BAB II

LANDASAN TEORI

II.1. Sistem

Secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel yang terorganisasi, saling berinteraksi, saling tergantung satu sama lain dan terpadu. Teori sistem secara umum pertama kali di uraikan oleh Kenneth Boulding, terutama menekankan pentingnya perhatian terhadap setiap bagian yang membentuk sebuah sistem. Kecenderungan manusia yang mendapat tugas untuk meminpin organisasi adalah dia terlalu memusatkan perhatiannya pada salah satu komponen sistem organisasi. Teori sistem mengadakan bahwa unsur pembentuk organisasi itu penting dan harus mendapat perhatian yang utuh supaya manajer dapat bertindak lebih efektif. Unsur atau komponen pembentuk organisasi disini bukan hanya bagian-bagian yang tampak secara fisik, tetapi juga hal-hal yang mungkin bersifat abstrak atau konseptual, seperti misi, pekerjan, kegiatan, kelompok informal, dan lain sebagainya. (Tata Sutabri; 2012: 3).

II.1.1. Karakteristik Sistem

Model umum sebuah sistem terdiri dari input, proses, dan output. Hal ini merupakan konsep sebuah sistem yang sangat sederhana mengingat sebuah sistem dapat mempunyai beberapa masukan dan keluaran sekaligus. Selain itu sebuah sistem juga memiliki karekteristik atau sifat-sifat tertentu, yang

mencirikan bahwa hal tersebut bisa dikatakan sebagi suatu sistem. Adapun karakteristik yang dimaksud adalah sebagai berikut:

a. Komponen Sistem (*Component*)

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang bekerja sama membentuk satu kesatuan. Komponen-koponen sistem tersebut dapat berupa suatu bentuk subsistem. Setiap subsistem memiliki sifat-sifat sistem yang menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai sistem yang lebih besar yang disebut dengan supra sistem.

b. Batasan Sistem (*Boundary*)

   Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem lainnya atau sistem dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan yang tidak dapat dipisah-pisahkan.

c. Lingkungan Luar Sistem (*Environtment*)

   Bentuk apapun yang ada di luar ruang lingkup atau batasan sistem yang mempengaruhi operasi sistem tersebut disebut dengan lingkungan luar sistem. Lingkungan luar sistem ini dapat menguntungkan dan dapat juga merugikan sistem tersebut. Lingkungan luar yang menguntungkan merupakan energi bagi sistem tersebut, yang dengan demikian lingkungan luar tersebut harus selalu dijaga dan dipelihara. Sedangkan lingkungan luar yang merugikan

harus dikendalikan. Kalau tidak maka akan mengganggu kelangsungan hidup sistem tersebut.

d. Penghubung Sistem (*Interface*)

Media yang menghubungkan sistem dengan subsistem yang lain disebut dengan penghubung sistem atau *interface*. Penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem yang lain. Keluaran suatu subsistem akan menjadi masukan untuk subsistem yang lain dengan melewati penghubung. Dengan demikian terjadi suatu integrasi sistem yang membentuk atu kesatuan.

e. Masukan Sistem (*Input*)

Energi yang dimasukan ke dalam sistem disebut masukan sistem, yang dapat berupa pemeliharaan (*maintenance input*) dan sinyal (*signal input*).

f. Keluaran Sistem (*Output*)

Hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna. Keluaran ini merupakan masukan bagi subsistem yang lain.

g. Pengolah Sistem (*Procces*)

Suatu sistem dapat mempunyai suatu proses yang akan mengubah masukan menjadi keluaran.

h. Sasaran Sistem (*Objective*)

Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat deterministik. Kalau suatu sistem tidak memiliki sasaran, maka

operasi sistem tidak ada gunanya. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuan yang telah direncanakan. (Tata Sutabri; 2012: 13-14)

II.1.2. Klasifikasi Sistem

Sistem merupakan suatu bentuk integrasi antara satu komponen dengan komponen lain karena sistem memiliki sasaran yang berbeda untuk setiap kasus yang terjadi di dalam sistem tersebut. Oleh karena itu sistem dapat diklasifikasikan dari beberapa sudut pandangan.

a. Sistem abstrak dan sistem fisik

   Sistem abstrak adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik.

   Sistem fisik merupakan sistem yang ada secara fisik.

b. Sistem Alamiah dan sistem buatan manusia

   Sistem alamiah adalah sistem yang terjadi melalui proses alam, tidak dibuat oleh manusia.

   Sistem buatan manusia merupakan sistem yang melibatkan hubungan manusia dengan mesin, yang disebut *human machine system.*

c. Sistem Deterministik dan sistem probabilistik

   Sistem yang beroperasi dengan tingkah laku yang dapat diprediksi disebut sistem deterministik.

