi, memperluas klien, server dan serviceservice. (Mulyani, Purnama, 2015 : 16).

II.6. *SQL Server 2008*

SQL *(Structured Query Language)* adalah sebuah bahasa yang dipergunakan umtuk mengakses data dalam basis data relasional. Bahasa ini secara *de facto* merupakan bahasa standar yang dipergunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya.

SQL terdiri dari dua bahasa, yaitu *Data Definition Language Manipulation Language (DML)*. Implementasi DDL dan DML sistem manajemen basis data *(SMBD),* namun secara umum implemen bahasa ini memiliki bentuk standar yang ditetapkan oleh *ANSI*. (Adelia, Setiawan : 2011 ; 115).

II.7. Pengertian Basis Data

Basis data adalah kumpulan data (arsip, atau file) yang saling berhubungan yang disimpan dalam media penyimpanan elektronis agar dapat dimanfaatkan kembali dangan cepat, dan mudah. Sedangkan sistem basis data adalah kumpulan file, atau tabel yang saling berhubungan yang memungkinkan beberapa pemakai, atau program lain untuk mengakses, dan memanipulasi file-file (tabel) tersebut (Kurniawan, 2015).

Sistem basis data mempunyai beberapa elemen penting yaitu (Oktafiansyah, 2012) :

1. Basis data sebagai inti dari sistem basis data.
2. Perangkat lunak (*software*) untuk perancangan dan pengelolaan basis data.
3. Perangkat keras (*hardware*) sebagai pendukung operasi pengolahan data.
4. Manusia (*brainware*) sebagai pemakai atau para spesialis informasi yang mempunyai fungsi sebagai perancang atau pengelola.

II.8. Normalisasi

Menurut Martin (1975), Normalisasi diartikan sebagai suatu teknik yang menstrukurkan/ mendekomposisi data dalam cara-cara tertentu untuk mencegah timbulnya permasalahan pengolahan data dalam basis data. Permasalahan yang dimaksud adalah berkaitan dengan penyimpangan-penyimpangan *(anomalies)* yang terjadi akibat adanya kerangkapan data dalam relasi dan in-efisiensi pengolahan (Sutanta ; 2011 : 174).

Proses normalisasi menghasilkan relasi yang optimal, yaitu (Martin, 1975) : (Sutanta ; 2011 : 175)

1. Memiliki struktur *record* yang konsisten secara logik;
2. Memiliki struktur *record* yang mudah untuk dimengerti;
3. Memiliki struktur *record* yang sederhana dalam pemeliharaan;
4. Memiliki struktur *record* yang mudah ditampilkan kembali untuk memenuhi kebutuhan pengguna;

5. Minimalisasi kerangkapan data guna meningkatkan kinerja sistem.

Secara berturut-turut masing-masing level normal tersebut dibahas berikut ini, dimulai dari bentuk tidak normal. (Sutanta ; 2011 : 176-179)

1. Relasi bentuk tidak normal (*Un Normalized Form* / UNF)

   Relasi-relasi yang dirancang tanpa mengindahkan batasan dalam defisi basis data dan karakteristik *Relational Database Management System* (RDBM) menghasilkan relasi *Un Normalized Form* (UNF). Bentuk ini harus di hindari dalam perancangan relasi dalam basis data. Relasi *Un Normalized Form* (UNF) mempunyai kriteria sebagai berikut.

   a. Jika relasi mempunyai bentuk *non flat file* (dapat terjadi akibat data disimpan sesuai dengan kedatangannya, tidak memiliki struktur tertentu, terjadi duplikasi atau tidak lengkap)
   b. Jika relasi membuat *set atribut* berulang *(non single values)*
   c. Jika relasi membuat *atribut non atomic value*

2. Relasi bentuk normal pertama (*First Norm Form* / 1NF)

   Relasi disebut juga *First Norm Form* (1NF) jika memenuhi kriteria sebagai berikut.

   a. Jika seluruh atribut dalam relasi bernilai *atomic (atomic value)*
   b. Jika seluruh atribut dalam relasi bernilai tunggal *(single value)*
   c. Jika relasi tidak memuat set atribut berulang
   d. Jika semua record mempunyai sejumlah atribut yang sama.

