BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pendukung Keputusan

Sistem pendukung keputusan (SPK) atau *Decision Support Systems (DSS)* adalah sistem informasi interaktif yang menyediakan informasi, pemodelan, dan pemanipulasian data yang digunakan untuk membantu pengambilan keputusan pada situasi yang semiterstruktur dan situasi yang tidak terstruktur di mana tek seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat. Konsep *DSS* dikemukakan pertama kali oleh Scott-Morton pada tahun 1971. Beliau mendefenisikan cikal bakal *DSS* tersebut sebagai ”Sistem berbasis komputer interaktif, yang membantu pengambil keputusan menggunakan data dan model untuk memecahkan persoalan-persoalan tidak terstruktur”.

DSS dibuat sebagai reaksi atas ketidakpuasan terhadap TPS dan MIS. Sebagaimana diketahui, TPS lebih memfokuskan diri dari pada perekaman dan pengendalian transaksi yang merupakan kegiatan yang bersifat berulang dan terdefenisi dengan baik, sedangkan MIS lebih berorientasi pada penyediaan laporan bagi manajemen yang sifatnya tidak fleksibel. *DSS* lebih ditujukan untuk mendukung manajemen dalam melakukan pekerjaan yang bersifat analitis, dalam situasi yang kurang terstruktur dan dengan kriteria yang kurang jelas. *DSS* tidak dimaksudkan untuk mengotomasikan pengambilan keputusan, tetapi memberikan perangkat interaktif yang memungkinkan pengambil keputusan dapat melakukan

berbagai analisis dengan menggunakan model-model yang tersedia (Abdul Kadir; 2014 : 108).

Sistem pendukung keputusan sebagai sistem berbasis komputer yang terdiri dari tiga komponen yang saling berinteraksi, sistem bahasa (mekanisme untuk memberikan komunikasi antara pengguna dan komponen sistem pendukung keputusan lain), sistem pengetahuan (respositori pengetahuan domain masalah yang ada pada sistem pendukung keputusan atau sebagai data atau sebagai prosedur), dan sistem pemrosesan masalah (hubungan antara dua komponen lainnya, terdiri dari satu atau lebih kapabilitas manipulasi masalah umum yang diperlukan untuk pengambilan keputusan) (Dicky Nofriansyah ; 2014 : 1).

Ada tiga fase dalam proses pengambilan keputusan diantaranya sebagai berikut :

1. *Intellegence*

   Tahap ini merupakan proses penelusuran dan pendektesian dari ruang lingkup problematika secara proses pengenalan masalah. Data masukan diperoleh, diproses, dan diuji dalam rangka mengidentifikasi masalah.

2. *Design*

   Tahap ini merupakan proses menemukan,mengembangkan dan menganalisa alternatiftindakan yang bisa dilakukan. Tahap ini meliputi menguji kelayakan solusi.

3. *Choice*

   Pada tahap ini dilakukan proses pemilihan diantara berbaga alternatif tindakan yang mungkin dijalankan. Hasil pemilihan tersebut kemudian diimplementasikan dalam proses pengambilan keputusan (Dicky Nofriansyah ; 2014 : 2-3).

II.1.1. Karakteristik Sistem Pendukung Keputusan

Berikut ini adalah uraian beberapa karakteristik SPK, yaitu:

1. Kapabilitas interaktif

   Yaitu SPK memberi pengambil keputusan akses cepat ke data dan informasi yang dibutuhkan.

2. Fleksibilitas

   Yaitu SPK dapat menunjang para manajer pembuat keputusan di berbagai bidang fungsional (keuangan, pemasaran, operasi produksi, dan lain-lain).

3. Kemampuan menginteraksi model

   Yaitu SPK memungkinkan para pembuat keputusan berinteraksi dengan model-model, termasuk memanipulasi model- model tersebut sesuai dengan kebutuhan.