   Sistem probabilistik adalah sistem yang kondisi masa depannya tidak dapat diprediksi, karena mengandung unsur probabilitas.

d. Sistem Terbuka dan sistem tertutup

Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh oleh lingkungan luarnya. Sistem ini bekerja secara otomatis tanpa campur tangan dari puhak luar. Sedangkan sistem terbuka adalah sistem yang berhubungan dan dipengaruhi oleh lingkungan luarnya, yang menerima masukan dan menghasilkan keluaran untuk subsistem lainnya. (Tata Sutabri; 2012: 15)

II.2. Informasi

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasikan untuk digunakan dalam proses pengambilan keputusan. Bila tidak ada pilihan atau keputusan maka informasi tidak diperlukan. (Tata Sutabri; 2012: 22)

II.3. Sistem Informasi

Sistem informasi adalah suatu sistem di dalamnya suatu organisasi yang mempertemukan kebutuhan pengelolahan harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan stategi dari suatu organisasi untuk dapat menyediakan laporan – laporan yang diperlukan oleh pihak luar tertentu. (Tata Sutabri; 2012: 38)

II.4. Data

Data merupakan bentuk mentah yang belum dapat bercerita banyak sehingga perlu di olah lebih lanjut. Data diolah melalui suatu model agar menghasilkan informasi. (Tata Sutabri; 2012: 25)

II.5. Sistem Pendukung Keputusan (SPK)

Sistem pendukung keputusan (SPK) atau dikenal dengan *Decision Support System (DSS)* pad tahun 1970-an sebagai pengganti isttila *management information systems* (*MIS*) . Tetapi pada dasarnya SPK merupakan pengembangan lebih lanjut dari MIS yang dirancang sedemikian rupa sehingga bersifat interaktif dengan pemakianya, maksud dan tujuan dari adanya sistem pendukung keputusan yaitu untuk mendukung pengambil keputusan memilih *alternative* keputusan yang merupakan hasil pengolahan informasi-informasi yang diperoleh dengan menggunakan model-model pengambilan keputusan serta menyelesaikan masalah-masalah bersifat terstruktur, semi terstruktur dan tidak terstruktur. (Jurnal Informasi dan teknologi ilmiah (inti), Vol:II, Nomor :1 ;Februari 2014:12)

Sistem penunjang atau pendukung keputusan (SPK) didefenisikan sebagai suatu sistem informasi untuk membantu manajer proses pengambilan setengah terstruktur (*semi structured*) supaya lebih efektif dengan menggunakan model-model analisi dan data yang tersedia.

Dari defenisi diatas, maka dapat diketahui tujuan SPK adalah sebagai berikut :

1. Membantu manager mengambil keputusan setengah terstruktur yang dihadapi oleh manajer level menengah.
2. Membantu atau mendukung manajemen mengambil keputusan bukan menggantikannya.

3. Meningkatkan efektifitas pengambilan keputusan manajemen bukan untuk meningkatkan efesiensi. Walaupun waktu manajer penting (efesiensi), tetapi efektifitas merupakan tujuan utama pengguna SPK. (Jurnal Pelita Informatika Budi Darma, Volume : VI , Nomor: April 2014:49-50) .

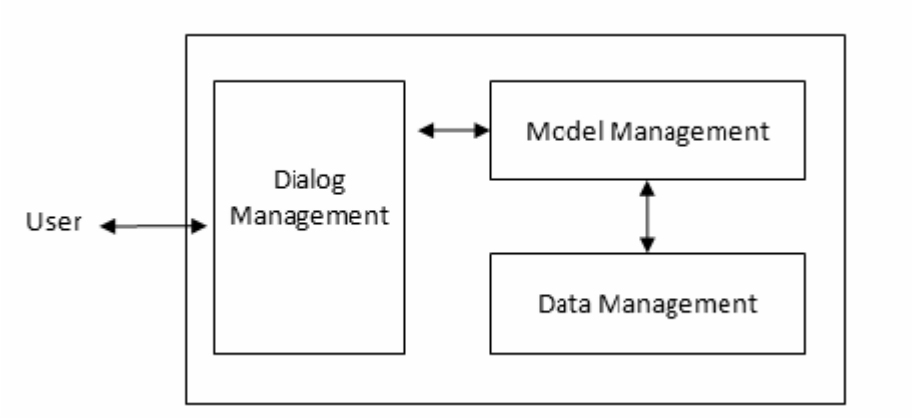
II.5.1. Karakteristik dan Kemampuan SPK.