   Permasalahan dalam *First Norm Form* (1NF) adalah sebagai berikut.

a. Tidak dapat menyisipkan informasi parsial

b. Terhapusnya informasi ketika menghapus sebuah *record*

3. Bentuk normal kedua *(Second Normal Form /* 2NF)

Relasi disebut sebagai *Second Normal Form (*2NF) jika memenuhi kriteria sebagai berikut

a. Jika memenuhi kriteria *First Norm Form (*1NF)

b. Jika semua atribut nonkunci *Functional Dependence* (FD) pada *Primary Key* (PK)

Permasalahan dalam *Second Normal Form /* 2NF adalah sebagai berikut:

a. Kerangkapan data *(data redundancy)*

b. Pembaharuan yang tidak benar dapat menimbulkan inkonsistensi data *(data inconsistency)*

c. Proses pembaharuan data tidak efisien

Kriteria tersebut mengidentifikasikan bahwa antara atribut dalam *Second Normal Form* masih mungkin mengalami *Third Norm Form*. Selain itu, relasi *Second Normal Form* (2NF) menuntut telah didefinisikan atribut *Primary Key* (PK) dalam relasi. Mengubah relasi *First Norm Form* (1NF) menjadi bentuk *Second Normal Form* (2NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

a. Identifikasikan *Functional Dependence* (FD) relasi *First Norm Form* (1NF)

b. Berdasarkan informasi tersebut, dekomposisi relasi *First Norm Form* (1NF) menjadi relasi-relasi baru sesuai *Functional Dependence* nya. Jika menggunakan diagram maka simpul-simpul yang berada pada puncak diagram ketergantungan data bertindak *Primary Key* (PK) pada relasi baru

4. Bentuk normal ketiga (*Third Norm Form* / 3NF)

   Suatu relasi disebut sebagai *Third Norm Form* jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Second Normal Form* (2NF)
   b. Jika setiap atribut nonkunci tidak (TDF) *(Non Transitive Dependeny)* terhadap *Primary Key* (PK)

   Permasalahan dalam *Third Norm Form* (3NF) adalah keberadaan penentu yang tidak merupakan bagian dari *Primary Key* (PK) menghasilkan duplikasi rinci data pada atribut yang berfungi sebagai *Foreign Key* (FK) (duplikasi berbeda dengan keterangan data).

   Mengubah relasi *Second Normal Form* (2NF) menjadi bentuk *Third Norm Form* (3NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

   a. Identifikasi TDF relasi *Second Normal Form* (2NF)
   b. Berdasarkan informasi tersebut, dekomposisi relasi *Second Normal Form* (2NF) menjadi relasi-relasi baru sesuai TDF-nya.

5. Bentuk normal *Boyce-Cood (Boyce-Codd Norm Form* / BCNF)

Bentuk normal *Boyce-Codd Norm Form* (BCNF) dikemukakan oleh R.F. Boyce dan E.F. Codd. Suatu relasi disebut sebagai *Boyce-Codd Norm Form* (BCNF) jika memenuhi kriteria sebagai berikut.

a. Jika memenuhi kriteria *Third Norm Form* (3NF)
b. Jika semua atribut penentu (determinan) merupakan CK

6. Bentuk normal keempat (*Forth Norm Form* / 4NF)

   Relasi disebut sebagi *Forth Norm Form* (4NF) jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Boyce-Codd Norm Form*.
   b. Jika setiap atribut didalamnya tidak mengalami ketergantungan pada banyak nilai.

7. Bentuk normal kelima (*Fifth Norm Form* / 5NF)

   Suatu relasi memenuhi kriteria *Fifth Norm Form* (5NF) jika kerelasian antar data dalam relasi tersebut tidak dapat direkonstruksi dari struktur relasi yang sederhana.

8. Bentuk normal kunci domain (*Domain Key Norm Form* / DKNF)

   a. Relasi disebut sebagai *Domain Key Norm Form* (DKNF) jika setiap batasan dapat disimpulkan secara sederhana dengan mengetahui sekumpulan nama atribut dan domainnya selama menggunkan sekumpulan atribut pada kuncinya..

II.9. *Unified Modeling Language* (UML)

