BAB II
TINJAUAN PUSTAKA

II.1. Sistem

Secara leksikal, sitem berarti susunan yang teratur dari pandangan, teori, asas dan sebagainya. Dengan kata lain, sistem adalah suatu kesatuan usaha yang terdiri dari bagian-bagian yang berkaitan satu sama lain yang berusaha mencapai suatu tujuan dalam suatu lingkungan kompleks. Pengertian tersebur mencerminkan adanya beberapa bagian dan hubungan antara bagian, ini menunjukkan kompleksitas dari sistem yang meliputi kerja sama antara bagian yang interpenden satu sama lain. Selain itu dapat dilihat bahwa sistem berusaha mencapai tujuan. Pencapaian tujuan ini menyebabkan timbulnya dinamika, perubahan-perubahan yang terus menerus perlu dikembangkan dan dikembalikan. Definisi tersebut menunjukkan bakwa sistem sebagai gugus dan elemen-elemen yang saling beriteraksi secara teratur dalam rangka mencapai tujuan atau subtujuan (Prof. Dr. Ir. Marimin, M.Sc ; 2008 : 1).

II.2 Pakar

Pakar adalah seorang yang mempunyai pengetahuan, pengalaman, dan metode khusus, serta mampu menerapkannya untuk memecahkan masalah atau memberi nasihat. Seorang pakar harus manpu menjelaskan dan mempelajari hal-hal baru yang berkaitan dengan topik permasalahan, jika perlu harus mampu menyusun kembali pengetahuan-pengetahuan yang didapatkan, dan dapat memecahkan aturan—aturan serta menentukan relevansi kepakarannya.

II.3 Sistem Pakar

II.3.1. Pengertian Sitem Pakar

Sistem Pakar adalah sistem komputet yang ditujukan untuk meniru semua aspek (*emulates*) kemampuan pengambilan keputusan (*decision making*) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selayaknya seorang pakar untuk memecahkan masalah (Rika Rosnelly;2012 : 2).

Sistem Pakar adalah aplikasi berbsais komputer yang digunakan untuk menyelesaikan masalah sebagaimana yang dipikirkan oleh pakar. Pakar yang dimaksud disini adalah orang yang mempunyai keahlian dibidangnya yang dapat menyelesaikan masalah yang orang lain tidak dapat di selesaikan oleh orang awam. Sebagai contoh mekanik mobil adalah seorang pakar yang mampu mendiagnosa kerusakan yang dialami mobil serta dapat memperbaiki kerusakan tersebut.

II.3.2. Manfaat Sistem Pakar

Menurut T.Sutojo, S.Si.,M.Kom, dkk, Sistem pakar memiliki beberapa manfaat, seperti :

1. Meningkatkan produktifitas, karena sistem pakar dapat bekerja lebih cepat daripada manusia.
2. Membuat seorang yang awam bekerja seperti layaknya seorang pakar.
3. Meningkatkan kualitas, dengan memberikan nasehat yang konsisten dan mengurangi kesalahan.
4. Mampu menagkap pengetahuan dan kepekaan seseorang.

5. Meningkatkan kemampuan untuk menyelesaikan masalah karena sistem pakar mengambil sumber pengetahuan dari banyak pakar

II.3.3. Konsep Dasar Sistem Pakar

1. Inferensi

Inferensi adalah sebuah prosedur (program) yang mempunyai kemampuan dalam melakukan penalaran. Inferensi ditampilkan pada suatu komponen yang disebut mesin inferensi yang mencakup prosedur-prosedur mengenai pemecahan masalah. Semua pengetahuan yang dimiliki oleh seorang pakar disimpan pada basis pengetahuan oleh sistem pakar. Tugas mesin inferensi adalah mengambil kesimpulan berdasarkan basis pengetahuan yang dimilikinya.