1. SPK menyediakan dukungan bagi pengambil keputusan utamanya pada situasi semi terstruktur dan tak terstruktur dengan memadukan pertimbangan manusia dan informasi terkomputerisasi. Berbagai masalah tidak dapat diselesaikan atau tidak dapat diselesaikan secara memuaskan oleh sistem terkomputerisasi lain seperti EDP (*Electronic Data Processing*) atau MIS (*Management Information System*), tidak juga dengan metode atau *tool* kuantitatif standar.
2. Dukungan disediakan untuk berbagai level manajerial yang berbeda, mulai dari pimpinan puncak sampai manajer lapangan.
3. Dukungan disediakan bagi individu dan juga bagi *group.* Berbagai masalah organisasi melibatkan pengambilan keputusan dari orang dalam *group*. Untuk masalah yang strukturnya lebih sedikit seringkali hanya membutuhkan keterlibatan beberapa individu dari departemen dan level organisasi yang berbeda.
4. SPK menyediakan dukungan ke berbagai keputusan yang berurutan atau saling berkaitan.

5. SPK mendukung berbagai fase proses pengambilan keputusan : *intelligence, design, choice* dan *implementation*. (Fahmi Maulana Volume : VI, Nomor: 3, April 2014).

II.5.2. Komponen SPK

Sistem Pendukung Keputusan mempunyai 3 komponen utama yaitu dialog manajemen, model manajemen dan data manajemen seperti terlihat pada gambar :



Gambar II.1. Komponen SPK

Sumber : (Jurnal Fahmi Maulana Volume : VI, Nomor: 3, April 2014)

Ketiga komponen ini merupakan komponen utama dari Sistem Pendukung Keputusan. Komponen pertama adalah dialog management atau user interface yaitu komponen untuk berdialog dengan pemakai sistem. Komponen ini didalah sistem informasi merupakan komponen input dan komponen output.

Komponen kedua dari SPK adalah model management, yaitu komponen yang merubah data menjadi informasi yang relevan. Model-model

yang banyak digunakan di Sistem Pendukung Keputusan adalah model matematik optimisasi seperti linier programming, dynamic programming dan lain sebagainya. Komponen ketiga adalah data management, yaitu komponen basis data yang terdiri dari semua basis data yang dapat diakses. Seperti halnya sistem informasi pada umumnya, sistem pendukung keputusan juga mempunyai komponen teknologi dan kontrol. Komponen teknologi terdiri dari perangkat keras dan perangkat lunak. Perangkat lunak spesifik yang digunakan oleh SPK misalnya spreadsheet, DBMS, bahasa query. (Fahmi Maulana Volume : VI, Nomor: 3, April 2014).

II.6. Metode *Simple Additive Weighting* (SAW)

Metode SAW merupakan metode MADM yang paling sederhana dan paling banyak digunakan. Metode ini juga metode yang paling mudah untuk diaplikasikan, karena mempunyai algoritma yang tidak terlalu rumit. Metode SAW sering juga dikenal sebagai metode penjumlahan terbobot. Konsep dasar metode SAW adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut. Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternatif yang ada.

$$r_{ij} = \begin{cases} \dfrac{x_{ij}}{\underset{i}{Max}\ x_{ij}} & jika\ \text{j adalah atribut keuntungan (benefit)} \\ \dfrac{\underset{i}{Min}\ x_{ij}}{x_{ij}} & \text{jika j adalah atribut biaya (cost)} \end{cases}$$

Gambar II.2. Formula untuk mencari normalisasi

Sumber *:* (Jurnal SPK Kelayakan Pemberian Kredit Motor : 2014)

Dimana :

Rij : Rating kinerja ternormalisasi

Maximum: Nilai maksimum dari setiap baris dan kolom

Minimum: Nilai minimum dari setiap baris dan kolom

Xij: Baris dan kolom dari matriks

Dimana rij adalah rating kinerja ternormalisasi dari alternatif A pada atribut Cj; i=1,2,...,m dan j=1,2,...,n.

Nilai preferensi untuk setiap alternative (Vi) diberikan sebagai :

$$V_i = \sum_{j=1}^{n} w_j r_{ij}$$

Gambar II.3. Formula untuk mencari nilai preverensi

Sumber : (Jurnal SPK Kelayakan Pemberian Kredit Motor : 2014)

Vi : Nilai Akhir Alternative

Wi : Bobot yang telah ditentukan

Rij : Normalisasi matriks

Nilai Vi yang lebih besar mengindikasikan bahwa alternatif A, lebih terpilih. (Destriyana Darmastuti : 2010).