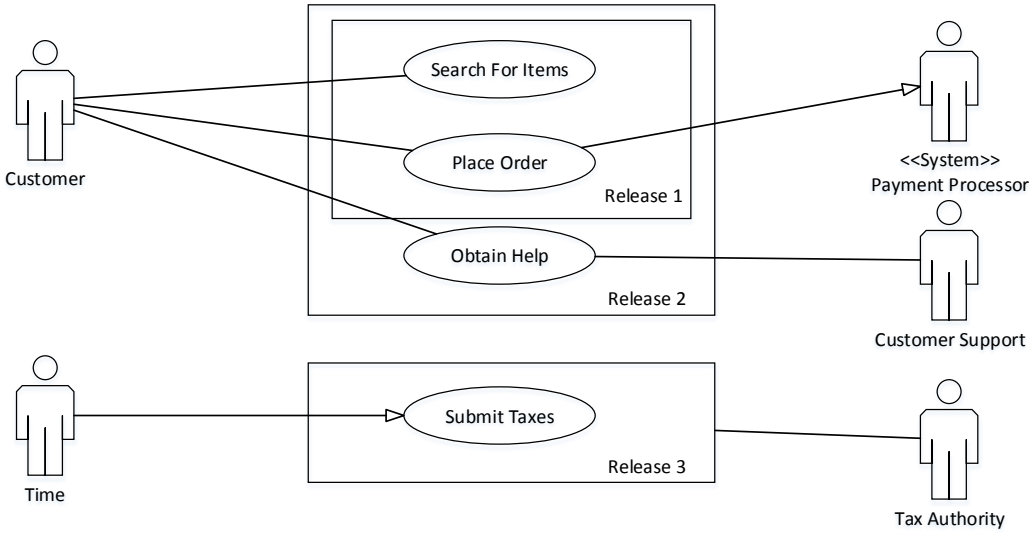
Unified Modeling Language (UML) adalah sebuah bahasa yang diterima dan digunakan oleh *software developer* dan *software analyst* sebagai suatu bahasa yang cocok untuk merepresentasikan grafik dari suatu relasi antar entitas-entitas software (Gornik, 2003). Dengan menggunakan UML, tim pengembang *software* akan mempunyai banyak keuntungan, seperti memudahkan komunikasi dengan sesama anggota tim tentang *software* apa yang akan dibuat, memudahkan integrasi ke dalam area pengerjaan *software* karena bahasa ini berbasiskan *meta-models* dimana *meta-models* bisa mendefinisikan proses-proses untuk mengkonstruksikan konsep-konsep yang ada. UML juga menggunakan formatur *input* dan *output* yang sudah mempunyai bentuk standar yaitu *XML Metadata Interchange* (XMI), menggunakan aplikasi dan pemodelan data yang universal, merepresentasikan dari tahap analisis ke implementasi lalu ke *deployment* yang terpadu, dan mendeskripsikan keutuhan tentang spesifikasi software.

UML menyediakan kumpulan alat yang sudah terstandarisasi, yang digunakan untuk mendokumentasikan analisis dan perancangan sebuah sistem perangkat lunak. (Kendall & Kendall, 2005, p663) Peralatan utama UML adalah diagram-diagram yang digunakan untuk membantu manusia dalam memvisualisasikan proses pengembangan sebuah sistem perangkat lunak, sama seperti penggunaan denah (*blueprint*) dalam pembuatan bangunan (Winata dan Setiawan, 2013).

II.9.1. *Use Case* Diagram

Diagram yang menggambarkan *actor*, *use case* dan relasinya sebagai suatu urutan tindakan yang memberikan nilai terukur untuk aktor. Sebuah *use case* digambarkan sebagai *elips horizontal* dalam suatu diagram UML *use case*. *Use Case* memiliki dua istilah, yaitu (Haviluddin, 2011) :

1. *System use case*; interaksi dengan sistem.
2. *Business use case*; interaksi bisnis dengan konsumen atau kejadian nyata.

