BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem adalah kumpulan elemen yang saling berkaitan yang bertanggung jawab memproses masukan (input) sehingga menghasilkan keluaran (output) (kusrini;2007:11).

II.1.1. Karakteristik Sistem

Dapat diindikasikan empat karakteristik utama dari Sistem Pendukung Keputusan yaitu :

1. SPK menggabungkan data dan model menjadi satu bagian.
2. SPK dirancang untuk membantu para manajer (pengambil keputusan) dalam proses pengambilan keputusan dari masalah yang bersifat tidak terstruktur.
3. SPK lebih cenderung dipandang sebagai penunjang penilaian manajer dan sama sekali bukan untuk menggantikan.
4. Teknik SPK dikembangkan untuk meningkatkan efektivitas dari pengambil keputusan. (Rudi Hartoyo; 2013; 60).

II.2. Sistem Pendukung Keputusan

Sistem pendukung keputusan merupakan sistem informasi interaktif yang menyediakan informasi, pemodelan, dan pemanipulasian data. Sistem ini

digunakan untuk mengambil keputusan dalam situasi yang semiterstruktur dan situasi yang tidak terstruktur, dimana tak seorang pun tahu secara pribadi bagaimana keputusan seharusnya dibuat (Kusrini, 2007:15).

Keputusan yang diambil untuk menyelesaikan suatu masalah dilihat dari keterstrukturannya yang bisa dibagi menjadi:

1. Keputusan terstruktur *(structured decision)*

   Keputusan terstruktur adalah keputusan yang dilakukan secara berulang-ulang dan bersifat rutin. Prosedur pengambilan keputusan sangatlah jelas. Keputusan tersebut terutama dilakukan pada manajemen tingakat bawah. Misalnya, keputusan pemesanan barang dan keputusan penagihan piutang.

2. Keputusan semiterstruktur *(semistructured decision)*

   Keputusan semiterstruktur adalah keputusan yang dimiliki dua sifat. Sebagai keputusan bisa ditangani oleh komputer dan yang lain tetap harus dilakukan oleh pengambil keputusan. Prosedur dalam pengambilan keputusan tersebut secara garis besar sudah ada, tetapi ada beberapa hal yang masih memerlukan kebijakan dari pengambil keputusan. Biasanya, keputusan semacam ini diambil oleh manajer level menengah dalam suatu organisasi. Contoh keputusan jenis ini adalah pengevaluasian kredit, penjadwalan produksi, dan pengendalian sediaan.

3. Keputusan tak terstruktur *(unstructured decision)*

   Keputusan tak terstruktur adalah keputusan yang penanganannya rumit karena tidak terjadi berulang-ulang atau tidak selalu terjadi. Keputusan tersebut menuntut pengalaman dan berbagai sumber yang bersifat eksternal. Keputusan

tersebut umumnya terjadi pada manajemen tingkat atas. Contohnya adlah keputusan untuk pengembangan teknologi baru, keputusan unutk bergabung dengan perusahaan lain, dan perekrutan eksekutif. (Kusrini, 2007:19).

Tujuan dari DSS adalah:

a. Membantu manajer dalam pengambilan keputusan atas masalah semi terstruktur.
b. Memberikan dukungan atas pertimbangan manajer dan bukannya dimaksudkan untuk menggantikan fungsi manajer.
c. Meningkatkan efektifitas keputusan yang diambil manajer lebih dari pada perbaikan efesiensinya.
d. Kecepatan komputasi. Komputer memungkinkan para pengambilan keputusan untuk melakukan banyak komputasi secara cepat dengan biaya yang rendah.
e. Peningkatan produktivitas.
f. Dukungan kualitas.
g. Berdaya saing.
h. Mengatasi keterbatasan kognitif dalam pemrosesan dan penyimpanan (Kusrini, 2007:16).

II.2.1. Klasifikasi Sistem

Sistem dapat diklasifikasikan dari beberapa sudut pandang, diantaranya adalah sebagai berikut :

1. Klasifikasi Output Menurut Alter

   Dibuat berdasarkan "tingkat implikasi tindakan dari output sistem" atau tingkat di mana output sistem dapat langsung mendukung keputusan.

