# BAB II

# LANDASAN TEORI

## II.1. Pengertian Sistem

Sistem yang abstrak adalah gagasan - gagasan atau konsepsi yang teratur yang saling bergantung. Sedangkan sistem yang bersifat fisik adalah serangkaian unsur yang bekerja sama untuk mencapai suatu tujuan (Gordon B. Davis, 2011).

Norman L. Enger menyatakan bahwa suatu sistem dapat terdiri atas kegiatan-kegiatan yang berhubungan berguna mencapai tujuan-tujuan perusahaan seperti pengendalian inventaris atau penjadwalan produksi (Tata Sutabri, 2012). Suatu sistem mempunyai karakteristik atau sifat - sifat tertentu, yaitu ;

1. Komponen sistem, bagian dari sistem yang saling bekerja sama membentuk suatu kesatuan.
2. Batasan sistem, daerah yang membatasi antara suatu sistem dengan sistem yang lainnya.
3. Lingkungan luas sistem, apapun diluar batasan dari sistem yang mempengaruhi operasi sitem.
4. Penghubungan sistem, media yang menghubungkan antara satu subsistem dengan subsistem lainnya.
5. Masukan sistem, energi dimasukkan kedalam sistem dapat berupa pemeliharaan sinyal.
6. Keluaran sistem, energi yang diolah dan diklasifikasi menjadi keluaran berguna.

### II.1.2. Pengertian Informasi

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterprestasikan untuk digunakan dalam proses pengambilan keputusan (Tata Sutabri, 2012). Sumber informasi adalah data. Data adalah kenyataan yang menggambarkan kejadian - kejadian dari kesatuan nyata. Informasi dihasilkan lebih berharga maka informasi harus memenuhi kriteria sebagai berikut :

1. Informasi harus tepat waktu, sehingga tidak ada keterlambatan ketika dibutuhkan.
2. Informasi harus relevan, benar - benar terasa manfaatnya bagi yang membutuhkan.
3. Informasi harus bernilai, informasi yang berharga untuk suatu pengambilan keputusan.
4. Informasi harus dipercaya, sehingga mendukung pihak manajemen dalam mengambil keputusan.

Kegunaan informasi adalah untuk mengurangi ketidak pastian didalam proses pengambilan keputusan tentang suatu keadaan. Nilai dari suatu informasi ditentukan dari dua hal yaitu manfaat dan biaya untuk mendapatkannya (Tata Sutabri, 2012).

## II.2. Pengertian Sistem Informasi

Sistem informasi dapat diartikan sebagai suatu sistem di dalam organisasi yang merupakan kombinasi dari orang-orang, fasilitas, teknologi, media,

prosedur-prosedur, dan pengendalian yang ditujukan untuk mendapatkan jalur kombinasi yang penting. (Sulindawati: 2011; 4)

Dalam suatu sistem informasi terdapat komponen - komponen sebagai berikut :

1. Perangkat keras *(hardware)*, mencakup berbagai peranti fisik seperti komputer dan printer.
2. Perangkat lunak *(software)* atau program, yaitu sekumpulan instruksi yang memungkinkan perangkat keras memproses data.
3. Prosedur, yaitu sekumpulan aturan yang dipakai untuk mewujudkan pemrosesan data dan pembangkitan keluaran yang dikehendaki.
4. Orang, yaitu semua pihak yang bertanggung jawab dalam pengembangan sistem informasi, pemerosesan dan penggunaan keluaran sistem informasi.
5. Basis data (*database*), yaitu sekumpulan *tabel*, hubungan dan lain-lain yang berkaitan dengan penyimpanan data.
6. Jaringan komputer dan komunikasi data, yaitu sistem penghubung yang memungkinkan sumber (*resource*) dipakai secara bersama atau diakses oleh sejumlah pemakai.

## II.3. Komponen Sebuah Sistem

Sistem informasi terdiri dari beberapa komponen, yaitu :

1. Masukan *(input)*

   Komponen yang mewakili data yang masuk kedalam sistem informasi. Input disini termasuk metode-metode dan media untuk menangkap data yang akan di masukkan yang dapat berupa dokumen - dokumen dasar.