II.6.1. Langkah – langkah Metode SAW

Adapun langkah – langkah dari metode SAW, sebagai berikut :

1. Menentukan kriteria – kriteria yang akan dijadikan acuan dalam pengambilan keputusan, yaitu c
2. Menentukan rating kecocokan setiap alternatif pada setiap kriteria
3. Membuat matriks keputusan berdasarkan kriteria (C), kemudian melakukan normalisasi matriks berdasarkan persamaan yang disesuaikan dengan jenis atribut (atribut keuntungan ataupun atribut biaya) sehingga diperoleh matriks ternormalisasi R, (Destriyana Darmastuti : 2010).

II.6.2. Kelebihan Metode *Simple Adiditive Weighting* (SAW)

Kelebihan dari model *Simple Additive Weighting* (SAW) dibandingkan dengan model pengambilan kepnnya untuk melakukan penilaian secara lebih tepat karena didasarkan pada nilai kriteria dan bobot preferensi yang sudah ditentukan, selain itu SAW juga dapat menyeleksi alternatif terbaik dari sejumlah alternatif yang ada karena adanya proses perankingan setelah menetukan nilai bobot untuk setiap atribut, (Destriyana Darmastuti : 2010).

II.7. Normalisasi

Normalisasi adalah proses dimana tabel – tabel pada *database* dites dalam hal kesalingtergantungan di antara *field – field* pada sebuah tabel. Misalnya jika pada sebuah tabel terdapat ketergantungan terhadap lebih dari satu *field* dalam tabel tersebut, maka tabel tersebut harus dipecah menjadi banyak tabel. Setiap tabel hanya boleh memiliki sebuah field kunci yang menjadi ketergantungan dari *field* lainnya dalam tabel tersebut.

Ada beberapa langkah dalam normalisasi tabel yaitu :

1. Dekomposisi (*Decomposition*)

   Dekomposisi adalah proses mengubah bentuk tabel supaya memenuhi syarat tertentu sebagai tabel yang baik. Dekomposisi dapat dikatakan berhasil jika tabel yang dikenai dekomposisi bila digabungkan kembali dapat menjadi tabel awal sebelum di dekomposisi. Dekomposisi akan sering dilakukan dalam proses normalisasi untuk memenuhi syarat – syaratnya.

2. Bentuk Tidak Normal

   Pada bentuk ini semua data yang ada pada tiap *entity* (diambil atributnya) masih ditampung dalam satu tabel besar. Data yang ada pada tabel inimasih ada yang redundasi dan ada juga yang kosong. Semuanya masih tidak tertata rapi.

3. Normal *Form* Pertama (*1st Normal Form*)

   Pada tahapan ini tabel di-dekomposisi dari tabel bentuk tidak normalyang kemudian dipisahkan menjadi tabel – tabel kecil yang memiliki kriteria

tidak memiliki atribut yang bernilai ganda dan komposit. Semua atribut harus bersifat atomik.

4. Normal *Form* Kedua (*2nd Normal Form*)

   Pada tahapan ini tabel dianggap memenuhi normal kedua jika pada tabel tersebut semua atribut yang bukan kunci primer bergantung penuh terhadap kunci primer tabel tersebut.

5. Normal *Form* Ketiga (*3rd Normal Form*)

   .Setiap atribut pada tabel selain kunci primer atau kunci utama harus bergantung penuh pada kunci utama. Bentuk normal ketiga biasanya digunakan bila masih ada tabel yang belum efisien. (Wahana Komputer, 2010:32-35).

II.8. Basis Data (*Database*)

Database atau basis data adalah sekumpulan data yang memiliki hubungan sara logika dan diatur dengan susunan tertentu serta disimpan dalam media penyimpanan komputer. Data itu sendiri adalah data representasi dari semua fakta yang ada pada dunia nyata yang ada. *Database* sering digunakan untuk melakukan proses terhadap data-data tersebut untuk menghasilkan informasi tertentu. Misalnya dari data nama siswa dan tanggal lahir siswa anda bisa mendapatkan informasi nama siswa yang berulang tahun pada hari ini. Tentu saja informasi tersebut akan anda dapatkan dari *software* pemrosesan *database* dengan cara anda memberikan perintah dalam bahasa tertentu yatu *SQL (Structured Query Language).*