4. Fleksibilitas output

   Yaitu SPK mendukung para pembuat keputusan dengan menyediakan berbagai macam output, termasuk kemampuan grafik menyeluruh atas pertanyaan- pertanyaan pengandaian (Suhermin ; 2012 : 4).

II.1.2. Komponen Sistem Pendukung Keputusan

Secara garis besar sistem pendukung keputusan dibangun oleh tiga komponen utama yaitu (Dicky Nofriansyah; 2014; 3-4) :

1. Subsistem Data (*Database*)

   Subsistem data merupakan komponen sistem pendukung keputusan yang berguna sebagai penyedia data bagi sistem. Data tersebut disimpan untuk diorganisasikan dalam sebuah basis data yang diorganisasikan oleh suatu sistem yang disebut dengan sistem manajemen basis data (*Database Management System*).

2. Subsistem Model (*Model Base)*

   Model adalah suatu tiruan dari alam nyata. Kendala yang sering dihadapi dalam merancang model adalah bahwa model yang dirancang tidak mampu mencerminkan seluruh variabel alam nyata, sehingga keputusan yang diambil tidak sesuai dengan kebutuhan. Oleh karena itu, dalam menyimpan berbagai model harus diperhatikan dan harus dijaga fleksibilitasnya. Hal lain yang harus diperhatikan adalah pada setiap model yang disimpan hendaknya ditambahkan rincian keterangan dan penjelasan yang komprehensif mengenai model yang dibuat.

3. Subsistem Dialog (*User System Interface*)

   Subsistem dialog adalah fasilitas yang mampu mengintegrasikan sistem yang terpasang dengan pengguna secara interaktif, yang dikenal dengan subsistem dialog. Melalui subsistem dialog sistem

diimplementasikan sehingga pengguna dapat berkomunikasi dengan sistem yang dibuat.

II.2. Metode *Simple Additive Weighting (*SAW)

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.

$$r_{ij} = \begin{cases} \dfrac{x_{ij}}{\underset{i}{Max}\, x_{ij}} & \textit{jika } \text{j atribut keuntungan (benefit)} \\ \dfrac{\underset{i}{Min}\, x_{ij}}{x_{ij}} & \text{jika j atribut biaya (cost)} \end{cases} \quad \text{..... (1)}$$

Keterangan :

a. Rij = nilai rating kinerja normalisasi
b. Xij = nilai atribut yang dimiliki dari setiap kriteria
c. Max xij = nilai terbesar dari setiap kriteria
d. Min xij = nilai terkecil dari setiapkriteria
e. Benefit = nilai terbesar adalah terbaik
f. Cost = nilai terkecil adalah terbaik

Dimana rij adalah rating kinerja ternormalisasi dari alternatif Ai pada atribut Cj : i = 1, 2, …, m dan j = 1, 2, …, n. Nilai preferensi untuk setiap alternative (Vi) diberikan sebagai berikut :

$$V_i = \sum_{j=1}^{n} w_j \, r_{ij}$$ ..... (2)

Keterangan :

a. Vi = rangking untuk setiap alternatif

b. Wj = nilai bobot dari setiap kriteria

c. rij = nilai rating kinerja ternormalisasi (Youllia Indrawaty ; 2011 : 34).

II.2.1. Langkah *Simple Additive Weighting (*SAW)

Langkah-langkah dari 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 (Destriyana Darmastuti ; 2011 : 3).

II.2.2. Kelebihan Metode *Simple Additive Weighting*

Kelebihan dari model *Simple Additive Weighting* (SAW) dibandingkan dengan model pengambilan keputusan yang lain terletak pada kemampuannya 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 menentukan nilai bobot untuk setiap atribut (Destriyana Darmastuti ; 2011 : 3).