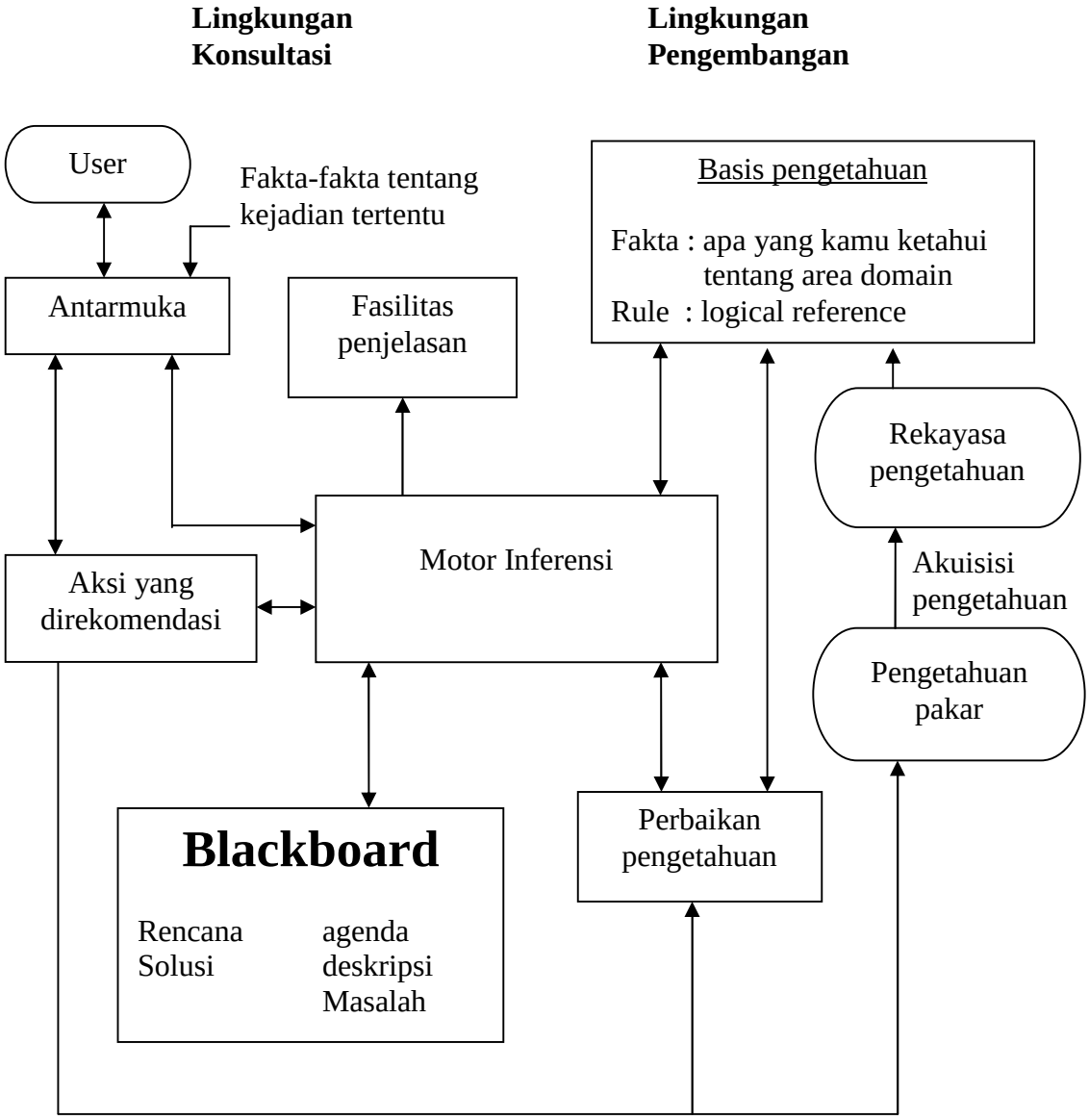
2. Aturan-aturan (*Rule*)

Kebanyakan software sistem pakar komersial adalah sistem yang berbasis rule (*rule-based system*), yaitu pengetahuan disimpan bentuk *rule*, sebagai prosedur-prosedur pemecahan masalah.