2. Klasifikasi menurut Holsapple dan Whinston

   Mengklasifikasikan DSS menjadi enam kerangka kerja: DSS berorientasi-teks, DSS berorientasi-database, DSS berorientasi-spreadsheet, DSS berorientasi-solver, DSS berorientasi-aturan (rule), dan DSS gabungan (compound DSS). (Turban;2005:165)

II.2.2. Tahap-Tahap Pengambilan Keputusan

Empat fase/tahapan dari pembuatankeputusan:

1. Fase Intelegensi

   Intelegensi dalam pengambilan keputusan meliputi scanning (pemindaian) lingkungan, entah secara intermiten ataupun terus-menerus. Inteligensi mencakup berbagai aktivitas yang menekankan identifikasi situasi atau peluang-peluang masalah.

2. Fase Desain

   Meliputi penemuan atau mengembangkan dan menganalisis tindakan yang mungkin untuk dilakukan.Hal ini meliputi pemahaman terhadap masalah dan

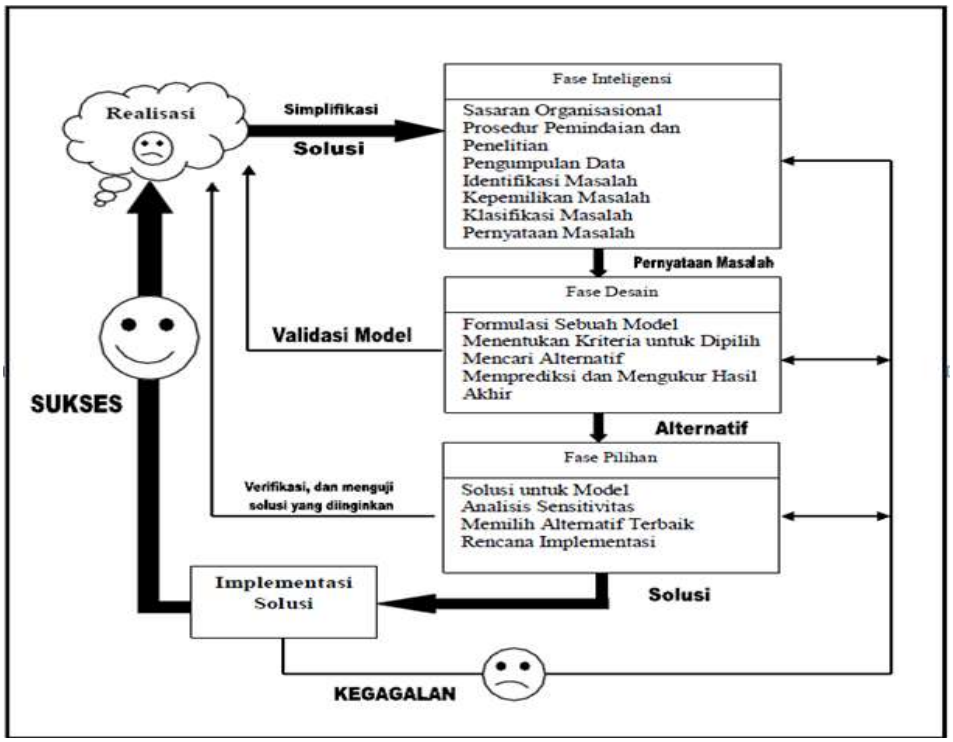
menguji solusi yang layak. Dan pada fase ini dikembangkan sebuah model masalah pengambilan keputusan untuk dikonstruksi, dites dan divalidasi.

3. Fase Pilihan

   Fase pilihan adalah fase dimana dibuat suatu keputusan yang nyata dan diambil suatu komitmen untuk mengikuti suatu tindakan tertentu. Fase pilihan meliputi pencarian, evaluasi,danrekomendasi terhadap suatu solusi yang tepat untuk model. Sebuah solusi untuk model adalah sekumpulan nilai spesifik untuk variabel-variabel keputusan suatu alternatif yang telah dipilih.