2. Model

   Komponen ini terdiri dari kombinasi prosedur, logika dan model matematik yang akan memanipulasi data input dan data yang akan tersimpan dibaris data dengan cara yang sudah tertentu untuk menghasilkan keluaran yang diinginkan.

3. Keluaran *(output)*

   Yaitu produk dari suatu sistem informasi yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta pemakai sistem.

4. Teknologi

   Teknologi digunakan untuk menerima input, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem secara keseluruhan.

5. Basis Data

   Merupakan kumpulan dari data yang saling berhubungan satu dengan yang lain, tersimpan dalam perangkat keras komputer dan digunakan untuk memanipulasinya. Data di dalam basis data perlu di organisasikan

sedemikian rupa agar dihasilkan informasi yang berkualitas. Organisasi basis data yang baik juga digunakan untuk efisiensi kapasitas penyimpanan.

6. Kendali

Banyak hal yang dapat merusak informasi, seperti misalnya bencana alam, kecurangan-kecurangan, kegagalan sistem itu sendiri dan lain sebagainya. Beberapa pengendalian perlu dirancang dan diterapkan untuk meyakinkan bahwa hal-hal yang dapat dicegah, ataupun bila terlanjur terjadi langsung cepat diatasi.

## II.5. Inventaris

Menurut Chabib Sholeh dan Heru Rochamnsjah (2011: 180), Inventaris merupakan kegiatan/tindakan untuk melakukan penghitungan, pengurusan, penyelenggaraan peraturan, pencatatan data dan pelaporan barang milik daerah dalam unit pemakaian. Menurut A. Gima Sugiama ( 2013: 173), Inventarisasi aset adalah serangkaian kegiatan untuk melakukan pendataan, pencatatan, pelaporan hasil pendataan aset, dan mendokumentasikannya baik aset berwujud maupun aset tidak berwujud pada suatu waktu tertentu. Inventarisasi aset dilakukan untuk mendapatkan data seluruh aset yang dimliki, dikuasai sebuah organisasi perusahaan atau instansi pemerintah. Seluruh aset perlu diinventarisasi baik yang diperoleh berdasarkan beban dana sendiri ( investasi ), hibah ataupun dari cara lainnya. (Khusnul Hasan Nugroho: 20011; 28)

## II.6. Barcode

Bar *coding* adalah sebuah bentuk *artificial identifier*. *Barcode* merupakan sebuah kode mesin yang dapat di baca. *Barcode* terdiri dari sebuah bentuk bar dan spasi (hitam dan putih) dalam rasio yang didefenisikan yang mempresentasikan karakter *alphanumeric*. (Albert Haryadi :2011 ;2)

### II.6.1. Cara Kerja Barcode

Proses penggunaan *barcode* pada aplikasi Sistem Otomasi Perpustakaan dengan Barcode di SLTPI Al Azhar 8 Kemang Pratama dimulai dengan pembuatan label pada buku dan media yang ada di perpustakaan dengan langkah-langkah berikut :

1. Petugas Perpustakaan terlebih dahulu menentukan kode pada masing-masing buku. Seperti kode klasifikasi, kode sekolah, kode buku, kode rak dan kode cek buku.
2. Petugas Perpustakaan menginput kode yang telah ditentukan pada masing-masing buku ke sistem.
3. Sistem mengubah angka-angka yang diinputkan menjadi batangan *barcode* dalam bentuk gambar.
4. Gambar *barcode* yang sudah ada dicetak ke kertas stiker yang telah disediakan
5. Setelah *barcode* tercetak Petugas Perpustakaan menempelkan hasilnya ke setiap buku.