Dalam *database* ada sebutan-sebutan untuk satuan data yaitu :

a. Karakter, ini adalah satuan data terkecil. Data terdiri atas susuna karakter yang pada akhirnya mewakili data yang memiliki arti dari sebuah fakta.
b. *Field* , adalah kumpulan dari karakter yang mewakili fakta tertentu misalnya seperti nama mahasiswa, tanggal lahir, dan lain-lain. Dalam dunia perancangan database, *field* juga disebut atribut. Bila dipandang dari sudu pemrograman berorientasi obyek aka sebuah *field* akan memiliki dua properti utama yaitu properti name dan *property type* Properti name atau nama adalah properti dari *field* yang berisi nama *field* yang mewakili data sejenis yang disimpan. Sedangkan *property type* adalah properti yang mengatur tipe data dari data yang akan ditampungnya. Misalnya nama *field*nya adalah nama siswa maka tipe datanya adalah char, bila nama *field*nya adalah tanggal lahir maka tipe datanya adalah *date*. *Field* dilihat seperti kolom.
c. *Record*, adalah kumpulan dari field. Pada *record* anda dapat menemukan banyak sekali informasi penting dengan cara mengkombinasikan *field-field* yang ada.
d. Tabel, adalah sekumpulan dari *record-record* yang memiliki kesamaan entity dalam dunia nyata. Kumpuan dari tabel adalah *database*, wujud fisik dalam *database* dalam sebuah komputer adalah sebuah file yang didalamnya terdapat berbagai tingktan data yang telah disebutkan diatas.

e. *File*, adalah bentuk fisik dari penyimpanan data file. *Database* berisi semua data yang telah disusun dan diorganisasikan sedemikian rupa sehingga memudahkan pemberian informasi.

Sistem *database* adalah sebuah kumpulan dari komponen-komponen *database-database* yang meliputi :

1. *Database*
2. *Database server*
3. Komponen *client software*
4. Aplikasi *database*

Aplikasi *database* adalah sebuah *software* khusus yang di desain dan digunakan olh user atau pihak lainya seperti penyedia jasa pemrograman atau konsultan. Sedangkan *client server* adalah salah satu komponen yang termasuk dalam sistem *database* yang memampukan *software* aplikasi *database* mengakses data secara *remote* pada sebuah *server database*. (Wahana Komputer, 2010:24-25).

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

Unified Modelling Language (UML) adalah sebuah alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual. Juga merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentuk atau menggambarkan sebuah sistem software yang terkait dengan objek.

Sejarah UML sendiri terbagi dalam dua fase ; sebelum dan sesudahnya munculnya UML. Dalam fase sebelum, UML sebenarnya sudah mulai diperkenalkan sejak tahun 1990-an namun notasi yang dikembangkan oleh parah ahli analisi dan desain berbeda-beda, sehingga daat dikatakan belum memiliki standarisasi.

Fase kedua; dilandasi dengan pemikiran untuk mempersatukan metode tersebut dan dimotori oleh *Object Management Group* (OMG) maka pengembangan UML dimulai pada akhir tahun 1994 ketika *Grady Booch* dengan metode OOD (*Object-Oriented Design*), Jim Rumbaugh dengan metode OMT (*Object Modelling Technique*) mereka ini bekerja pada Rasional Software Corporation dan Ivar Jacobson dengan metode OOSE (*Object-Oriented Software Engineering*) yang bekerja pada perusahaan Objectory Rasional.

Sebagai pencetus metode-metode tersebut mereka bertiga berinisiatif untuk menciptakan bahasa pemodelan terpadu sehingga pada tahun 1996 mereka berhasil merilis UML versi 0.9 dan 0.91 melalui *Request for Proposal* (RFP) yang dikeluarkan oleh *OMG (Braun, et.al*. 2001).

Kemudian pada Januari 1997 IBM, *ObjecTime*, *Platinum Technology, Ptech, Taskon, Reich Technologies* dan *Softeam* juga menanggapi *Request for Proposal (RFP)* yang dikeluarkan oleh *OMG* tersebut dan menyatakan kesediaan untuk bergabung.

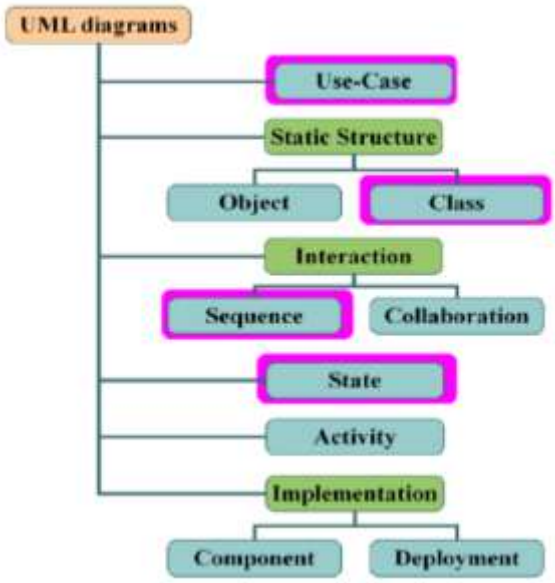
Perusahaan-perusahaan ini menyumbangkan ide-ide mereka, dan bersama para mitra menghasilkan UML revisi 1.1. Fokus dari UML versi rilis 1.1 ini

adalah untuk meningkatkan kejelasan UML Semantik versi rilis 1.0. Hingga saat ini UML versi terbaru adalah versi 2.0 (http://www.uml.org/).