4. Implementasi

   Implementasi berarti membuat suatu solusi yang direkomendasikan bisa berkerja untuk mengatasi masalah.(Turban;2005:64)



Gambar II.1 : Proses Pengambilan Keputusan sumber : (Turban; 2005;65)

II.2.3. Komponen Sistem Pendukung Keputusan

Sistem pendukung keputusan terdiri atas tiga komponen utama yaitu :

1. Subsistem pengelolaan data (*database*).

   Subsistem manajemn data memasukkan satu database yang berisi data yang relevan untuk situasi dan dikelola oleh perangkat lunak yang disebut sistem manajemen database (DBMS).

2. Subsistem pengelolaan model (*modelbase*).

   Merupakan paket perangkat lunak yang memasukkan model keuangan, statistik, ilmu manajemen atau model kuantitatif lainnya yang memberikan kapabilitas analitik dan manajemen perangkat lunak yang tepat. Bahasa-bahasa pemodelan untuk membangun model-model kustom juga dimasukkan. Perangkat lunak ini sistem manajemen basis model (MBMS)

3. Subsistem pengelolaan dialog (*userinterface*).

   Pengguna berkomunikasi dengan dan memerintahkan SPK melalui subsistem ini. (Turban;2005:137)

II.3. Simple Additive Weighting (SAW)

Metode SAW sering juga dikenal istilah metode penjumlahan terbobot. Konsep dasar metode SAW adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif dari semua atribut . Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternatif yang ada (Rudi Hartoyo;2013;60).

Diberikan persamaan sebagai berikut :

$$r_{ij} = \begin{cases} \dfrac{{}_iX_{ij}}{\text{Max } X_{ij}} & \text{Jika j atribut Keuntungan (benefit)} \\ \dfrac{\text{Min } X_{ij}}{X_{ij}} & \text{Jika j atribut biaya (cost)} \end{cases}$$

Keterangan :

r_{ij} = nilai rating kinerja ternormalisasi

X_{ij} = nilai atribut yang dimiliki dari setiap kriteria

Max X_{ij} = nilai terbesar dari setiap kriteria

Min X_{ij} = nilai terkecil dari setiap kriteria

benefit = jika nilai terbesar adalah terbaik

cost = jika nilai terkecil adalah terbaik

Nilai preferensi untuk setiap alternatif (V_i)diberikan sebagai:

$$Vi = \sum \boldsymbol{wj\ rij}$$

V_i = nilai prefensi

W_j = bobot rangking

r_{ij} = rating kinerja ternormalisasi

Nilai V_i yang lebih besar mengindikasikan bahwa alternatif A_i lebih terpilih. (Rudi Hartoyo;2013; 60).

II.3.1. Langkah Penyelesaian metode SAW

Adapun langkah-langkah dari penyelesaian metode SAW adalah:

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.
4. Hasil akhir diperoleh dari proses perankingan yaitu penjumlahan dari perkalian matriks ternormalisasi R dengan vector bobot sehingga diperoleh nilai terbesar yang dipilih sebagai alternatif terbaik (A) sebagai solusi (Rudi Hartoyo;2013;61)

II.4. Visual Basic. Net 2010

Visual Basic 2010 merupakan bahasa pemrograman yang paling populer. Aneka ragam program dapat dibuat menggunakan visual basic 2010. Dengan memelajari cara pemrograman di visual basic 2010 akan membuat anda memiliki paradigma yang luas sebagai programmer yang dituntut serba bisa. Di samping itu, anda juga dituntut untuk selalu meng-update kemampuan anda dalam bidang IT.

Bahasa pemrogrman ini merupakan bahasa pemrograman yang sudah cukup lama, merupakan hasil pengembangan dari visual basic yang telah memiliki komunitas pengguna yang cukup besar.