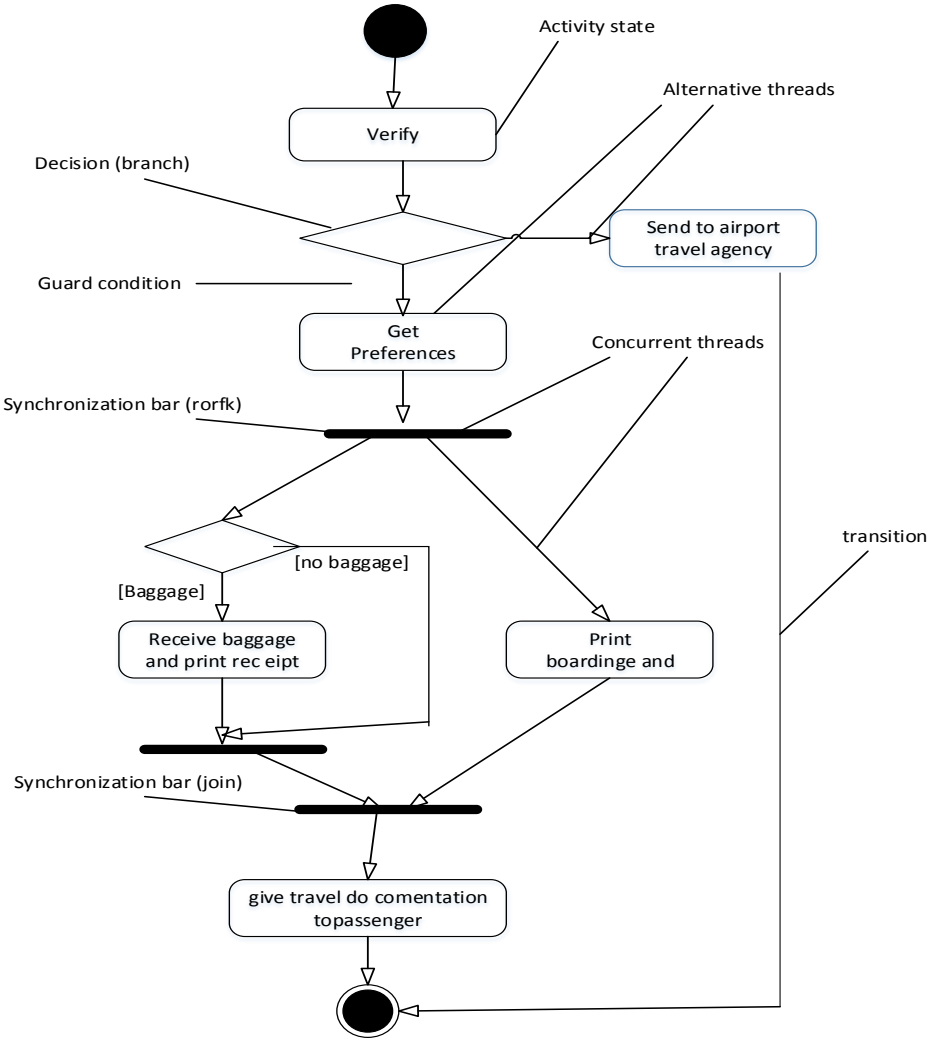


Gambar II.1. Notasi *Use Case* Diagram
(Sumber : Haviluddin, 2011)

II.9.2. *Activity* Diagram

Activity diagram menggambarkan aktifitas-aktifitas, objek, *state*, *transisi state* dan *event*. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku sistem untuk aktivitas (Haviluddin, 2011).

Teknik untuk menjelaskan *business process*, *procedural logic*, dan *work flow*. Bisa dipakai untuk menjelaskan teks *use case* dalam notasi grafis dengan menggunakan notasi yang mirip *flow chart*, meskipun terdapat sedikit perbedaan notasi (Winata dan Setiawan, 2013).



Gambar II.2. Notasi *Activity* Diagram

(Sumber : Haviluddin, 2011)

II.9.3. *Class* Diagram

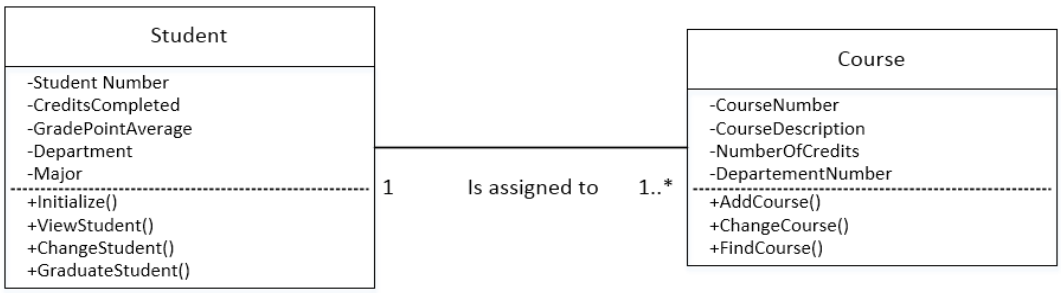
Class diagram merupakan diagram paling umum yang dijumpai dalam pemodelan berbasis UML. Didalam *Class* diagram terdapat *class* dan *interface* beserta atribut-atribut dan operasinya, relasi yang terjadi antar objek, *constraint* terhadap objek-objek yang saling berhubungan dan *inheritance* untuk organisasi *class* yang lebih baik. *Class* diagram juga terdapat *static view* dari elemen pembangun sistem. Pada intinya *Class* diagram mampu membantu proses pembuatan sistem dengan memanfaatkan konsep *forward* ataupun *reverse engineering* (Winata dan Setiawan, 2013).