II.3. Pengertian Penilaian Kinerja

Organisasi juga membantu para karyawan dalam memperbaiki kinerja mereka dengan memberikan umpan baik tentang kinerja mereka di waktu sebelumnya. Penilaian kinerja (*performance appraisal*) adalah sebuah evaluasi terhadap kinerja dari pekerjaan karyawan, dengan cara membandingkan antara hasil *actual* dengan hasil yang diinginkan. Berdasarkan hasil evaluasi ini, manajer membuat keputusan objektif tentang kompensasi, promosi, kebutuhan pelatihan tambahan, transfer, dan pemutusan hubungan kerja. Memberikan penilaian terhadap kinerja para karyawan serta mengkomunikasikan persepsi tentang kekuatan dan kelemahan mereka adalah elemen penting dalam meningkatkan produktivitas maupun laba perusahaan. Penilaian kinerja tidak hanya terbatas dalam dunia bisnis. Agen pemerintah, organisasi nirlaba, dan institusi pendidikan juga melaksanakan penilaian kinerja.

Beberapa perusahaan melaksanakan penilaian terhadap rekan sekerjanya, di mana karyawan memberikan penilaian terhadap kinerja rekan kerjanya, sementara perusahaan lain memberikan kesempatan kepada karyawan untuk memberikan penilaian terhadap para pengawas (*supervisor*) dan manajer. Salah satu jenis metode penilaian kinerja adalah penilaian kinerja 360 derajat, yaitu proses yang mengumpulkan umpan baik dari sebuah panel penilaian yang terdiri atas 8 sampai 12 orang, termasuk rekan sekerja, pengawas, anggota tim kerja, bawahan, dan kadang-kadang pelanggan. Idenya adalah mendapatkan umpan balik sebanyak-banyaknya dari berbagai perspektif. Bagaimanapun, pendekatan penilaian kinerja ini cenderung untuk menambah pekerjaan, baik bagi karyawan maupun manajer, setiap orang harus memberikan penilaian terhadap 20 orang atau lebih dan juga harus menghadapi tumpukan dokumen pekerjaan. Juga, karena evaluasi yang dilakukan tidak mencatumkan identitas, staf yang terlibat, yang memiliki itikad buruk dapat menggunakannya untuk mengambil keuntungan dalam perselisihan pribadi (Boone & Kurtz ; 2011 : 432).

II.4. Pengertian Basis Data

Basis data (*database)* adalah suatu pengorganisasian sekumpulan data yang saling terkait sehingga memudahkan aktifitas untuk memperoleh informasi. Basis data dimaksudkan untuk mengatasi problem pada sistem yang memakai pendekatan berbasis berkas.

Untuk mengelola basis data diperlukan perangkat lunak yang disebut *Database Management System* (*DBMS*). *DBMS* adalah perangkat lunak sistem

yang memungkinkan para pemakai membuat, memelihara, mengontrol dan mengakses basis data dengan cara yang praktis dan efisien. *DBMS* dapat digunakan untuk mengakomodasikan berbagai macam pemakai yang memiliki kebutuhan akses yang berbeda-beda (Abdul Kadir ; 2014: 218).

II.5. Pengertian ERD

ERD adalah suatu diagram untuk menggambarkan desain konseptual dari model konseptual suatu basis data relasional. ERD juga merupakan gambaran yang merelasikan antara objek yang satu dengan objek yang lain dari objek di dunia nyata yang sering dikenal dengan hubungan antar entitas. Sebagai contoh jika membuat ERD dari sistem perpustakaan maka bahan sebagai objek ERD bisa berupa anggota, buku, peminjam, pengembalian dan sebagainya. ERD terdiri dari 4 komponen utama, yaitu (Robi Yanto ; 2016 : 32) :

Tabel II.1. Simbol ERD

Notasi	Keterangan
	**Entitas**, adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
	**Relasi**, menunjukkan adanya hubungan di antara sejumlah entitas yang berbeda.
	**Atribut**, berfungsi mendeskripsikan karakter entitas (atribut yg berfungsi sebagai key diberi garis bawah)
	Garis, sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