Saat ini sebagian besar para perancang sistem informasi dalam menggambarkan informasi dengan memanfaatkan UML diagram dengan tujuan utama untuk membantu tim proyek berkomunikasi, mengeksplorasi potensi desain, dan memvalidasi desain arsitektur perangkat lunak atau pembuat program.

Secara filosofi UML diilhami oleh konsep yang telah ada yaitu konsep permodelan *Object Oriented* karena konsep ini menganalogikan sistem seperti kehidupan nyata yang didominasi oleh obyek dan digambarkan atau dinotasikan dalam simbol-simbol yang cukup spesifik. (Haviluddin , Jurnal Informatika Muawarman, VOL :6 , No.1 , Februari : 2011, 1-2).

Berikut gambar dari diagram UML



Gambar II.4 Diagram UML
Sumber : (Jurnal Informatika Muawarman, VOL :6 , No.1 , Februari : 2011)

1. **Use Case**

   Use Case Diagram atau diagram use case merupakan pemodelan untuk menggambarkan kelakuan (*behavior*) sistem yang akan dibuat. Diagram *use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Dengan pengertian yang cepat, diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Muawarman, VOL :6 , No.1 , Februari : 2011).

2. ***Sequence* Diagram**

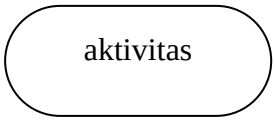
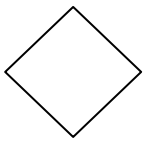
   Diagram Sekuenc*e* menggambarkan kelakuan/perilaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuence maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimilki kelas yang diinstansi menjadi objek itu. (Haviluddin , Jurnal Informatika Muawarman, VOL :6 , No.1 , Februari : 2011).

3. ***Activity* Diagram**

   Diagram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi

aktivitas yang dapat dilakukan oleh sistem. Diagram aktivitas mendukung perilaku pararel. (Edhy Sutanta, 2011)

Tabel II.1. Simbol – Simbol *Activity* Diagram

Simbol	Deskripsi
Status awal 	Status wal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah setatus awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / *join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memilikinsebuah status akhir
Swimalane	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Sumber :(M.Shalahuddin : 2013)

4. *Class* Diagram

Diagram kelas atau *class* diagram menggambarkan struktur sistem dari segi pendefinisian kelas – kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

- Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.
- Atribut mendeskripsikan properti dengan sebaris teks didalam kontak kelas tersebut.
- Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas

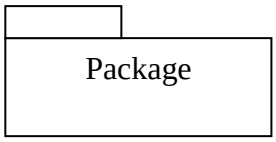
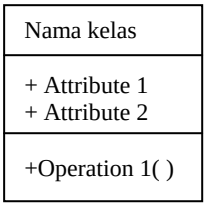
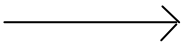
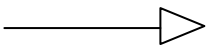
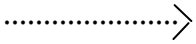
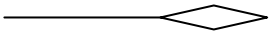
Diagram kelas mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan setatis yang terdapat di antara mereka. Diagram kelas juga menunjukkan properti dan operasi sebuah kelas dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut.

Diagram kelas menggambarkan struktur dan deskripsi class, package dan objeck beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi dan lain-lain.

Kelas memiliki tiga area pokok :