Setelah semua buku diberikan label pada Petugas Perpustakaan maka informasi terhadap buku tersebut dapat dimasukan ke dalam sistem sehingga

kinerja dari Petugas Perpustakaan menjadi lebih efisien. Dalam pembuatan label ini tipe data *barcode* yang digunakan yaitu EAN 8 dan EAN 13. Sebagai contoh :

1. Judul: Belajar Visual Basic 2010
2. Kode Perpustakaan :

1-2 Digit = Kode Sekolah

3-5 Digit = Kode Klasifikasi

6-9 Digit = Kode Buku/Nomer Urut

10-12 Digit = Kode Rak Buku

13 Digit = Cek Digit

Dikarenakan pembuatan label mengikuti aturan kode perpustakaan diatas maka label *barcode* yang dihasilkan menjadi **0200003014231**. (Endang Ripmiatin :2012 ; 4)

Adapun model-model barcode yang sering digunakan dan di temukan dalam kehidupan sehari-hari adalah seperti berikut :

1. QR Barcode, berikut contoh gambar QR barcode.

2. de, berikut contoh gambar Micro QR barcode.

3. Aztect Code, berikut contoh gambar Aztect Barcode.

4. Code One, berikut contoh gambar Code One Barcode.

5. Data Matrix, berikut contoh gambar data matrix barcode.

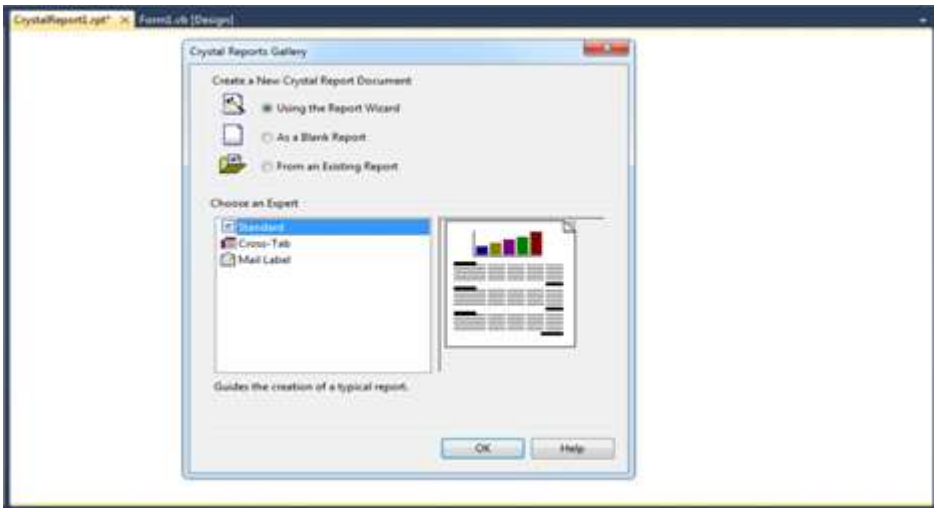
6. Grid Matrix, berikut contoh gambar data grid matrix barcode.

7. PDF-417, berikut contoh gambar data PDF-417 code.

8. Micro PDF-417, berikut gambar data Micro PDF-417 code.

## II.7. *Cristal Report*

*Crystal Report* adalah suatu aplikasi *windows* yang dikembangkan oleh *Seaget Software* yang berguna untuk membuat format laporan yang terpisah dari program *Microsoft Visual Studio* 2010, tetapi keduanya dapat dihubungkan. Mencetak laporan dengan *Crystal Report* hasilnya akan lebih baik dan lebih mudah untuk dikerjakan, karena pada *Crystal Report* banyak tersedia objek yang terdapat pada *Crystal Report* (Ardian M., 2013).




**Gambar II.9. Tampilan *Crystal Report***

(Sumber : Nursal: 2011)

Ada beberapa komponen yang terdapat pada *Crystal Report*, yaitu :

1. *Report Header*, merupakan tempat dimana biasanya judul laporan diletakkan atau informasi lain yang ingin kita munculkan diawal halaman.
2. *Page Header*, merupakan tempat dimana apabila kita ingin memunculkan informasi diatas setiap halaman, contohnya nama bab, nama dokumen atau informasi lainnya yang serupa.

3. *Details*, merupakan inti dari laporan, yang akan memunculkan setiap *record* dari *database*.
4. *Report Footer*, merupakan tempat apabila kita ingin memunculkan informasi hanya sekali disetiap akhir laporan, contoh *Grand* Total.
5. *Page Footer*, merupakan tempat dimana biasanya nomor halaman diletakkan atau informasi lain yang muncul dihalaman ini.