II.5. SQL Server 2008

SQL server 2008 adalah sebuah RDBMS (*Relational Database Management System*) yang di develop oleh Microsoft, yangdigunakan untuk menyimpan dan mengolah data. Pada SQL 2008, kita bisa melalukan pangambilan dan mendidikasi data yang ada dengan cepat dan efisien. Pada SQL Server 2008, kita bisa membuat object-object yang sering digunakan pada aplikasi bisnis, seperti membuat database, table, function, stored procedure, trigger dan view. Selain object, kita juga menjalankan perintah SQL (*Structured Query Language*) untuk mengambil data.

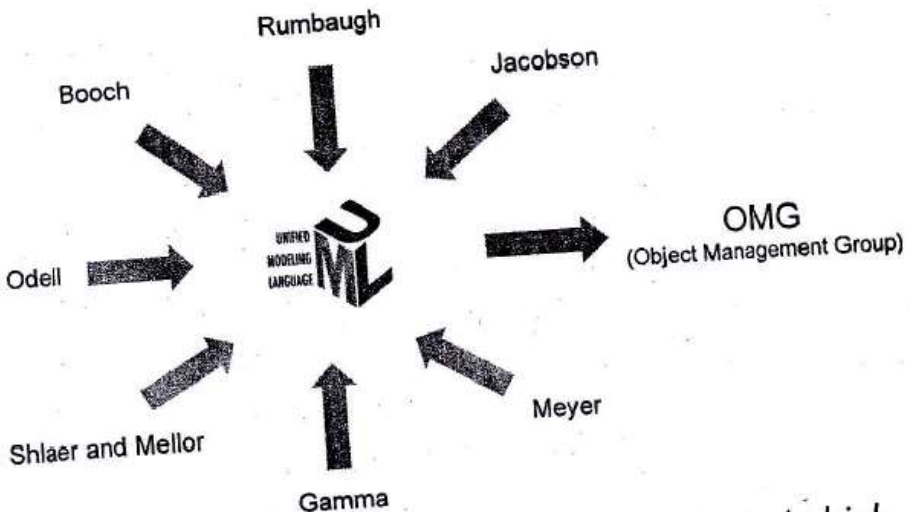
II.6. UML (*Unified Modelling Language*)

Unified Modelling Language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang sebuah sistem.

Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class*dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-

bahasa berorientasi objek seperti C++, java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C.

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantic. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *sytax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (Object-Oriented Design), Jim Rumbaugh OMT (Object Modeling Technique), dan Ivar Jacobson OOSE (Object-Oriented Software Engineering).



Gambar II.2. Metodelogi Pemodelan Berorientasi Objek

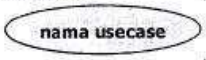
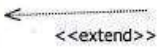
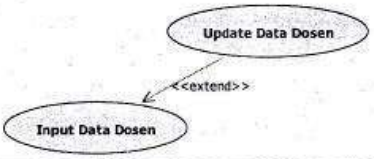
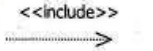
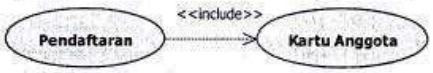
Sumber : (Yuni Sugiarti; 2013: 35)

II.7. Use Case Diagram

Dalam membuat sebuah sistem, langkah awal yang perlu dilakukan adalah menentukan kebutuhan.Terdapat dua kebutuhan yaitu kebutuhan fungsional dan nonfungsional. Kebutuhan fungsional adalah kebutuhan pengguna dan stakeholder sehari-hari yang akan dimiliki oleh sistem, dimana kebutuhan ini akan digunakan oleh pengguna atau stakeholder. Sedangkan kebuthan nonfungsional adalah kebutuhan yang memperhatikan hal-hal berikut yaitu performansi, kemudahan dalam menggunakan sistem, kehandalan sistem, keamanan sistem, keuangan, legalitas, operasional.

Kebutuhan fungsional akan digambarkan melalui sebuah gambar yang dinamakan diagram 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 actor dengan sistem yang dibuat. Dengan pengertian yang cepat, diagram use case digunakan untuk mengetahui fungsi apa saja yang ada dalam sebuah sistem dan apa saja yang berhak menggunakan fungsi-fungsi tersebut. Terdapat beberapa simbol dalam menggambarkan use case, yaitu use case, actor, relasi.