1. Nama
2. Atribut
3. Operasi

Tabel II.2. Simbol – Simbol *Class* Diagram

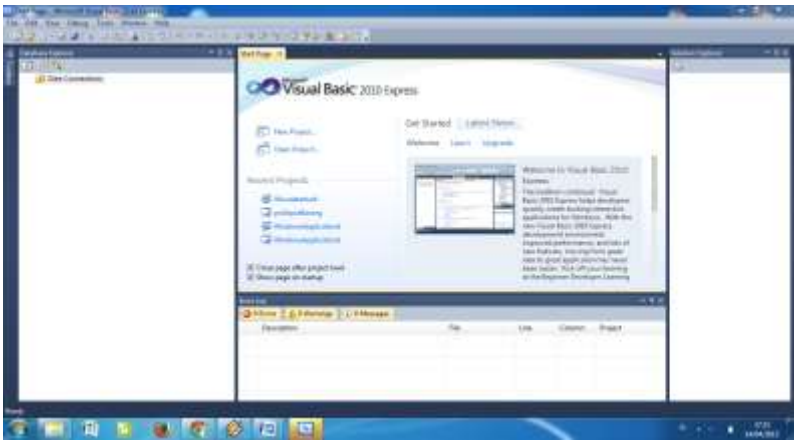
Simbol	Deskripsi
Package 	Package merupakan sebuah bungkusan dari satu atau lebih kelas
Operasi 	Kelas pada struktur sistem
Antarmuka / interface  **Interface**	Sma dengan konsep interface dalam pemograman berorientasi objeck
Asosiasi 1 1.. *	Relasi antara kelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity
Asosiasi berarah / directed asosiasi 	Relasi antara kelas dengan makna kelas satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity
Generalisasi 	Relasi antara kelas dengan makna generalisasi-spesialisasi (umum Khusus)
Kebergantungan / defedency 	Relasi antara kelas dengan makna keberuntungan antara kelas
Agregasi 	Relasi antara kelas dengan maknasemua-bagian (whole-part)

Sumber : (Yuni Sugiarti, M.Kom : 2013)

II.10. Bahasa Pemograman *Microsoft Visual Basic* 2010

Visual Basic 2010 merupakan salah satu bagian dari pemrograman terbaru yang dikeluarkan oleh *Microsoft Visual Studio* 2010. Sebagai pengembangan terintegrasi atau *IDE* andalan yang dikeluarkan oleh *Microsoft, Visual Studio* 2010 menambahkan perbaikan –perbaikan fitur dan fitur baru yang lebih lengkap dibandingkan versi *Visual Studio* pendahulunya, Yaitu *Microsoft Visual Studio* 200. Visual studio berisi beberapa jenis *IDE* pemrograman seperti *Visual Basic, Visual C++ , Visual Web Developer, Visual C# dan Visual F#*. (Wahana Komputer, 2010,2)

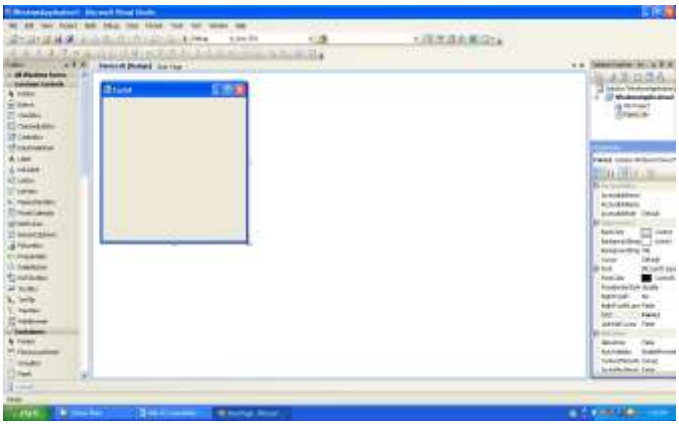
Untuk melihat tampilan *visual studio* 2010 dapat dilihat pada gambar II.11. sebagai berikut :



Gambar II.5. Tampilan Utama Visual Studio 2010
Sumber : (Wahana Komputer;2010:12)

Di dalam *Visual Studio.NET* menyediakan tampilan *Interface* yang sangat mudah untuk para pengguna merancang dan memodifikasi bentuk atau *Interface* dari program yang akan dibuat, dimana pada tampilan ini difasilitasi

dengan *Tool* dan fasilitas pendukung lainnya untuk mempermudah pengerjaannya.



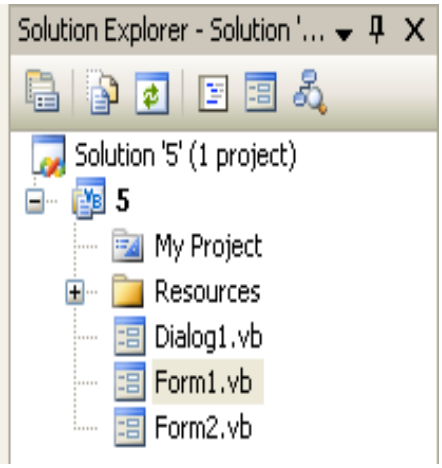
Gambar II.6. Tampilan *Area* Kerja *Visual Studio.NET*

Sumber: (Wahana Komputer; 2010: 10)

Dalam buku Belajar Pemrograman *Visual Basic* terbitan Wahana Komputer dmenjelaskan tampilan Microsoft *Visual Studio.NET*. Ada banyak hal penting yang harus diketahui oleh seorang programmer, diantaranya adalah:

a. *Solution Explorer*

Merupakan fitur yang terletak di sebelah kanan atas di bawah menu aplikasi yang digunakan untuk menampilkan daftar desain *form* dengan *struktur tree* dari project yang sedang dibuka. Dengan adanya fitur ini, memudahkan untuk berpindah antardesain *form* yang sudah anda buat secara cepat dan mudah.