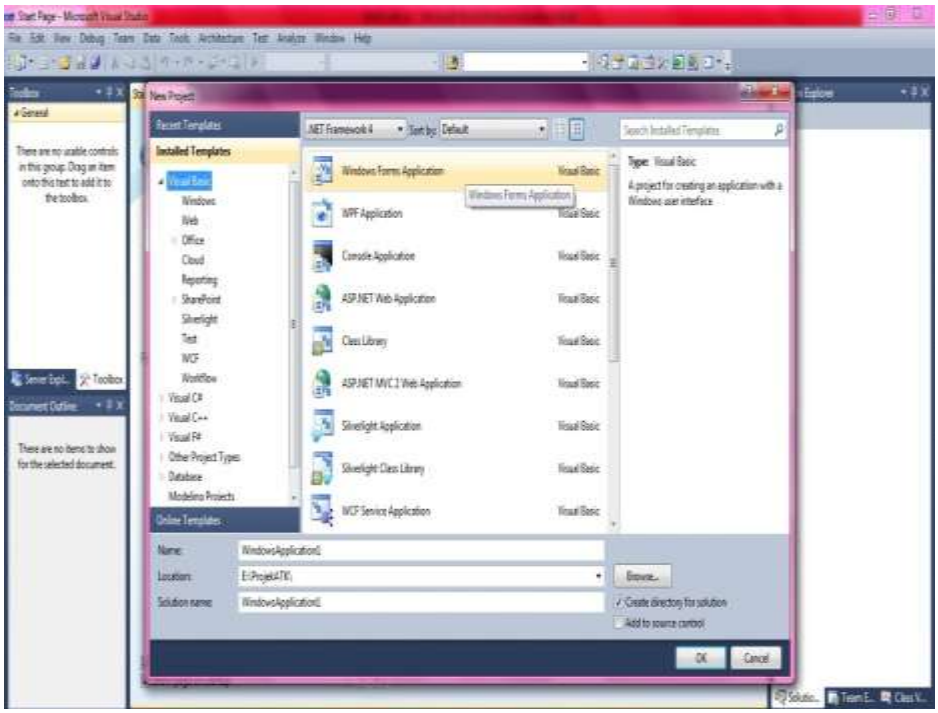
## II.8. *Microsoft Visual Basic 2010*

(Wahana Komputer 2011 : 3) Visual Basic 2010 merupakan salah satu bagian produk pemrograman terbaru yang dikeluarkan oleh Microsoft, yaitu Microsoft Visual Studio 2010. Visual Studio merupakan produk pemrograman andalan dari Microsoft Corporation, dimana didalamnya berisi beberapa jenis IDE pemrograman seperti *Visual Basic*, *Visual C++*, *Visual Web Developer*, *Visual C#*, dan *Visual F#*.

### II.8.1. Memulai *Microsoft Visual Studio* 2010

Untuk memulai pemrograman dengan *micorosoft visual basic* kita harus terlebih dahulu menjalankan program *microsoft visual studio* 2010. Selanjutnya pada tampilan awal akan ditampilkan kotak dialog *new project* seperti gambar berikut ini. Pada kotak dialog tersebut terdapat tab yang terdiri dari :

1. *New* (menampilkan daftar pilihan untuk membuat project baru).




**Gambar II.1. Kotak Dialog *New Project***

(Sumber : Handphan Rianto : 2012)

Untuk pembuatan program pertama kali pilih *new project*, pilih *Windows Forms Application* lalu ketik nama, klik OK.

### II.8.2.Komponen-Komponen Visual Studio 2010

1. Menu *Bar*

Menampilkan daftar menu yang berisi daftar perintah-perintah yang dapat digunakan saat bekerja pada visual basic. Terdiri dari menu *file, edit, view, project, build, debug, team, data, tools, architecture, test, analyze, window dan help*

.