II.3.4. Struktur Sistem Pakar

Ada dua bagian penting dari sistem pakar, yaitu lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangan digunakan oleh pembuat sistem pakar untuk membangun komponen-komponennya dan memperkanalkan pengetahuan ke dalam *knowledge base* (basis pengetahuan). Lingkungan konsultasi digunakan oleh pengguna olah pengguna untuk berkonsultasi sehingga pengguna mendapatkan pengetahuan dan nasihat dari sistem pakar layaknya berkomunikasi

dengan seorang pakar. Gambar II.1 menunjukan komponen-komponen yang penting dalam sebuah sistem pakar.



Gambar II.1 : Komponen-Komponen Penting Dalam Sebuah Sistem Pakar

Sumber : (T.Sutojo, S.Si.,M.Kom, dkk ; 2011 : 167)

Keterangan :

1. Akuisisi Pengetahuan

   Subsistem ini digunakan untuk memasukkan pengetahuan dari seorang pakar dengan cara merekayasa pengetahuan agar bisa diproses oleh komputer dan menaruhnya ke dalam basis pengetahuan dengan format tertentu (dalam bentuk representasi pengetahuan). Sumber-sumber pengetahuan bisa diproleh dari pakar, buku, dokumen, multimedia, basis data, laporan riset khusus, dan informasi yang terdapat di web.

2. Basis Pengetahuan

   Basis pengetahuan mengandung pengetahuan yang di perlukan untuk memahami, memformulasikan dan menyelesaikan masalah. Basis pengetahuan terdiri dari dua elemen dasar, yaitu :

   a. Fakta, misalnya situasi, atau permasalahan yang ada.
   b. Rule (Aturan), untuk mengarahkan pengguna pengetahuan dalampemecahan masalah.

3. Mesin Inferensi (Inference Engine)

   Mesin inferensi adalah sebuah program yang berfungsi untukmemandu proses penalaran terhadap suatu kondisi berdasarkan pada basis pengetahuan yang ada,manipulasi dan mengarahkan kaidah, model, dan fakta yang disimpan dalam basis pengetahuan untuk mencapai solusi atau kesimpulan.dalam prosesnya, mesin inferensi menggunakan strategi pengendalian, yaitu strategi yang berfungsi sebagai panduan arah dalam melakukan proses penalaran. Ada

tiga teknik pengendalian yang digunalkan, yaitu *forward chaining, bacward chaining*, dan gabungan dari kedua teknik tersebut.

4. Daerah Kerja (*Blackboard*)

   Untuk merekan hasil sementara yang akan dijadikan sebagai keputusan dan untuk menjelaskan sebuah maslah yang sedang terjadi, sistem pakar ,membutuhkan *Blacboard*, yaitu area pada memori yang berfungsi sebagai basis data. Tiga tipe keputusan yang dapat direkam pada *Blackboard*, yaitu

a. Rencana : bagaimana menghadapi masalah
b. Agenda : aksi-aksi potensial yang sedang mengunggu untuk dieksekusi
c. Solusi : calon aksi yang akan dibangkitkan

5. Antarmuka (User *interface*)

   Digunakan sebagai media komunikasi antara pengguna dan sistem pakar. Komunikasi ini paling bagus bila di sajikan dalam bahasa alami (*natural language*) dan dilengkapi dengan grafik, menu, dan formulir elektronik. Pada bagian ini akan terjadi dialog antara sistem pakar dan pengguna.

6. Subsistem Penjelasan (*Explanation Subsystem / Justifier*)

   Berfungsi memberi penjelasan kepada pengguna, bagaimana suatu kesimpulan dapat diambil. Kemampuan seperti ini sangat penting bagi pengguna untuk mengetahui proses pemindahan keahlian pakar maupun dalam pemecahan masalah.