Adapun simbol use case diagram dapat dilihat di tabel II.1. berikut:

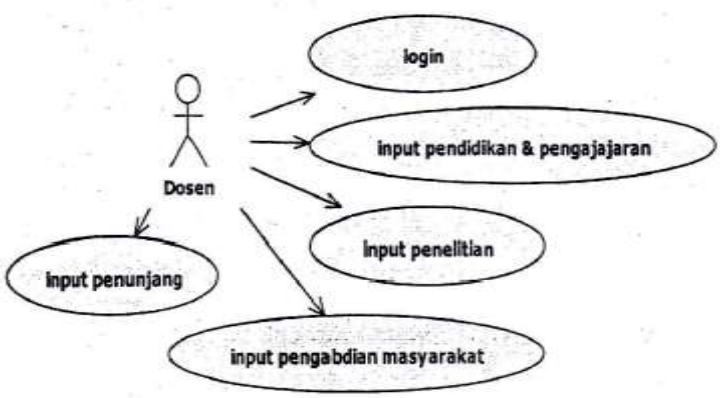
Simbol	Deskripsi
Use case 	fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awalfrase namause case
Aktor  nama actor	orang, proses, atau sistem lain yangberinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yangakan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang;biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Asosiasi / association 	komunikasi antara aktor danuse caseyangberpartisipasi pada use case atauuse casememiliki interaksi dengan aktor
Extend 	relasiuse case tambahan ke sebuah use case dimana use caseyang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya use casetambahan memiliki nama depan yang sama dengan use case yang ditambahkan, arah panah menunjuk pada use case yang dituju.contoh : 
Include 	relasiuse case tambahan ke sebuahuse casedimanause case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankanusecaseini. Ada dua sudut pandang yang cukup besar mengenai include diuse case, include berartiuse case yang ditambahkan akan selalu dipanggil saat use case tambahan dijalankan, contoh : 

Tabel II.1: Simbol Use Case Diagram

Sumber : (Yuni Sugiarti;2013:42)

Use Case Diagram adalah sebuah diagram yang menjelaskan apa yang harus dilakukan oleh sistem pada level konseptual sehingga kita akan memahami apakah keputusan yang diambil oleh sistem adalah benar atau tidak.

Adapun contoh use case adalah ditujukan oleh gambar II.3 sebagai berikut:



Gambar II.3. contoh Use Case Diagram

Sumber : (Yuni Sugiarti;2013:45)

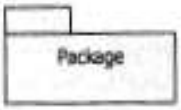
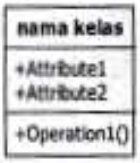
II.8. Class Diagram

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

- Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas.
- Atribut mendeskripsikan property dengan sebaris teks di dalam kotak tersebut.
- Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh satu kelas.

Diagram kelas mendeskripsikan jenis jenis objek dalam sistem dan berbagai hubungan statis yang terdapat diantara mereka.Diagram kelas juga menunjukan property dan operasi sebuah kelas dan batasan-batasanyang terdapat dalam hubungan-hubungan objek tersebut. Berikut adalah simbol-simbol yang ada pada diagram kelas dapat dilihat pada tabel II.2.

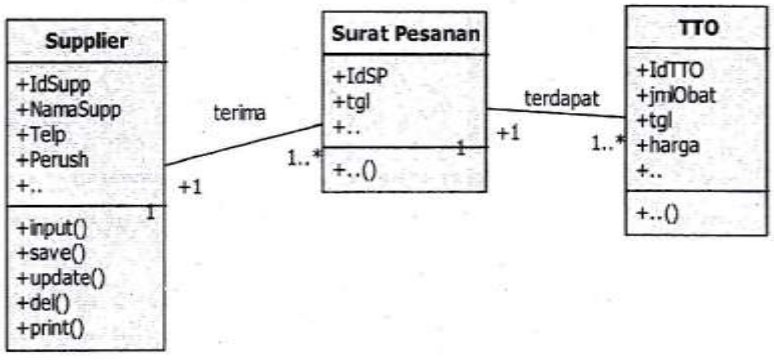
Adapun simbol Class Diagram dapat dilihat di tabel II.2. berikut:

Simbol	Deskripsi
Package 	Package merupakan sebuah bungkusan dari satu atau lebih kelas
Operasi 	Kelas pada struktur sistem
Antarmuka / interface 	sama dengan konsepinterface dalam pemrograman berorientasi objek
Asosiasi	relasi antar kelas dengan maknaumum, asosiasi biasanya juga disertai dengan multiplicity
Asosiasi berarah/directed asosiasi	relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya jugadisertai dengan multiplicity
Generalisasi 	relasi antar kelas dengan maknageneralisasi-spesialisasi (umumkhusus)
Kebergantungan/ defedency	relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi	relasi antar kelas dengan maknasemua-bagian (whole-part)

Tabel II.2: Simbol Class Diagram

Sumber : (Yuni Sugiarti;2013:59)

Adapun contoh Class Diagram dapat dilihat pada gambar II.4.berikut:



Gambar II.4. contoh Class Diagram

Sumber : (Yuni Sugiarti;2013:60)

II.8.1. Activity Diagram

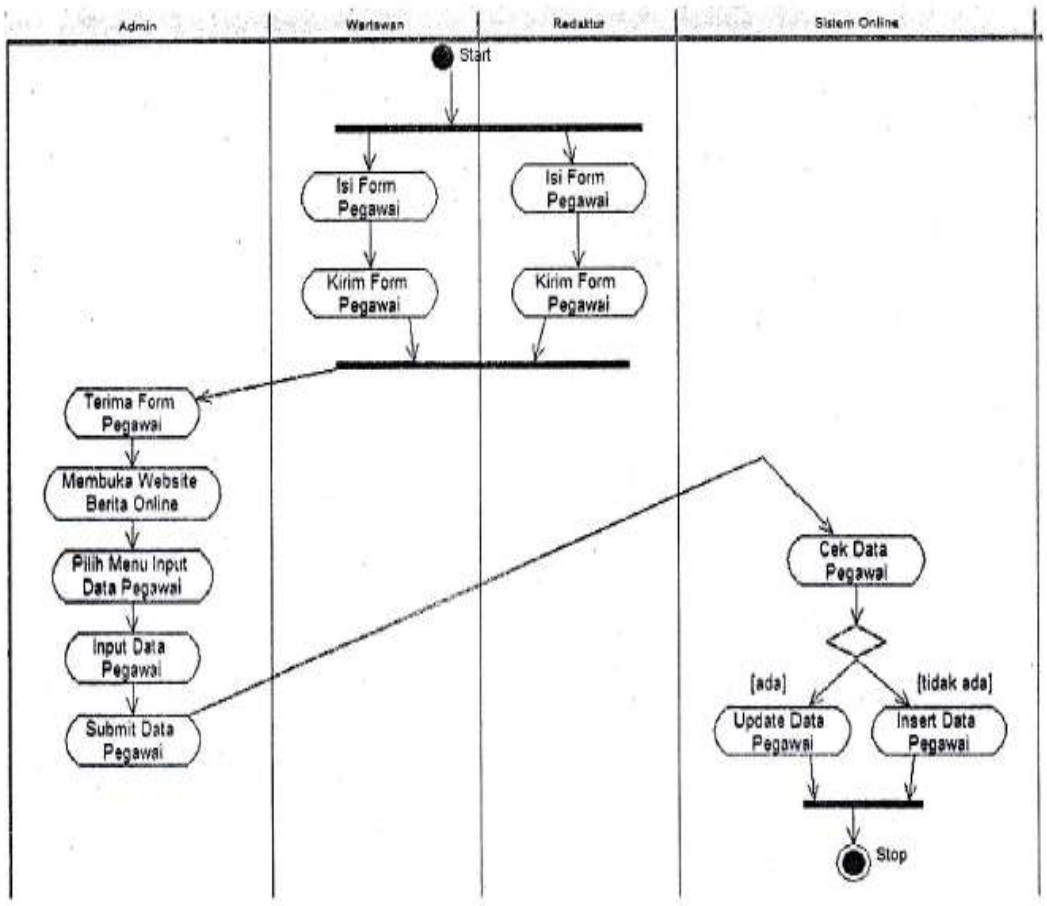
Diagram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu di perhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan actor, jadi aktivitas yang dapat dilakukan oleh sistem. Diagram aktivitas mendukung prilaku paralel.