**Gambar II.2. Menu *Bar***

(Sumber : Handphan Rianto: 2012)

2. *Toolbar*

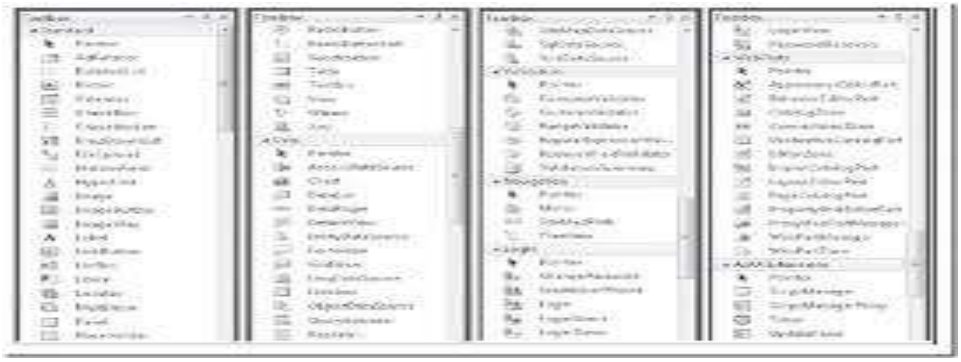
Digunakan untuk mengakses perintah-perintah dalam menu yang sering dipakai secara cepat.


**Gambar II.3. Menu *Toolbar***

**(**Sumber : Handphan Rianto: 2012)

3. *Toolbox*

Merupakan daftar komponen - komponen yang dapat digunakan untuk mendesain tampilan program aplikasi yang akan dibuat.



**Gambar II.4. *Toolbox***

**(**Sumber : Hadphan Rianto: 2012)

4. *Solution Explorer*

Menampilkan daftar *form* dan modul yang ada dalam project yang sedang aktif.


**Gambar II.5. Tampilan *Solution Explorer***

(Sumber : Handphan Rianto: 2012)

5. *Property Window*

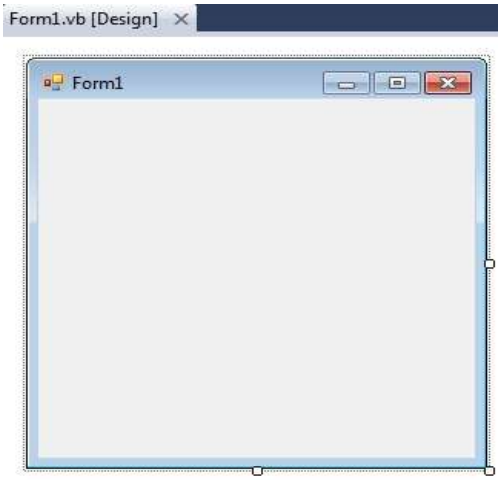
Digunakan untuk mengatur property dari komponen-komponen yang sedang diaktifkan. *Property* merupakan karakteristik dari sebuah objek. Berikut ini adalah tampilan dari *property form.*


**Gambar II.6. Tampilan *Property Window***

(Sumber : Handphan Rianto: 2012)

6. *Form Designer*

Merupakan jendela yang digunakan untuk melakukan perancangan tampilan dari aplikasi yang akan dibuat.

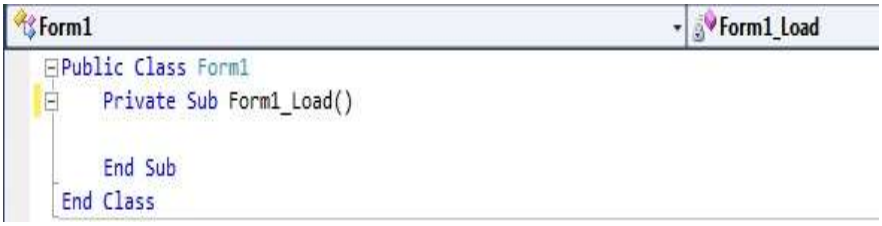



**Gambar II.7. Tampilan Awal *Form***

(Sumber : handphan Rianto: 2012)

7. *Code Window*

Merupakan jendela yang digunakan untuk menuliskan kode program.