7. Sitem Perbaikan Pengetahuan (*Knowledge Refining System*)

   Kemampuan memperbaiki pengetahuan (*Knowledge Refining System*) dari seorang pakar diperlukam untuk menganalisis pengetahuan, belajar dari

kesalahan masa lalu, kemudian memperbaiki pengetahuan sehingga dapat dipakai dimasa yang mendatang. Kemampuan evaluasi diri seperti itu diperlukan oleh program agar dapat menganalisis alasan-alasan kesuksesan dan kegagalannya dalam mengambil kesimpulan.

8. Pengguna (*User*)

   Pada umumnya pengguna sistem pakar bukanlah seorang pakar (*non-expert*) yang membutuhkan solusi, saran, atau pelatihan (*training*) dari berbagai permasalahan yang ada.

II.4. Visual Basic 2010

Visual basic diturunkan dari bahas *BASIC*. Visual Basic terkenal sebagai bahasa pemrograman yang mudah untuk digunakan terutama untuk membuat aplikasi yang berjalan diatas *platform Windows.*

Pada tahun 90an, Visual basic menjadi bahasa pemrograman yang paling populer dan menjadi pilihan utama untuk mengembangkan program berbasis windows. Versi Visual Basic terakhir sebelum berjalan diatas .Net Framework adalah VB6 atau Visual Studio 1998.

Visual Basic .Net dirilis pada bulan Februari tahun 2002 bersamaan dengan platform .Net Framework 1.0. Kini sudah ada beberapa versi dari Visual Basic yang berjalan pada platform .net, yaitu VB 2002 (VB7), VB 2005 (VB8), VB 2008 (VB9), dan yang terakhir VB 2010 (VB10) yang dirilis bersamaan dengan Visual Studio 2010.

Selain Visual Basic 2010, Visual Studio juga mendukung beberapa bahasa lain, yaitu C#, C++, F# (bahasa baru untuk functional programming), IronPhyton, dan IronRuby (bahasa baru untuk dynamic programming) (Kurniawan ; 2011 :1).

II.5. Metode Demsper Shafer

Penerapan penghitungan kemungkinan penyakit yang diderita oleh burung kenari dengan metode Dempster-Shafer. Teori Dempster-Shafer merupakan teori matematika dari evidence. Teori tersebut dapat memberikan sebuah cara untuk menggabungkan evidence dari beberapa sumber dan mendatangkan atau memberikan tingkat kepercayaan (direpresentasikan melalui fungsi kepercayaan) dimana mengambil dari seluruh evidence yang tersedia. Secara umum Teori Dempster-Shafer ditulis dalam suatu interval :

- [Belief,Plausibility] Belief (Bel) adalah ukuran kekuatan evidence dalam mendukung suatu himpunan proposisi. Jika bernilai 0 maka mengindikasikan bahwa tidak ada evidence, dan jika m bernilai 1 menunjukkan adanya kepastian. Dimana nilai bel yaitu (0-0.9).
- Plausibility (Pl) dinotasikan sebagai:

$Pl(s) = 1 – Bel(\neg s)$

Plausibility juga bernilai 0 sampai 1. Jika kita yakin akan –s, maka dapat dikatakan bahwa $Bel(\neg s)=1$, dan $Pl(\neg s)=0$. Plausability akan mengurangi tingkat kepercayaan dari evidence. Pada teori Dempster-Shafer kita mengenal adanya frame of discernment yang dinotasikan dengan θ dan mass function yang

dinotasikan dengan m. Fungsi kombinasi m1 dan m2 sebagai m3 dibentuk dengan persamaan :

$$m3(Z) = \frac{\sum x \cap y = z\ m1(X).m2(Y)}{1-\sum x \cap y = \theta m1(X).m2(Y)}$$

II.6. Mesin VVT-i

Mesin berteknologi **VVT-i** (**Variable Valve Timing with intelligence**) adalah mesin berteknologi variable valve timing yang dikembangkan oleh Toyota. VVT-i menggantikan teknologi VVT Toyota yang sudah mulai diterapkan tahun 1991 di mesin Toyota *4A-GE* 5 silinder. Mesin yang sudah dipakai di sebagian besar mobil Toyota ini diklaim membuat mesin semakin efisien dan bertenaga, ramah lingkungan serta hemat bahan bakar.