Gambar II.7. *Solution Explorer*

Sumber: (Wahana Komputer; 2010:14)

b. *Properties*

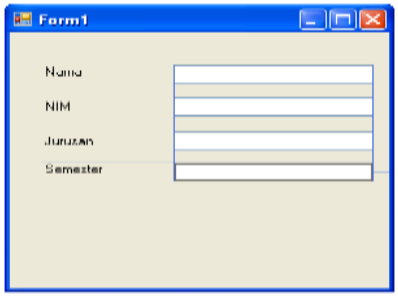
Merupakan fitur yang digunakan untuk melakukan pengaturan properti dari objek-objek yang digunakan dalam desain *form* yang akan dibuat, seperti pengaturan *text, visible, font, name* dan lain sebagainya.

Gambar II.8. Jendela *Properties*

Sumber: (Wahana Komputer; 2010: 15)

c. *Form* Designer

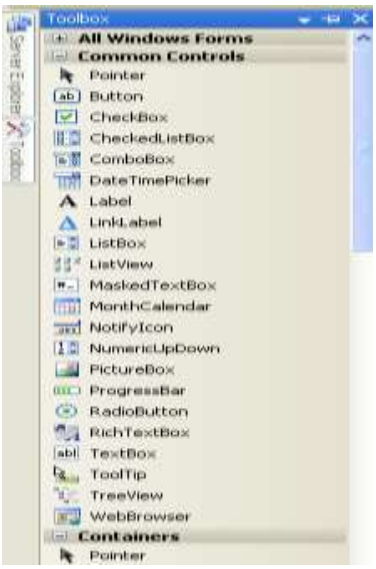
Merupakan fitur yang digunakan untuk membuat desain antarmuka atau *interface* dari aplikasi yang akan dikembangkan. Dengan mengadopsi fitur *click and drop*, proses pemanbahan komponen pada *form* menjadi semakin dinamis dan mudah. Selain itu tersedia pula fitur *guidelines* yang memudahkan anda dalam menata komponen yang terdapat pada desain *form* yang sedang anda kerjakan.



Gambar II.9. *Form Designer*
Sumber: (Wahana Komputer; 2010: 15)

d. *Component Toolbox*

Merupakan fitur *Visual Basic.NET* yang digunakan untuk menampilkan daftar dari komponen baik visual maupun non visual yang bias ditambahkan ke dalam desain *form* yang akan dibuat. Dalam komponen *toolbox*, terdapat berbagai macam kategori sesuai dengan kelompok komponen yang diakses sehingga akan memudahkan dalam mencari komponen yang akan ditambahkan kedalam desain *form*.

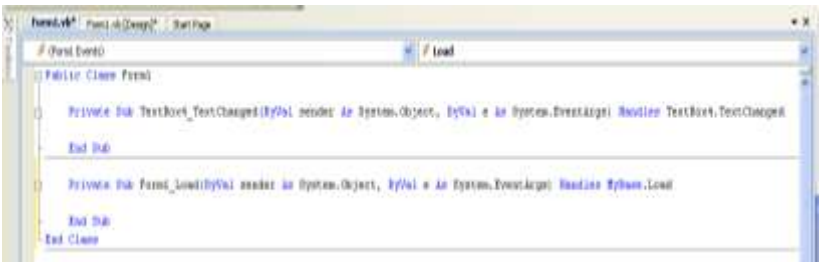


Gambar II.10. *Component Toolbox*

Sumber: (Wahana Komputer; 2010: 15)

e. *Code Editor*

Merupakan fitur yang digunakan untuk menambahkan kode program dari aplikasi atau *project* yang akan dikerjakan.



Gambar II.11. *Code Editor*

Sumber: (Wahana Komputer; 2010: 16)

II.11. *Sql Server* 2008

Sql Server 2008 merupakan sebuah terobosan terbaru dari *Microsoft* dalam bidang *database. Sql Server* adalah sebuah *DBMS* (*Database Management System*) yang dibuat oleh *Microsoft* untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahuluannya seperti

IBM dan *Oracle*. *SQL Server* 2008 dibuat pada saat kemajuan dalam bidang *hardware* sedemikian pesat. Oleh karena itu sudah dapat dipastikan bahwa *SQL Server* 2008 membawa beberapa terobosan dalam bidang pengolahan dan penyimpanan data. (Wahana Komputer : 2010)