**Gambar II.8. Jendela *Code Window***

(Sumber : Handphan Rianto: 2012)

## II.9. *SQL Server 2008 R2*

Menurut (Emma Utami & Sukrisno 2008 : 34) *SQL (Structured Query Language)* adalah bahasa komputer standar yang ditetapkan untuk mengakses dan

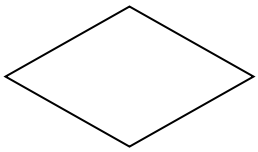
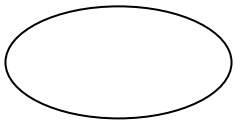
memanipulasi sistem *database*. SQL Server 2008 adalah teknologi yang mendukung *development* dan administrasi dari *Business Intellegence* (BI) *Aplication*. *SQL Server Reporting and Integration service* adalah element dari BI, tapi inti dari BI tersebut adalah SQL Server 2008 Analysis Servis (SSAS). *SQL Server 2008* adalah sebuah DBMS (*Database Management System*) yang dibuat oleh Microsoft untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan Oracle.( GI MDP : 2011 ; 2).

## II.10. *Entity Relational Diagram* (ERD)

Sebuah *entit*y adalah sebuah benda atau objek didunia yang dapat dibedakan dari semua objek lain nya *entity sets* adalah sekumpulan *entity* yang mempunyai tipe yang sama. Kesamaan tipe ini dapat dilihat dari atribut/*property* yang dimiliki oleh *entity*. Dibawah ini merupakan gambar dari ERD dan keterangannya.

**Tabel II.4. *Entity Relational Diagram* (ERD*)***

| Simbol | Keterangan |
|---|---|
|  | Entitas adalah suatu objek yang dapat didefenisikan dalam lingkungan pemakai. |

| | |
|---|---|
|  | Relasi menunjukkan adanya hubungan diantara sejumlah entitas yang berbeda. |
|  | Atribut yang berfungsi sebagai key. |
|  | Sebagai penghubung antara himpunan relasi. |

**(Sumber : Stevalin Betshani: 2013)**

## II.11. Normalisasi

Normalisasi adalah suatu teknik untuk memecah/ mendekomposisi data dalam cara-cara tertentu untuk mencegah timbulnya permasalahan pengolahan data dalam basis data dalam bisnis data. Proses normalisasi terdiri dari beberapa level, yaitu :

1. Bentuk Tidak Normal (*Un Normalized Form*/UNF). Kriteria dari bentuk ini adalah :
   a. Jika relasi mempunyai bentuk nonflat file.
   b. Jika relasi memuat set attribut berulang.
   c. Jika relasi memuat attribut non atomic value.
2. Bentuk Normal Pertama (*First* Normal *Form*/1NF). Kriteria dari bentuk ini adalah :

a. Jika seluruh attribut dalam relasi bernilai atomaik.

b. Jika seluruh attribut dalam relasi bernilai tunggal.

c. Jika relasi tidak memuat set attribut berulang.

d. Jika semua *record* mempunyai sejumlah attribut yang sama.

Permasalahan dalam adalah :

a. Tidak dapat menyisipkan informasi parsial.

b. Terhapusnya informasi ketika menghapus sebuah record.

c. Pembaharuan attribut non kunci mengakibatkan sejumlah record harus diperbaharui.

Untuk mengubah relasi UNF menjadi bentuk 1NF dilakukan dengan cara :

a. Melengkapi nilai-nilai dalam attribut.

b. Mengubah struktur relasi.

3. Bentuk Normal Kedua (Second Normal Form/2NF). Kriteria dari bentuk ini adalah :

a. Jika memenuhi kriteria 1NF

b. Jika semua attribut non kunci memiliki ketergantungan fungsional terhadap attribut kunci.

Permasalahan dalam 2NF adalah :

a. Kerangkapan data.

b. Pembaharuan yang tidak benar dapat menimbulkan inkonsistensi data.

c. Proses pembaharuan data tidak efisien.

d. Penyimpanan pada saat penyisipan, penghapusan dan pembaharuan.

Untuk mengubah relasi 1NF menjadi bentuk 2NF dilakukan dengan cara :

a. Identifikasi ketergantungan fungsional pada relasi 1NF.
b. Berdasarkan ketergantungan fumgsional, dekomposisi relasi 1NF menjadi relasi-relasi baru sesuai dengan ketergantungan fungsionalnya.