VVT-i (sering disalahartikan dengan injeksi) bisa diterjemahkan dalam kalimat awam pengaturan pintar waktu buka tutup valve yang variatif. VVT-i diperkenalkan pada tahun 1996. Mobil Toyota Avanza dengan mesin berteknologi VVT-i pertama kali hadir di Indonesia pada tahun 2004.

II.7. Database

Database adalah sekumpulan tabel-tabel yang berisi data dan merupakan kumpulan dari *field* atau kolom. Struktur file yang menyusun sebuah database adalah *Data Record* dan *Field* (Anhar ; 2010 : 46).

a. Data adalah satu satuan informasi yang akan diolah. Sebelum diolah, data dikumpulkan di dalam suatu *file database*.

b. *Record* adalah data yang isinya merupakan satu kesatuan seperti NamaUser dan *Password.* Setiap keterangan yang mencakup NamaUser dan *Password* dinamakan satu *record*. Setiap *record* diberi nomor urut yang disebut nomor record *(Record Number).*

c. *Field* adalah sub bagian dari *record*. Dari contoh isi *record* di atas, maka terdiri dari 2 *field*, yaitu: *field* NamaUser dan *Password*

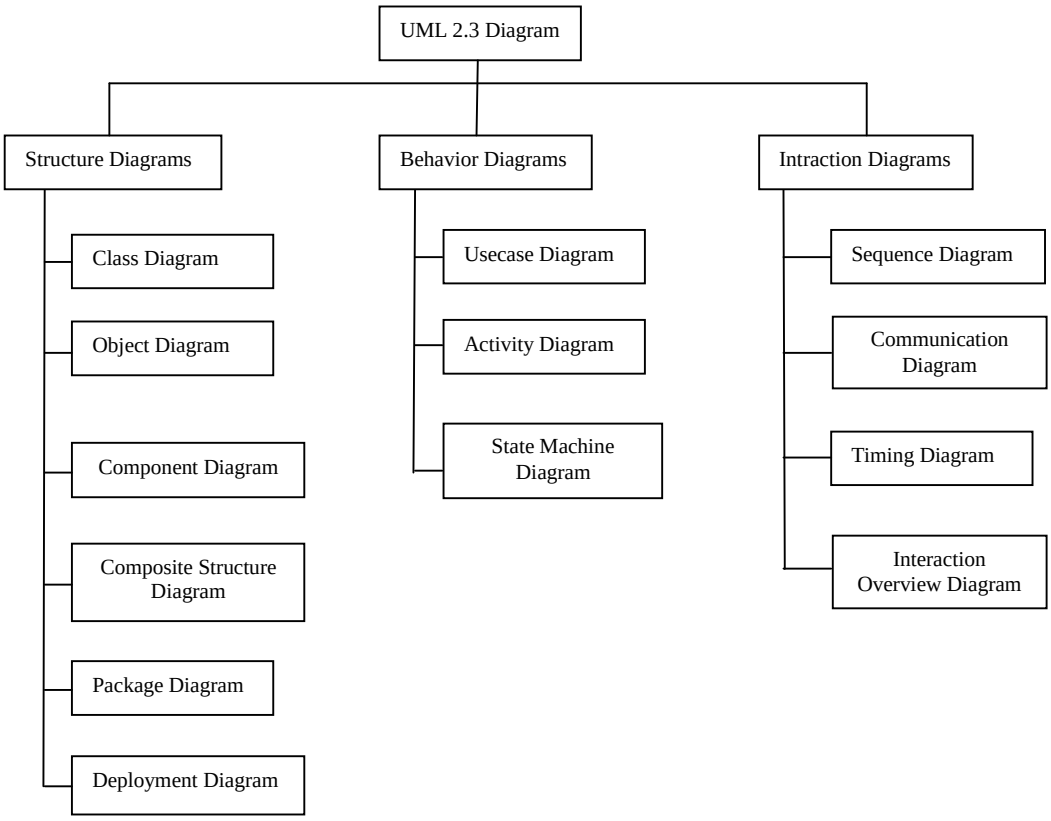
   Selain berisi data, *database* juga berisi *metadata*. Metadata adalah data yang menjelaskan tentang struktur dari data itu sendiri. Sebagai contoh, Anda dapat memperoleh informasi tentang nama-nama kolom dan tipe yang ditampilkan tersebut disebut *metadata.*