Berbagai simbol yang hadir didalam *class* diagram antara lain adalah:

1. *Class*, yang berfungsi untuk merepresentasikan tipe dari data yang dimilikinya. *Class* diagram dapat ditampilkan dengan menunjukkan atribut dan operasi yang dimilikinya atau hanya menunjukkan nama *class*-nya saja. Dapat juga kita tuliskan nama *class* dengan atributnya saja atau nama *class* dengan operasinya.
2. *Attribute*, merupakan data yang terdapat didalam *class* dan *instance*-nya dengan operator.
3. *Operation*, berfungsi untuk merepresentasikan fungsi-fungsi yang ditampilkan oleh *class* dan *instance*-nya dengan operator.
4. *Association*, digunakan untuk menunjukkan bagaimana dua *class* berhubungan satu sama lainnya. *Association* ditunjukkan dengan sebuah garis yang terletak diantara dua *class*. Didalam setiap *association* terdapat *multiplicity,* yaitu simbol yang mengindikasikan

berapa banyak *instance* dari *class* pada ujung *association* yang satu dengan *instance class* di ujung *association* lainnya.

5. *Generalizations*, berfungsi untuk mengelompokkan *class* ke dalam hirarki *inheritance*.
6. *Aggregation*, merupakan bentuk khusus dari *association* yang merepresentasikan hubungan "*part-whole*". Bagian "*whole*" dari hubungan ini sering disebut dengan *assembly* atau *aggregate*. *Class* yang satu dapat dikatakan merupakan bagian dari *class* yang lain yang ikut membentuk *class* tersebut.
7. *Composition*, merupakan jenis *aggregation* yang lebih kuat diantara dua *class* yang memiliki *association* dimana jika *whole* ditiadakan, maka *part*-nya juga ikut ditiadakan. Berbeda dengan *aggregation*, *part* akan tetap bisa berdiri sendiri meskipun bagian *whole*-nya ditiadakan.
8. Penggunaan operator (+) dalam *class* diagram diartikan dengan *public*, operator (-) diartikan *private*, dan operator (#) diartikan *protected*.

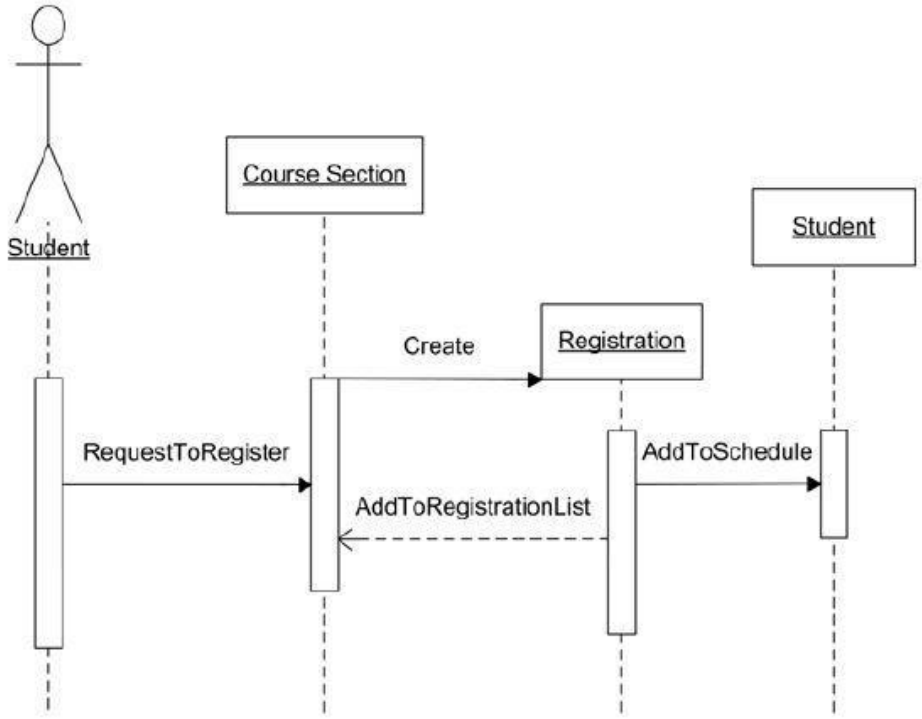


Gambar II.3. Contoh *Class* Diagram

(Sumber : Winata dan Setiawan, 2013)

II.9.4. *Sequence* Diagram

Menjelaskan interaksi obyek-obyek yang saling berkolaborasi (berhubungan), mirip dengan *activity* diagram yaitu menggambarkan alur kejadian sebuah aktivitas tetapi lebih detil dalam menggambarkan aliran data termasuk data atau behaviour yang dikirimkan atau diterima namun kurang mampu menjelaskan detil dari sebuah algoritma (Winata dan Setiawan, 2013).



Gambar II.4. Notasi *Sequence* Diagram
(Sumber : Winata dan Setiawan, 2013)