Diagram aktivitas juga banyak digunakan untuk medefenisikan hal-hal berikut:

- Rancangan proses bisnis dimana setiap urutan aktivitas yang di gambarkan merupakan proses bisnis sistem yang di defenisikan.
- Urutan atau pengelompokan tampilan dari sistem/ user interface dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.

- Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefenisikan kasus ujiannya.

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 meggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Berikut adalah contoh activity diagram yang dapat dilihat pada gambar II.5.



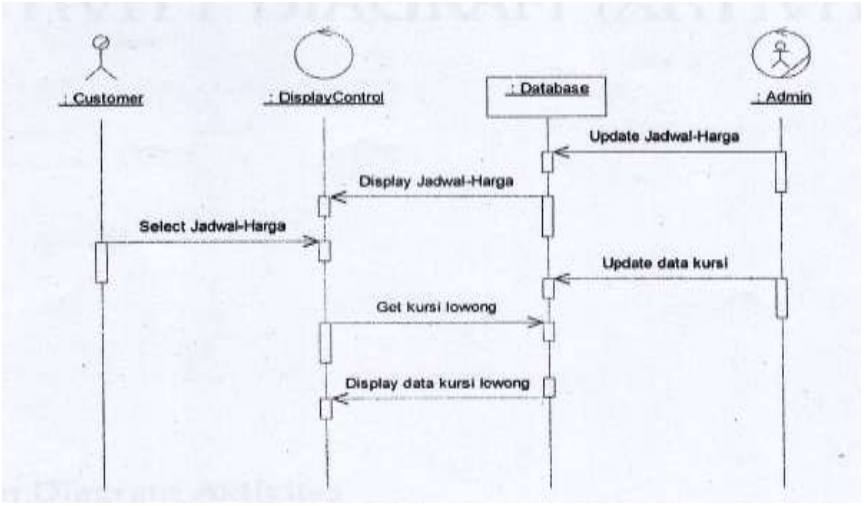
Gambar II.5. contoh Activity diagram

Sumber : (Yuni Sugiarti;2013:77)

II.8.2. Squence Diagram

Sequence diagram menggambarkan kelakuan/prilaku objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan di terima antar objek. Oleh karena itu untuk menggambar sequence diagram maka harus di ketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diintansiasi menjadi objek tertentu.

Banyaknya diagram *sequence* yang harus digambar adalah sebanyak pendefenisian *use case* yang memiliki proses sendiri atau yang penting semua use case yang telah di defenisikan interaksi jalannya pesan sudah dicakup pada diagram *sequence* sehingga semakin banyak *use case* yang di defenisikan maka diagram sequence yang harus di buat juga semakin banyak. Berikut contoh *Sequence* Diagram yang dapat dilihat pada gambar II.5 berikut:



Gambar II.6. Contoh *Sequence* Diagram

Sumber : (Yuni Sugiarti;2013:71)

II.8.3. Entity Relationship Diagram (ERD)

ERD adalah suatu model jaringan yang menggunakan susunan data yang disimpan dalam sistem secara abstrak. Jadi, jelaslah bahwa ERD ini berbeda dengan DFD yang merupakan suatu model jaringan fungsi yang akan dilaksanakan oleh sistem, sedangkan ERD merupakan model jaringan data yang menekankan pada struktur-struktur dan relationship data (Al-Bahra bin Ladja Muddin B ; 2004 : 123).