II.7.1. Pemodelan Data

Menurut **(**Yudi Priyadi : 2014 : 10) Terdapat beberapa penjelasanmengenai pemodelan basis data. Suatu basis data dapatdigunakan secara bebas untuk menggambarkan dan memberikan deskripsi mengenai kumpulan informasi uang tersimpan dalam *data storage* komputer. Secara sederhana, defenisi untuk model basis data adalah sekumpulan notasi atau simbol untuk menggambarkan data dan relasinya, berdasarkan suatu konsep dan aturan tertentu suatu permodelan.

II.8 Diagram-Diagram UML

Menurut (Rosa A.S & M. Shalahuddin ; 2011 : 120) Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar II.2 :



Gambar II.2 : Diagram UML

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 121)

Berikut ini penjelasan singkat dari pembagian kategori tersebut :

1. *Structure Diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

2. *Behavior Diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
3. *Interaction Diagrams* yaitu kumpulan diagram yang digunakan untukmenggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

A. *Class Diagram*

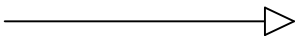
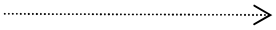
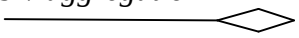
Diagram kelas atau *Class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem.

Kelas memiliki apa yang disebut atribut dan metode atau operasi.

1) Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
2) Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas

Berikut adalah simbol-simbol yang ada pada diagram kelas :

Simbol	Deskripsi
Kelas Nama_kelas +atribut +operasi()	Kelas pada struktur sistem
Antarmuka / *interface* Nama_*interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *association* Asosiasi berarah / *Directed association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity* Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi	Relasi antar kelas dengan makna

	generalisasi-spesialisasi (umum khusus)
Kebergantungan / *Dependency* 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi / *aggregation* 	Relasi antar kelas dengan makna semua bagian (*whole part*)

Gambar II. 3 : Diagram Kelas

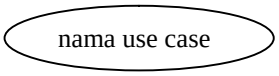
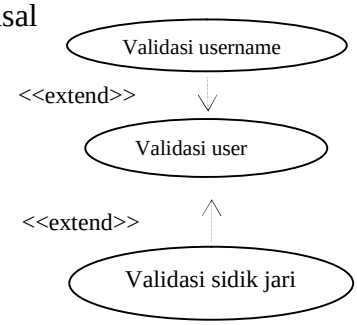
(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 123)

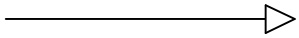
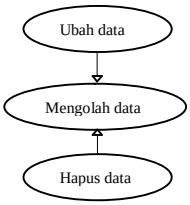
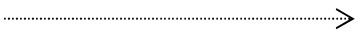
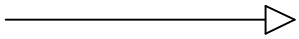
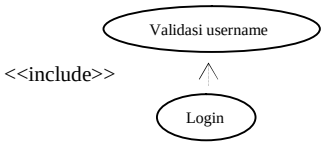
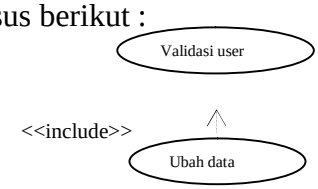
B. *Use Case Diagram*

Use case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behaviour)*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. Syarat penamaan pada *use case* adalah nama didefenisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefenisian apa yang disebut aktor dan *use case.*