4. Bentuk normal Ketiga (*Third* Normal *Form*/3NF). Kriteria dari bentuk ini adalah :

a. Jika memenuhi kriteria 3NF.
b. Jika semua attribut non kunci tidak memiliki ketergantungan transitif kunci. (Putu Manik Prihatin: 2012 ; 5-6).

## II.12. *Unified Modelling Language* (UML)

Pengembangan sistem adalah aktivitas manusia. Tanpa adanya kemudahan untuk memahami sistem notasi, proses pengembangan kemungkinan besar akan mengalami kesalahan. UML adalah sistem notasi yang sudah dibakukan di dunia pengembangan sistem, hasil kerjasama dari Grady Booch, James Rumbaugh dan Ivar Jacobson. UML yang terdiri dari serangkaian diagram memungkinkan bagi sistem analis untuk membuat cetak biru sistem yang komperhesif kepada klien, programmer dan tiap orang yang terlibat dalam proses pengembangan tersebut. Dengan UML akan dapat menceritakan apa yang seharusnya dilakukan oleh sistem bukan bagaimana yang seharusnya dilakukan oleh sebuah sistem.

UML adalah bahasa untuk menspesifikasi, memvisualisasi, membangun dan mendokumentasikan bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya. UML tidak hanya merupakan sebuah bahasa pemograman visual saja, namun juga dapat secara langsung duhubungkan keberbagai bahasa pemograman seperti JAVA, C++, Visual Basic atau bahkan dihubungkan secara langsung kedalam sebuah *object oriented database*.Begitu juga mengenai pendokumentasian dapat dilakukan seperti *requitments*, arsitektur, *design*, *source*, *project plan*, *tests* dan *prototypes*. UML memiliki 8 tipe diagram, namun pada penulisan skripsi ini penulis akan menggunakan 4 tipe diagram UML yaitu *use case diagram, sequence diagram, activity diagram* dan *class diagram.*

**II.12.1.Tipe Diagram UML**

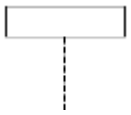
Adapun beberapa simbol diagram yang akan sering digunakan adalah UML sperti berikut :

1. *Sequence Diagram*

*Sequence diagram* menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *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. Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal.

**Tabel II.3. *Sequence Diagram***

| Gambar | Nama | Keterangan |
|---|---|---|
|  | *LifeLine* | Objek *entity*, antarmuka yang saling berinteraksi. |
|  | *Message* | Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi |
|  | *Message* | Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi |

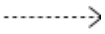
**(Sumber : Adi Nugroho: 2011)**

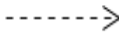
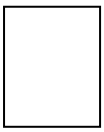
2. *Use Diagram*

*Use case diagram* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau

mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

**Tabel II.1. *Use Case Diagram***

| Gambar | Nama | Keterangan |
|---|---|---|
|  | *Actor* | Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*. |
|  | *Dependency* | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*). |
|  | *Generalization* | Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek |

| | | |
|---|---|---|
| | | induk (*ancestor*). |
|  | *Include* | Menspesifikasikan bahwa *use case* sumber secara *eksplisit*. |
|  | *Extend* | Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan. |
|  | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
|  | *System* | Menspesifikasikan paket yang menampilkan sistem secara terbatas. |
|  | *Use Case* | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor |
|  | *Collaboration* | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi). |
|  | *Note* | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi |

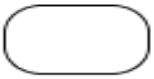
**(Sumber : Adi Nugroho: 2011)**

3. *Activity Diagram*

*Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *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.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

**Tabel II.2. *Activity Diagram***

| Gambar | Nama | Keterangan |
|---|---|---|
|  | *Actifity* | Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain |
|  | *Action* | State dari sistem yang mencerminkan eksekusi dari suatu aksi |
|  | *Initial Node* | Bagaimana objek dibentuk atau diawali. |
|  | *Actifity Final Node* | Bagaimana objek dibentuk dan dihancurkan |
|  | *Fork Node* | Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran |

**(Sumber : Adi Nugoroho, 2011)**