(Sumber : Robi Yanto ; 2016 : 32)

II.6. Pengertian Kamus Data

Kamus data umumnya berguna pada pengembangan model sistem dan dapat digunakan untuk menangani semua informasi dari semua tipe model sistem. Kamus data sederhanya adalah daftar alfabetis dari nama-nama termasuk pada berbagai model sistem. Seperti namnya, kamus harus mencakup deskripsi yang berhubungan dengan entitas bernama tersebut dan, jika nama itu merepresentasikan objek komposit, mungkin saja ada deskripsi mengenai komposisinya. Informasi lain seperti tanggal pembuatan, pembuatnya dan representsi entitas juga dapat dimasukkan, tergantung pada tipe model yang sedang dikembangkan. Keuntungan penggunaan kamus data adalah :

1. Kamus data merupakan mekanisme untuk manajemen nama. Banyak orang yang harus menciptakan nama untuk entitas dan relasi ketika mengembangkan model sistem yang besar. Nama-nama ini harus dipakai secara konsisten dan tidak boleh bentrok. Perangkat lunak kamus data dapat memeriksa keunikan nama dan memberitahu analils persyaratan sekiranya terjadi duplikasi nama.
2. Kamus data berfungsi sebagai tempat penyimpanan informasi organisasional yang dapat menghubungkan analisis, desain, implementasi dan evolusi. Sementara sistem dikembangkan, informasi diambil untuk memberitahu perkembangan. Informasi baru ditambahkan pada sistem. Semua informasi mengenai entitas berada pada satu tempat (Ian Sommerville ; 2012 : 151).

Tabel II.2 Entri Kamus Data

Nama	Keterangan	Tipe	Tanggal
Has-labels	Relasi 1 N antara entitas bertipe node atau link dan entitas bertipe label	Relasi	5.10.1998
Label	Berisi informasi terstruktur atau tidak terstruktur mengenai node atau link. Label dipresentasikan oleh ikon (yang bisa berupa kotak transparan) dan teks yang berhubungan	Entitas	8.10.1998
Link	Relasi 1 : 1 antara entitas desain yang dipresentasikan sebagai node. Link diketikkan dan dapat diberi nama	Relasi	8.10.1998
Name (Label)	Setiap label memiliki nama yang mengidentifikasikan tipe label. Nama harus unik dalam set tipe label yang dipakai pada desain	Atribut	8.10.1998
Name (Node)	Setiap node harus memiliki nama yang unik di dalam sebuah desain. Panjang nama bisa mencapai 64 karakter.	Atribut	15.10.1998

(Sumber : Ian Sommerville ; 2012 : 152)

II.7. Pengertian Normalisasi

Normalisasi adalah suatu proses untuk membuat data yang tidak normal menjadi data yang normal. Bentuk data yang tidak normal / data mentah biasa disebut juga *unnormalized form*. Masing – masing level normalisasi mempunyai aturan tersendiri.

1. *First Normal Form*

   Suatu tabel dikatakan dalam keadaan *first normal form* (1NF) jika :

   a. Tidak ada perulangan record data dalam tabel.

   b. Setiap sel memiliki satu nilai saja. Artinya tidak ada perulangan group dan array.

   c. Data yang diinputkan memiliki tipe data yang sama dengan tipe data kolom dalam tabel.

2. *Second Normal Form*

   Suatu tabel dikatakan dalam keadaan *Second Normal Form* (2NF) jika tabel tersebut sudah dalam keadaan *First Normal Form* (1NF) dan jika semua atribut yang bukan kunci tabel, baik *primary key* maupun *foreign key* tergantung pada semua kunci dalam tabel.

3. *Third Normal Form*

   Suatu tabel dikatakan dalam keadaan *third normal form* (3NF) jika tabel tersebut sudah dalam keadaan *second normal form* (2NF) dan jika tidak terdapat ketergantungan yang transitif. Artinya, data-data yang mungkin diisi berulang-ulang dapat dibuat sebuah tabel baru.