1) Aktor merupakan 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.
2) *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Berikut adalah simbol-simbol yang ada pada diagram *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 frase nama *use case*
Aktor / *actor*  nama aktor	Orang, proses, atau sistem yang lain berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan di buat itu sendiri
Asosiasi / *association*	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case*, atau *usecase* memiliki interasi dengan aktor
Ekstensi / *extend* 	Relasi usecase tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanppa *use case* tambahan itu, mirip dengan prinsip inheritance pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan misal 

	arah panah mengarah pada *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 lainya misalnya :  Arah panah mengarah pada use case yang menjadi generalisasinya (umum)
Menggunakan / *include* / *uses* <<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 Ada 2 sudut pandang yang cukup besar mengenai include di usecase 1. include berarti use case yang ditambahkan akan selalu dipanggil saat use case dijalankan misal pada kasus berikut :  2. include berarti use case yang tambahan akan selalu melakukan pengecekan apakah use case yang di tambahkan telah di jalankan sebelum use case tambahan di jalankan, misal pada kasus berikut : 

	Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

Gambar II.4 : Diagram Use case

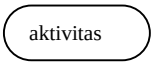
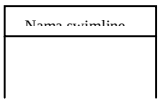
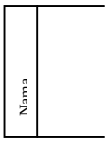
(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 131)

C. *Activity Diagram*

Diagram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Diagram aktivitas juga banyak digunakan untuk mendefenisikan hal-hal berikut :

1) Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sitemyang didefenisikan
2) Urutan atau pengelompokan tampilan dari sistem/user interface dimana setiap aktivitasdianggap memiliki sebuah rancangan antarmuka tampilan
3) Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefenisikan kasus ujinya.

Berikut adalah simbol-simbol yang ada pada diagram aktivitas :

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / decesion 	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 status akhir
Swimlane  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Gambar II.5 : Diagram Aktivitas

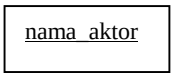
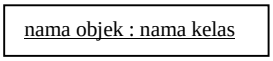
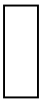
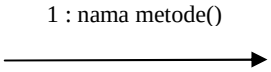
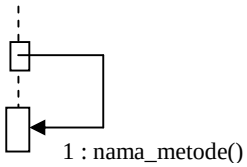
(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 134)

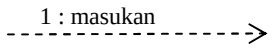
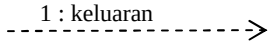
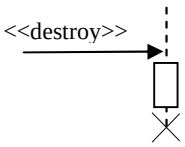
D. *Sequence Diagram*

Diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan messasge yang dikirimkan dan diterima antar objek. Banyaknya diagram objek yang digambarkan adalah sebanyak pendefinisian use case yang memiliki proses sendiri atau yang penting semua use case yang telah didefenisikan interaksi jalanya pesan sudah dicakup pada diagram

sekuen sehingga semakin banyak use case yang didefenisikan maka diagram sekuen yang harus dibuat juga semakin banyak.

Berikut adalah simbol-sombol yang ada pada diagram sekuen :

Simbol	Deskripsi
Aktor  atau  tampa waktu aktif	Orang, proses, atau sistem lain yang 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 di nyatakan menggunakan kata benda di awali frase nama aktor
Garis hidup / lifeline	Menyatakan kehidupan suatu objek
Objek 	Menyatakan objek yang berintaraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe create 	Objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tope call 	Menyatakan suatu objek memanggil operasi / metode yang ada pada objek lain atau dirinya sendiri,  Arah panah mengarah pada objek yang memiliki operasi / metode, karena ini memanggil operasi / metode maka operasi / metode yang di panggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi

Pesan tipe send 	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainya, arah panah mengarah pada objek yang dikirimi
Pesan tipe return 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasiatau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
Pesan tipe destroy 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy

Gambar II.6 : Diagram Squence

(Sumber : Rosa A.S & M. Shalahuddin ; 2011 : 138)