II.9. Normalisai

Normalisasi adalah proses pengelompokan data ke dalam bentuk tabel atau relasi atau file untuk menyatakan entitas dan hubungan mereka sehingga terwujud satu bentuk database yang mudah di modifikasi (Al-Bahra bin Ladja Muddin B ; 2004 : 174).

1. Bentuk Normal Ke Satu (1-NF)

   Pada tahap ini dilakukan penghilangan beberapa group elemen yang berulang agar menjadi satu harga tunggal yang berinterkasi di antara setiap baris pada suatu tabel, Syarat normal kesatu (1-NF) yaitu tidak ada set atribute yang berulang atau bernilai ganda.

2. Bentuk Normal Ke Dua (2-NF)

   Bentuk normal kedua didasari atasa konsep full functional dependency (ketergantungan fungsional sepenuhnya). Syarat normal kedua (2-NF) yaitu bentuk telah memenuhi kireteria bentuk normal kesatu, dan attribute bukan kunci (non-key) haruslah memiliki ketergantungan fungsional sepenuhnya

pada kunci utama/primary key. (Al-Bahra bin Ladja Muddin B ; 2004 : 187).

II.10. Kamus Data

Kamus data adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan kamus data sistem analis dapat mendefinisikan data yang mengalir pada sistem dengan lengkap.

Kamus data dibuat berdasarkan arus data yang ada pada data flow diagram. Arus data yang ada di DFD bersifat global dan hanya menunjukan nama arus datanya saja. Keterangan lebih lanjut tentang struktur dari suatu arus data di DFD dapat dilihat pada kamus data. Kamus data atau data dictionary harus dapat mencerminkan keterangan yang jelas tentang data yang dicatatnya. Untuk keperluan ini maka kamus data harus membuat hal-hal sebagai berikut (Tata Sutabri, S.Kom., MM;2004:170):

a. Arus Data

   Arus data menunjukkan dari mana data mengalir dan kemana data akan menuju. Keterangan arus data ini perlu dicatat di kamus data untuk memudahkan mencari arus data di dalam data flow diagram (DFD).

b. Nama Arus Data

   Karena kamus data dibuat berdasarkan arus data yang mengalir di data flow diagram, maka nama dari arus data juga harus dicatat dikamus data, sehingga mereka yang membaca DFD dan memerlukan penjelasan lebih

lanjut tentang suatu arus data tertentu di data flow diagram dapat langsung mencari dengan mudah di kamus data.

c. Tipe Data

Telah diketahui bahwa arus data dapat mengalir dari hasil suatu proses ke proses yang lain. Data yang mengalir ini biasanya dalam bentuk laporan serta dokumen hasil cetakan komputer. Dengan demikian bentuk dari data yang mengalir dapat berupa dokumen dasar atau formulir, dokumen hasil cetakan komputer, laporan tercetak, tampilan layar di monitor, variable, parameter dan field-field. Bentuk data seperti ini perlu dicatat di kamus data.

d. Struktur Data

Struktur data menunjukkan arus data yang dicatat pada kamus data yang terdiri dari item-item atau elemen-elemen data.

e. Alias

Alias atau nama lain dari data juga harus dituliskan. Alias perlu ditulis karena data yang sama mempunyai nama yang berbeda untuk orang atau departemen lainnya.

f. Volume

Volume yang perlu dicatat di dalam kamus data adalah volume rata-rata dan volume puncak dari arus data. Volume rata-rata menunjukkan banyaknya arus data yang mengalir dalam satu periode tertentu sementara volume puncak menunjukkan volume yang terbanyak.

g. Periode

Periode ini menunjukkan kapan terjadinya arus data. Periode perlu dicatat di kamus data kerena dapat digunakan untuk mengidentifikasikan kapan input data harus dimasukkan ke dalam sistem, kapan proses program harus dilakukan dan kapan laporan-laporan harus dihasilkan.

h. Penjelasan

Untuk memperjelas makna dari arus data yang dicatat di kamus data, maka bagian penjelasan dapat diisi dengan keterangan-keterangan tentang arus data tersebut (Tata Sutabri, S.Kom., MM;2004:171).