4. *Boyce-Codd Normal Form* (BCNF)

   Tabel dikatakan dalam keadaan *boyce-codd normal form* (BCNF) jika tabel tersebut dalam keadaan *third normal form* (3NF) dan setiap determinan adalah kunci kandidat.

5. *Fourth Normal Form* (4NF)

   Suatu tabel dikatakan dalam keadaan *fourth normal form* (4NF) jika tabel tersebut dalam keadaan *boyce-codd normal form* (BCNF) dan jika tidak terdapat ketergantungan nilai ganda.

6. *Fiveth Normal Form* (5NF)

   Tabel dikatakan dalam keadaan *Fiveth Normal Form* (5NF) jika tabel tersebut dalam keadaan *fourth normal form* (4NF) dan jika setiap ketergantungan dalam join ada pada tabel sudah konsekuen dengan kunci kandidat pada tabel tersebut (Ema Utami ; 2012 : 73-76).

II.8. Pengertian SQL

SQL *Server* 2008 adalah sebuah terobosan baru 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 pendahulunya 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 ; 2013 : 2).

II.9. Pengertian Visual Basic

VB.NET adalah salah satu bahasa pemrograman tingkat tinggi yang mendekati bahasa manusia. Kemunculan bahasa VB.NET ini sebagai jawaban untuk menyederhanakan bahasa pemrograman pada *platform* .NET yang diluncurkan tahun 2002 dan untuk menjembatani programmer *Visual Basic*. Bahasa VB.NET secara teknis mengadopsi sintak bahasa *Visual Basic*. Konsistensi API membuat bahasa VB.NET menjadi pilihan dalam membuat kode program diatas *platform* Windows. Fitur baru bahasa VB.NET dibandingkan *Visual Basic* bahwa bahasa VB.NET mendukung *object-oriented* dan juga *dynamics* programming. Ini menambah daftar kemudahan untuk belajar bahasa VB.NET (Agus Kurniawan ; 2013 : 10).

Toolbox adalah tempat di mana kontrol-kontrol dan komponen-komponen diletakkan. Kontrol dan komponen disimpan pada *toolbox* dengan berbagai kategori, yaitu :

1. *Common Controls*, berisi kontrol-kontrol umum yang sering digunakan seperti *button*, *label*, *textbox*, dsb.
2. *Containers,* berisi control penyimpan control lainnya seperti *panel, group box, tabcontro*l, dsb.
3. *Menu* & *Toolbars,* berisi control dan komponen menu, *context menu* dan *toolbar*.
4. *Data,* berisi control dan komponen pengolahan data.
5. *Components,* berisi komponen-komponen seperti *timer, imagelist*, dsb.
6. *Printing*, berisi komponen untuk pencetakan dokumen.
7. *Dialogs,* berisi komponen untuk berinteraksi dengan pengguna dalam hal membuka *file*, menyimpan *file*, membuka *folder* dll.
8. *WPF Interoperability*, berisi komponen untuk *Windows Presentation Foundation*.
9. *Reporting,* berisi kontrol untuk membuat laporan pada visual studio 2010
10. *Visual Basic PowerPacks,* berisi beberapa kontrol tambahan untuk menggambar (contok: oval, garis, persegi) dan fungsi tambahan lainnya.
11. *General*, jika ada kontrol atau komponen yang mau ditambakan dan belum memiliki kategori yang jelas, maka bisa dimasukkan ke kategori ini (Priyanto Hidayatullah ; 2012 : 27-28).

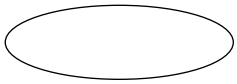
II.10. UML (*Unified Modeling Language*)

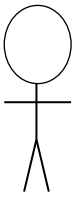
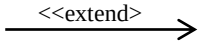
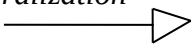
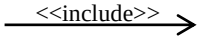
Pada perkembangan teknik pemograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objeek, yaitu *Unified Modelig Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung (Rosa A.S & M. Shalahuddin ; 2016 : 137).

1. *Use case* Diagram

   Use case atau diagram *use case* merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Berikut adalah simbol-simbol yang ada pada diagram *use case* :

Tabel II.3. Simbol *Use Case*

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 awal di awal frase nama *use case*.

Aktor / *actor* 	orang, proses, atau sistem yang lain berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan 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 dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
Ekstensi / *extend* 	relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahakn dapat berdiri sendiri walaupun tanpa *use case* tambahan itu; mirip dengan prinsip *inheritance* pada pemograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
Generalisasi / *generalization* 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
Menggunakan / *include* / *uses* 	relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

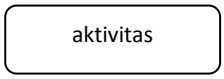
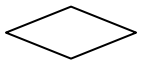
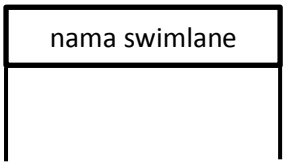
(Sumber : Rosa A.S & M. Shalahuddin ; 2016 : 156-157)

2. Diagram Aktivitas (*Activity Diagram*)

Diagram aktivitas atau *activity Diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram

aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, aktivitas yang dapat dilakukan bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Berikut adalah simbol-simbol yang ada pada diagram aktivitas:

Tabel II.4. Simbol *Activity Diagram*

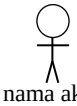
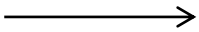
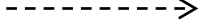
Simbol	Deskripsi
status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
percabangan / *decision* 	asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan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

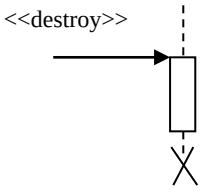
(Sumber : Rosa A.S & M. Shalahuddin ; 2016 : 162)

3. Diagram Urutan (*Sequence Diagram*)

Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Berikut adalah simbol-simbol yang ada pada diagram sekuen:

Tabel II.5. Simbol *Sequence Diagram*

Simbol	Deskripsi
Aktor  atau nama aktor	orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan 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.
garis hidup / *lifeline*	menyatakan kehidupan suatu objek
Objek nama objek : nama kelas	menyatakan objek yang berinteraksi pesan
Waktu aktif 	menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya.
Pesan tipe create << create >> 	menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe call 1 : nama_metode()	menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri, arah panah mengarah pada objek yang memiliki operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas objek yang berinteraksi
Pesan tipe send 1 : masukan	menyatakan bahwa suatu objek mengirim data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim
Pesan tipe return 1 : masukan 	menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian

Pesan tipe destroy <<destroy>> 	menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy

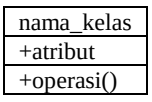
(Sumber : Rosa A.S & M. Shalahuddin ; 2016 : 165-167)

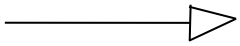
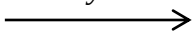
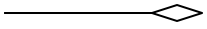
4. Diagram Kelas (*Class Diagram*)

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem.

Diagram kelas dibuat agar pembuat program atau programmer membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Berikut adalah simbol-simbol yang ada pada diagram kelas:

Tabel II.6. Simbol *Class Diagram*

Simbol	Deskripsi
Kelas 	kelas pada struktur sistem
antarmuka / *interface* 	sama dengan konsep *interface* dalam pemrograman berorientasi objek
asosiasi / *association* 	relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicty*
asosiasi berarar / *directed association* 	relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicty*
generalisasi	relasi antarkelas dengan makna generalisasi-

	spesialisasi (umum khusus)
Ketergantunagan / *dependency* 	relasi antarkelas dengan makna ketergantungan antarkelas
agregasi / *aggregation* 	relasi antarkelas dengan makna semua-bagian (*whole-part*)

(Sumber : Rosa A.S & M. Shalahuddin ; 2016 : 146-147)